Georgia Institute of Technology
Faculty of Interactive Computing

CS 8803-DRL – ASSIGNMENT #2
Deep Reinforcement Learning

Instructor: Prof. Animesh Garg

Due date: 11/08/2024

Name: ______________________________

Student Number: ______________________________

This writing assignment contains 7 pages (including this cover page) and 17 questions. Total of points is 56. Good luck and Happy reading work!

The assignment must be completed before **11:59pm on Friday, November 8**, 2024. Make sure to submit all the components mentioned in Section 4

This assignment consists of two parts – a writing part and a coding part. For the theory question, your submission should be typeset in LaTeX. You are required to submit a PDF file containing your responses to the writing questions. For the coding question, you need to submit your code along with the running results, including the reward plot and a testing GIF file. Clearly specify which algorithm you are using to generate these plots.

The coding question is divided into two sections: the offline learning section in `hw2_offline.ipynb` and the model-based RL section in `hw2_mbrl.ipynb`. You can complete in whatever order you like except that Section 2 should not be completed before the coding assignment is completed.

Please ensure that you start this assignment as early as possible, as it will take time to run and fine-tune your code. The assignment must be completed by each student alone. Collaboration with other students is strictly prohibited. **Questions may be asked on Ed in case of unclear formulations. Do not post answers, partial answers or hints to answers!** Furthermore, we expect all students to adhere to the standards of academic integrity of the Georgia Institute of Technology.

1 Offline RL; Policy constraint to manage distribution shift (18 points)

1.1 Policy constraint and reward bonuses

Action distribution shift is one of the biggest issues in offline RL methods. In particular, when we use an *actor-critic (AC)* method, the target Q network gets queried on actions a' sampled from the current policy π. These actions a' can also be *out of distribution (OOD)*, or unseen samples. Since we train on the distribution defined by the behavior policy π_β that never sampled actions a', therefore using them in training can lead to erroneous (often over optimistic) Q values.

One way to address this problem is that of *policy constraint* – while training we make sure that the policy π being learned stays 'close' to the data generating policy π_β. This closeness constraint can be expressed and implemented in many ways. In this assignment we will explore some similarity metrics to measure the closeness of π and π_β, and how *reward bonuses* can be used to constrain the π close to π_β.

1.2 Example; Entropy regularization with reward bonuses

Consider an MDP $\mathcal{M} := (\mathcal{S}, \mathcal{A}, r, p, H)$ with state and action spaces $\mathcal{S}, \mathcal{A}$ respectively, reward function $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, transition function $p : \mathcal{S} \times \mathcal{A} \to \Delta\mathcal{S}$ and finite horizon H. We are given offline data $\mathcal{D}$ collected by some behavior policy π_β. We denote by p_π the distribution over states induced by π. As a motivating example, consider the *soft AC (SAC)* algorithm (discussed in HW 1). When updating, SAC adds an adjustment of $b(s,a) := -\log \pi(a|s)$ to the target values $\mathbb{E}_{\pi(a|s)}\left[Q(s,a) + b(s,a)\right]$ to enforce a maximum entropy regularizer on the policy. Alternatively, we could impose a similar form of entropy regularization by adding the bonus directly to the reward. In expectation, we would optimize

$$\begin{aligned}
\mathbb{E}_{s\sim p_\pi, a\sim\pi}[r(s,a) + \lambda b(s,a)] &= \frac{1}{H}V^\pi - \lambda \mathbb{E}_{s\sim p_\pi}\mathbb{E}_{a\sim\pi}\log\pi(a|s) \\
&= \frac{1}{H}V^\pi + \lambda \mathbb{E}_{s\sim p_\pi}\mathcal{H}[\pi(a|s)].
\end{aligned}$$

Thus, adding the bonus $b(s,a)$ to rewards is equivalent to regularization with entropy $\mathcal{H}$. In the following parts, you will show how a similar reward bonus can be used for policy constraint to address the action distribution shift in offline RL.

1.3 Constraining the policy with reward bonuses

We wish to learn a Q function and policy π from the offline data $\mathcal{D}$ under some constraint $D(\pi, \pi_\beta) \le \epsilon$ with the following update:

$$Q(s,a) \leftarrow r(s,a) + \mathbb{E}_{a'\sim\pi}[Q(s',a')], \tag{1}$$

$$\text{where} \quad \pi := \arg\max_\pi \mathbb{E}_{s\sim p_\pi, a\sim\pi}[Q(s,a)] \quad \text{s.t.} \quad D(\pi,\pi_\beta) \le \epsilon. \tag{2}$$

Directly enforcing the constraint in Eq. (2) is challenging with the environment rewards $r(s,a)$, so we will implicitly enforce the constraint with a Lagrangian, modifying the reward to $\bar{r}(s,a) \coloneqq r(s,a) + \lambda b(s,a)$ in Eq. (1). The overall optimization then becomes:

$$\begin{aligned} Q(s,a) &\leftarrow \bar{r}(s,a) + \mathbb{E}_{a' \sim \pi}[Q(s',a')], \\ \text{where} \quad \pi &\coloneqq \arg\max_{\pi} \mathbb{E}_{s \sim p_\pi, a \sim \pi}[Q(s,a)]. \end{aligned}$$

You may assume that $\lambda > 0$ is selected appropriately to enforce the constraint as follows:

$$\left(\arg\max_{\pi} \mathbb{E}_{s \sim p_\pi, a \sim \pi}[Q(s,a)] - \lambda D(\pi, \pi_\beta)\right) = \left(\arg\max_{\pi} \mathbb{E}_{s \sim p_\pi, a \sim \pi}[Q(s,a)] \quad \text{s.t.} \quad D(\pi, \pi_\beta) \leq \epsilon\right).$$

You may also assume access to the distributions $\pi(a|s)$ and $\pi_\beta(a|s)$ in your answers.

For each of the following questions, please clearly write your solution in clear and unambiguous steps. Also clearly provide your justification (and any comments, if necessary,) for each step in plain english. Only answers typed in LaTeXwill be accepted.

1. (5 points) Suppose we wish to learn π under a KL-divergence constraint, i.e.,

$$D(\pi, \pi_\beta) \coloneqq \mathbb{E}_{s \sim p_\pi} D_{\mathrm{KL}} \left[\pi\left(a|s\right) \| \pi_\beta\left(a|s\right)\right].$$

   How should we define $b(s,a)$ in order to enforce this constraint by adding the bonus to the reward $\bar{r}(s,a) \coloneqq r(s,a) + \lambda b(s,a)$?

2. (5 points) The f-divergence is a generalization of the KL-divergence that can be defined for distributions P and Q by

$$D_f[P\|Q] \coloneqq \int Q(x) f\left(\frac{P(x)}{Q(x)}\right) dx$$

   where f is a convex function with zero at 1. We can state an f-divergence policy constraint as

$$D(\pi, \pi_\beta) \coloneqq \mathbb{E}_{s \sim p_\pi} D_f \left[\pi(a|s) \| \pi_\beta(a|s)\right].$$

   How can you extend your answer from part (1) to account for an arbitrary f-divergence? Your answer should be a more general alternate expression for $b(s,a)$ in terms of f.

3. (8 points) Now we want to constrain divergence in the distribution of trajectories of states under π and π_β. We can express the KL divergence between the (state) trajectory distributions for $\tau \coloneqq (s_1, s_2, \ldots, s_H)$ as follows:

$$D(\pi, \pi_\beta) \coloneqq D_{\mathrm{KL}}[p_\pi(\tau) \| p_{\pi_\beta}(\tau)].$$

   What expression for $b(s,a)$ enforces this constraint? If you do not have access to the model $p(.|s,a)$ (which is the case in offline RL), can you still enforce this constraint?

2 Coding Assignment Questions (26 points)

Hopefully you didn't complete the coding assignment with ChatGPT! In this section we'll ask you some questions requiring you to reflect on what you built.

2.1 Behavior Cloning and Offline RL (12 points)

1. (3 points) Explain, with reference to specific characteristics of the dataset we trained on for this assignment, why you would expect an offline RL algorithm to achieve stronger performance than a behavior cloning algorithm.
2. (3 points) How does IQL avoid overestimation on OOD state-action pairs?
3. (3 points) What is the difference between the IQL policy update and the BC policy update and how does this difference lead the policy to choose better actions than BC?
4. (3 points) Can IQL ever learn to execute an actions better than those in the dataset? If so, how so and if not, why not?

2.2 Model-Based RL (14 points)

1. (3 points) In this homework you learned a good policy for the CartPole environment using only 26 demonstrations. Why is PETS so much more efficient than many model-free algorithms?
2. (5 points) I decided not to make you implement the cross entropy method optimizer (you're welcome) but it's still important to understand how it works, so please explain here how it is used to sample trajectories that minimize the expected cost. What do the `popsize` and `num_elites` parameters change?
3. (3 points) Give one reason that CEM is a better optimizer for this setting than an optimizer requiring gradients such as gradient descent?
4. (3 points) In order to compute costs we made a big assumption about our environment that we usually cannot make, especially in the real world. What assumption was that and how might you go about changing the algorithm to relax this assumption?

3 Multiple Choices Questions (12 points)

For each of the following multiple-choice questions, select the correct answer. After choosing an option, briefly explain your reasoning in 2-3 sentences. Your explanation should highlight why the selected option is correct and why the other options are not. Use examples where relevant to support your answer.

1. (2 points) What is the main goal of offline RL?

☐ a. Recover the policy π.

☐ b. Recover the optimal policy π^*.

☐ c. Recover the behavior policy π_β.

☐ d. Recover a policy that maximizes Q over $\mathcal{D}$.

☐ e. Recover a policy that minimizes regret over $\mathcal{M}$.

2. (2 points) In Section 1, we assumed access to π_β. If we want to do offline RL, how can we circumvent this assumption being false?

☐ a. Learn π using imitation learning.

☐ b. Estimate π_β using imitation learning.

☐ c. Learn π via offline RL without policy constraint.

☐ d. Estimate π_β via offline RL without policy constraint.

☐ e. None – not possible to learn π.

3. (2 points) **Both** CQL and IQL set out to address this issue with offline RL:

☐ a. Instability during online finetuning.

☐ b. Q overestimation on out of distribution actions

☐ c. Distribution shift.

☐ d. Sample inefficiency.

4. (2 points) What issue does the ensemble in the PETS dynamics model help to address?

☐ a. Epistemic uncertainty

☐ b. Aleatoric uncertainty

☐ c. MuJoCo simulator incompatible with Windows

☐ d. Sample efficiency

☐ e. Slow MPC inference

5. (2 points) What are possible issues with MPC-based model-based RL methods like PETS?

☐ a. Data inefficiency.

☐ b. Model inaccuracies over long prediction horizons.

☐ c. Worse asymptotic performance than model-free alternatives.

☐ d. Slow test-time inference speeds

6. (2 points) Vanilla model based policy optimization (MBPO) does the following.

☐ a. Augments the data with samples from the model.

☐ b. Uses pessimistic estimate of model predictions.

☐ c. Generates synthetic roll-outs.

☐ d. Uses uncertainty in model estimates.

☐ e. Uses expected value of model predictions.

4 Checklist for Assignment Submission

Congratulations on completing Assignment 2! We have created a checklist for you to double-check your submission. The files you need to submit are:

- ☐ **hw2.zip**: a zip file with your completed PDF and code.
  - ☐ **Jupyter Notebook files**:
    `hw2_offline.ipynb`, `hw2_mbrl.ipynb`
  - ☐ **Coding results**:
    - ∗ `bc_policy.gif`
    - ∗ `Behavior Cloning_returns.png`
    - ∗ `Implicit Q Learning_returns.png`
    - ∗ `iql_policy.gif`
    - ∗ `pets_policy.gif`
- ☐ **Writing assignment PDF file**: `CS8803_DRL_A2.pdf`
- ☐ **hw2_offline.html**:
  Convert the `hw2_offline.ipynb` to a HTML file **containing your running results.**
- ☐ **hw2_mbrl.html**:
  Convert the `hw2_mbrl.ipynb` to a HTML file **containing your running results.**
- ☐ **Completing the Multiple Choices Questions on Canvas.**