Homework 6

CMU 10–703: Deep Reinforcement Learning (Fall 2025)

OUT: October 30rd

DUE: November 14th by 11:59 pm ET

Instructions: START HERE

- **Collaboration policy:** You may work in groups of up to three people for this assignment. It is also OK to get clarification (but not solutions) from books or online resources after you have thought about the problems on your own. You are expected to comply with the University Policy on Academic Integrity and Plagiarism[1].
- **Late Submission Policy:** You are allowed a total of 10 grace days for your homeworks. However, no more than 2 grace days may be applied to a single assignment. Any assignment submitted after 2 days will not receive any credit. Grace days do not need to be requested or mentioned in emails; we will automatically apply them to students who submit late. We will not give any further extensions so make sure you only use them when you are absolutely sure you need them. See the Assignments and Grading Policy for more information about grace days and late submissions [2]
- **Submitting your work:**
  - **Gradescope:** Please write your answers and copy your plots into the provided LaTeX template, and upload a PDF to the GradeScope assignment titled "Homework 3." Additionally, export your code ([File → Export .py (if using Colab notebook)]) and upload it the GradeScope assignment titled "Homework 3: Code." Each team should only upload one copy of each part. Regrade requests can be made within one week of the assignment being graded.

[1] https://www.cmu.edu/policies/

[2] https://cmudeeprl.github.io/703website_f25/logistics/

Introduction

In this assignment we will explore how we can make value-based RL algorithms (in particular TD3 from Homework 3) work in the Offline RL setting - when we have access to a static dataset of environment interactions collected with some behavior policy, and no online access to the environment for learning. In particular, we will implement behavior regularization and Conservative Q-Learning.

Note - Make sure to take a look at the `ReadMe.md` to set up your conda environment for this homework assignment.

Problem 0: Collaborators

Please list your name and Andrew ID, as well as the names and Andrew IDs of your collaborators.

Problem 1: TD3 + Behavior Regularization (50 pts)

In this homework, we will study the Offline RL setting, in which we want to learn a policy given a static dataset of previous interactions with the environment $\mathcal{D} = \{(s, a, r, s')\}$ instead of learning by directly interacting with the environment. Our first step will be understanding what goes wrong if we simply keep using the same algorithms as before.

Problem 1.1: Loading the offline dataset into the replay buffer

The implementations in this homework will build on top of the TD3 code from Homework 3 (you should, however, start from the new starter code for Homework 6). We would like to run the critic and policy updates from TD3 using data uniformly sampled from the offline dataset (as opposed to from the online replay buffer normally used for TD3).

The offline dataset we are going to use comes from Minari, an Offline RL tasks library. In particular, we are going to use `hopper-medium-v0`, a classic Offline RL task. In `utils.py`, implement `get_buffer_and_environments_for_task`, which loads data from the offline dataset into the replay buffer, and creates the environment in which we will evaluate the policies we train.

Note: for this assignment, we will not collect any online data into the replay buffer (i.e., the only data we will have in the replay buffer will come from the static offline dataset). Therefore, the size of the replay buffer can simply be the number of timesteps in the offline dataset.

Problem 1.2: Running TD3 without any offline learning modifications

Run vanilla TD3 on the `hopper-medium-v0` task with the default hyperparameters, and plot the resulting learning curves including Evaluation Returns and predicted Q-Values for dataset state-action pairs.

In our reference implementation, we get evaluation returns of around 10 and Q-values of more than 3000 by step 50k. These trends of returns and Q-values give us a clue about what is going wrong here. What insights can we draw from evaluation returns being low but predicted Q-values being high? **Questions:**

Command (30 min):

```
python runner.py --agent td3 --total_steps 100000
```

Solution

Problem 1.3: Behavior Regularized TD3

Next, add a behavior-cloning loss to the TD3 policy objective. The policy objective from vanilla TD3 (which comes from DDPG) is given by:

$$\mathcal{L}_{\pi}^{(\text{DDPG})} = -\mathbb{E}_{s\sim\mathcal{D}}\left[Q_1(s, \mu_\theta(s))\right]. \tag{1}$$

We are going to modify it by adding a Behavior-Cloning term to encourage the policy to stay close to the actions seen in the offline dataset:

$$\mathcal{L}_{\pi}^{(\text{DDPG + BC-Reg})} = -\mathbb{E}_{s,a\sim\mathcal{D}}\left[Q_1(s, \mu_\theta(s))\right] + \alpha_{\text{reg}} \|a - \mu_\theta(s)\|_2^2. \tag{2}$$

Implement BC Regularization in `_td3_update_step` inside `td3_agent.py`, and train an agent with $\alpha_{\text{reg}} = 10.000$.

Our reference implementation gets evaluation returns of around 500 at 50k steps.

Solution

Problem 1.4: Conceptual question about picking α_{reg}

A key choice when implementing DDPG + Behavior Regularization is the choice of the BC-regularization coefficient α_{reg}. Answer the following questions:

1. The effect of the value of α_{reg} is environment and dataset dependent. For which kinds of **datasets** will we expect **larger** values of α_{reg} to work **better**? And conversely, for which datasets will we expect **lower** values of α_{reg} to work better?
2. We would like a systematic way of choosing α_{reg}. Let's say that you first complete a training run with $\alpha_{\text{reg}} = 1.0$, and you have access to any metric you require from this training run; how would you choose a reasonable value of α_{reg} if you could only run a single more run?

Solution

Problem 2: Conservative Q-Learning (50 pts)

The behavior-cloning regularization ued in Problem 1 constraints the policy to stay close to the actions seen in the dataset, so that the policy can't exploit value overestimations. However, the Q-function itself is left unconstrained, so we might still suffer from this problem to some extent. Conervative Q-Learning (CQL) addresses the issue of out-of-distribution over-estimation by **constraining the Q-function** instead of the policy. Please read the CQL paper before completing this problem.

Problem 2.1: Implementing the CQL loss

In `td3_agent.py _get_cql_loss`, follow the instructions to implement the CQL regularization loss. In particular, implement the following regularizer that will get added to the standard TD error objective:

$$\alpha\mathbb{E}_{s,a\sim\mathcal{D}}\left[\log(\sum_{a'\sim\text{uniform}}(\exp(Q(s,a'))) - Q(s,a)\right] \tag{3}$$

Where a' are actions sampled uniformly inside the action bounds of the environment. Then, run training with CQL alpha 5.

Note: Our reference implementation got around 1.500 returns. However, depending on the implementation some seeds might lead to poor performance. If you don't get at least 1.000, re-run with 2 more seeds.

Solution

Problem 3: Conceptual Questions on Offline (50 pts)

(i) Select all that apply and justify regarding the offline RL methods that apply regularization on the value function for training. For this question, consider three algorithms:

- Conservative Q-Learning: as we discussed in class, this algorithm applies a regularizer to minimize large Q-values during training.
- Reward penalty: similar to offline model-based RL methods, we penalize the reward value at *each* state-action pair by subtracting an uncertainty estimate from it.
- Policy constraint penalty: this approach subtracts $D_{\mathrm{KL}}(\pi_\theta, \pi_\beta)$ from the policy update and the value function update.

Select all that apply from below:

1. Conservative Q-learning is likely to perform better than policy constraints on tasks with offline datasets that exhibit higher coverage.
2. Reward penalty should perform better than conservative Q-learning methods because penalizing the Q-function at each unseen action will lead to more conservatism than using uncertainty estimates to penalize the reward function.
3. Conservative Q-learning is addressing problems of over-conservative behavior with policy constraint penalty methods.

(ii) In class, we studied several methods for offline RL based on policy constraints. Recall that there were two types of methods: one based on distributional constraints (e.g., optimize the Q-function subject to the KL-divergence between the learned policy and the behavior policy) and one based on support constraints (e.g., optimize the Q-function subject to a support constraint between the learned policy and the behavior policy). Describe a scenario where distribution constraint methods would be preferable to support constraints. *Hint:* in class, we discussed when support constraint methods will be better. Think about when that is not the case and answer this question.

(iii) Select all that apply and justify regarding the performance of model-based offline RL methods that we studied in class.

1. Model-based offline RL methods are generally better than model-free offline RL methods (that do not use a dynamic model), given any offline dataset.
2. Using distributional policy constraints within a model-based offline RL methods can be helpful when the distribution of the offline dataset is **quite narrow**.
3. Using distributional policy constraints within a model-based offline RL methods can be helpful when the distribution of the offline dataset is **quite broad and exhibits high coverage**.

(iv) Given a fixed static offline dataset of state-action-next state-reward transitions, why can offline RL (not filtered imitation learning) perform better than imitation learning? Give a qualitative example of a dataset where offline RL is expected to outperform imitation and one where imitation is expected to outperform offline RL.

Problem 4: Feedback

Feedback: You can help the course staff improve the course by providing feedback. What was the most confusing part of this homework, and what would have made it less confusing?

Time Spent: How many hours did you spend working on this assignment? Your answer will not affect your grade.

Alone	
With teammates	
With other classmates	
At office hours	