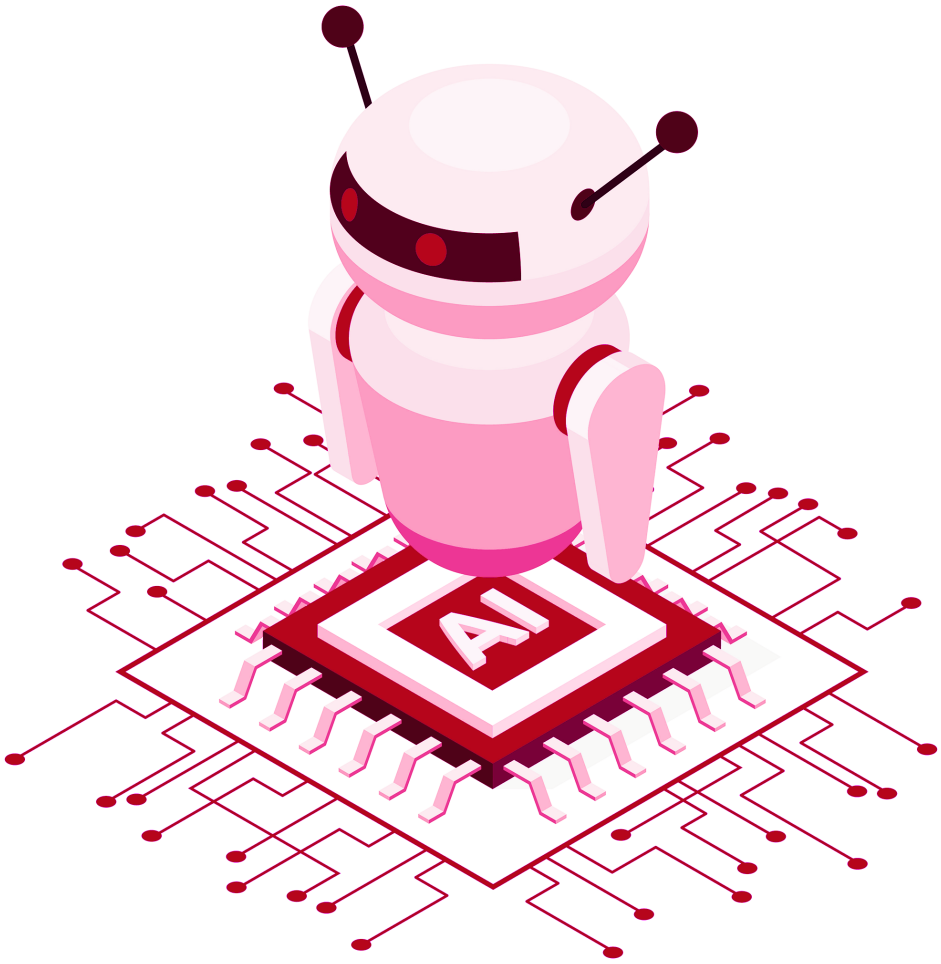

Deep Reinforcement Learning

Professor Mohammad Hossein Rohban

Solution for Homework 12:

Offline Methods

By:

Ashkan Majidi

400109984

RIML

Spring 2025

Contents

1 Part 1 [60-points]

1. Considering the Bellman update, explain with reasoning why value estimation suffers from overestimation in the offline framework. [10-points]

$$Q(s,a) \leftarrow r(s,a) + \gamma \mathbb{E}_{a' \sim \pi_{new}}[Q(s',a')]$$

Solution:
Value overestimation is primarily caused by distribution shift combined with function approximation error. The learning policy, π_{new}, queries the Q-function on actions that may be out-of-distribution (OOD) with respect to the fixed dataset. The key issues are:

- **Function Approximator Error:** For OOD actions, the Q-network's estimates are unreliable and can be arbitrarily high.
- **Maximization Bias:** The policy update, which seeks to maximize expected future rewards, or the explicit $\max$ operator in Q-learning, will preferentially select actions with these erroneously high Q-values. This turns random approximation errors into a systematic positive bias.
- **Bootstrapping:** This overestimation error is then bootstrapped into the Q-values of previous states during the update process, causing the error to propagate and leading to a flawed, overly optimistic value function.

2. One of the solutions to address the overestimation problem in the offline framework is CQL, whose objective function for computing the value is given below. Explain the role of each of the four terms in this objective function. [20-points]

$$\begin{aligned}\hat{Q}^T = \arg\min_Q \max_\mu \; & \alpha \mathbb{E}_{s\sim D, a\sim\mu(a|s)}[Q(s,a)] \\ & - \alpha \mathbb{E}_{(s,a)\sim D}[Q(s,a)] \\ & - \mathbb{E}_{s\sim D}[\mathcal{H}(\mu(\cdot|s))] \\ & + \mathbb{E}_{(s,a,s')\sim D}\left[(Q(s,a) - (r(s,a) + \gamma \mathbb{E}[Q(s',a')]))^2\right]\end{aligned}$$

Solution:
This objective function trains a Q-function that is both accurate according to the Bellman equation and "conservative" about the values of out-of-distribution actions. The overall structure is a min-max game where we minimize the Q-function against an adversarial policy μ that tries to find high-value actions. Let's analyze each term:

(a) **Term 1:** $\alpha \mathbb{E}_{s\sim D, a\sim\mu(a|s)}[Q(s,a)]$
This term represents the expected Q-value under a policy μ for states sampled from the dataset D. The outer optimization involves a $\max_\mu$, meaning we are searching for a policy μ that finds actions maximizing the Q-values. This term acts as an adversary, trying to push Q-values up. If an action is truly out-of-distribution but the Q-function assigns it a high value, μ will find it.

(b) **Term 2:** $-\alpha \mathbb{E}_{(s,a)\sim D}[Q(s,a)]$
This is the core *conservative regularization* term of CQL. It directly penalizes or pushes down the Q-values of the state-action pairs (s,a) that are present in the offline dataset D. By combining this with Term 1, the overall objective for the Q-function (which is a $\min_Q$) is to minimize the gap between the Q-values of actions found by the adversary μ and the Q-values of actions seen in the data. This effectively forces the learned Q-values for in-distribution actions to be higher than those for out-of-distribution actions, thus preventing overestimation.

(c) **Term 3:** $-\mathbb{E}_{s\sim D}[\mathcal{H}(\mu(\cdot|s))]$
This term is the negative entropy of the policy μ. Since the objective is to maximize over μ, maximizing the negative entropy is equivalent to minimizing the entropy. This encourages the adversarial policy μ to become more deterministic, focusing its probability mass on the single action that has the highest estimated Q-value for a given state s. This creates a stronger adversary that aggressively seeks out single points of overestimation. (Note: In some formulations, this term has a positive sign to encourage a stochastic, exploratory adversary).

(d) **Term 4:** $+\mathbb{E}_{(s,a,s')\sim D}\left[(Q(s,a)-(r(s,a)+\gamma\mathbb{E}[Q(s',a')]))^2\right]$
This is the standard **Mean Squared Bellman Error (MSBE)**. It is the cornerstone of any Q-learning method. This term ensures that the learned Q-function, despite the conservative regularization, still accurately reflects the transition dynamics and rewards observed in the dataset D. It anchors the Q-function to the ground truth provided by the Bellman equation, ensuring that the policy is still learning to maximize the actual returns. The expectation $\mathbb{E}[Q(s',a')]$ is typically taken over actions from the current policy, and a target network is used for stability.

3. Rewrite the optimization problem from part 2 as a minimization-only problem. [20-points]
Solution:
Note: entropy term considered positive in this solution.
The modified objective, with a positive entropy term, is:

$$\begin{aligned}\hat{Q}^T = \arg\min_Q \max_\mu \quad & \alpha\mathbb{E}_{s\sim D,a\sim\mu(a|s)}[Q(s,a)] - \alpha\mathbb{E}_{(s,a)\sim D}[Q(s,a)] \\ & + \mathbb{E}_{s\sim D}[\mathcal{H}(\mu(\cdot|s))] + \mathbb{E}_{(s,a,s')\sim D}\left[\text{MSBE}\right]\end{aligned}$$

We need to solve is:

$$\max_{\mu(\cdot|s)} \left(\alpha\mathbb{E}_{a\sim\mu(a|s)}[Q(s,a)] + \mathcal{H}(\mu(\cdot|s))\right)$$

We can solve this constrained optimization problem using the method of Lagrange multipliers. For a fixed s, let $\mu_a = \mu(a|s)$ and $Q_a = Q(s,a)$. The Lagrangian $\mathcal{L}$ is:

$$\mathcal{L}(\mu,\lambda) = \left(\alpha\sum_a \mu_a Q_a - \sum_a \mu_a \log\mu_a\right) - \lambda\left(\sum_a \mu_a - 1\right)$$

To find the maximum, we take the derivative of $\mathcal{L}$ with respect to μ_a and set it to zero:

$$\begin{aligned}\frac{\partial\mathcal{L}}{\partial\mu_a} &= \alpha Q_a - (\log\mu_a + 1) - \lambda = 0 \\ \log\mu_a + 1 &= \alpha Q_a - \lambda \\ \log\mu_a &= \alpha Q_a - \lambda - 1\end{aligned}$$

Exponentiating both sides gives us the form of the optimal policy:

$$\mu_a = \exp(\alpha Q_a - \lambda - 1) = \exp(\alpha Q_a)\cdot\exp(-\lambda - 1)$$

The term $\exp(-\lambda-1)$ is constant for all actions a. We can find its value by enforcing the probability

constraint $\sum_a \mu_a = 1$:

$$\sum_a \exp(\alpha Q_a) \cdot \exp(-\lambda - 1) = 1$$
$$\exp(-\lambda - 1) \sum_a \exp(\alpha Q_a) = 1$$
$$\exp(-\lambda - 1) = \frac{1}{\sum_{a'} \exp(\alpha Q_{a'})}$$

Substituting this back into the expression for μ_a, we get the optimal policy $\mu^*(a|s)$:

$$\mu^*(a|s) = \frac{\exp(\alpha Q(s,a))}{\sum_{a'} \exp(\alpha Q(s,a'))}$$

This is a **softmax distribution** over the scaled Q-values. It balances exploiting high Q-values with maintaining high entropy (being stochastic).
Now, we substitute this optimal policy μ^* back into the expression we were maximizing, $\alpha \mathbb{E}_{a\sim\mu^*}[Q(s,a)] + \mathcal{H}(\mu^*(s))$, to find its maximum value.
Let $Z(s) = \sum_{a'} \exp(\alpha Q(s,a'))$ be the partition function. From our derivation of μ^*, we know that $\log \mu^*(a|s) = \alpha Q(s,a) - \log Z(s)$. We can rearrange this to get $\alpha Q(s,a) = \log \mu^*(a|s) + \log Z(s)$.
Let's evaluate the expression:

$$\begin{aligned}
&\alpha \mathbb{E}_{a\sim\mu^*}[Q(s,a)] + \mathcal{H}(\mu^*(s)) \\
&= \alpha \sum_a \mu^*(a|s) Q(s,a) - \sum_a \mu^*(a|s) \log \mu^*(a|s) \\
&= \sum_a \mu^*(a|s)(\alpha Q(s,a) - \log \mu^*(a|s)) \\
&= \sum_a \mu^*(a|s)((\log \mu^*(a|s) + \log Z(s)) - \log \mu^*(a|s)) \quad \text{(substituting for } \alpha Q(s,a)\text{)} \\
&= \sum_a \mu^*(a|s) \log Z(s) \\
&= \log Z(s) \sum_a \mu^*(a|s) \\
&= \log Z(s) \cdot 1 \\
&= \log \left(\sum_a \exp(\alpha Q(s,a)) \right)
\end{aligned}$$

Finally, we replace the entire inner max_μ expression with the 'log-sum-exp' value we just derived, averaged over the states in the dataset D. This yields the complete, minimization-only objective function:

$$\begin{aligned}
\hat{Q}^T = \arg\min_Q \quad & \mathbb{E}_{s\sim D} \left[\log \left(\sum_a \exp(\alpha Q(s,a)) \right) \right] - \alpha \mathbb{E}_{(s,a)\sim D}[Q(s,a)] \\
& + \mathbb{E}_{(s,a,s')\sim D} \left[(Q(s,a) - (r(s,a) + \gamma \mathbb{E}_{a'\sim\pi(\cdot|s')}[Q_{target}(s',a')]))^2 \right]
\end{aligned}$$

4. To apply this method in model-based reinforcement learning, what changes are needed in the objective function? Rewrite the new objective function. [10-points] **Solution:**
In model-based reinforcement learning (MBRL), we first learn a model of the environment's dynamics, $\hat{P}(s'|s,a)$, and potentially the reward function, $\hat{r}(s,a)$, from the offline dataset D. The key idea of applying CQL in an MBRL setting (as in algorithms like COMBO) is to use this learned model to generate the targets for the Bellman update, while still using the CQL regularizer to prevent value overestimation on actions outside the original dataset.
The primary change is in the **Mean Squared Bellman Error (MSBE) term**. Instead of calculating the Bellman target using next states s' drawn directly from the dataset D, we use next states s' sampled from our learned dynamics model $\hat{P}$.
The new objective function would be:

$$\hat{Q}^T = \arg\min_Q \underbrace{\alpha \left(\mathbb{E}_{s\sim D}[\log \sum_a \exp(Q(s,a))] - \mathbb{E}_{(s,a)\sim D}[Q(s,a)] \right)}_{\text{CQL Regularizer (Unchanged)}} + \underbrace{\mathbb{E}_{(s,a)\sim D}\left[(Q(s,a) - (\hat{r}(s,a) + \gamma \mathbb{E}_{s'\sim \hat{P}(\cdot|s,a), a'\sim\pi(\cdot|s')}[Q_{target}(s',a')]))^2 \right]}_{\text{Model-Based Bellman Error (Changed)}}$$

References

[1] Cover image designed by freepik