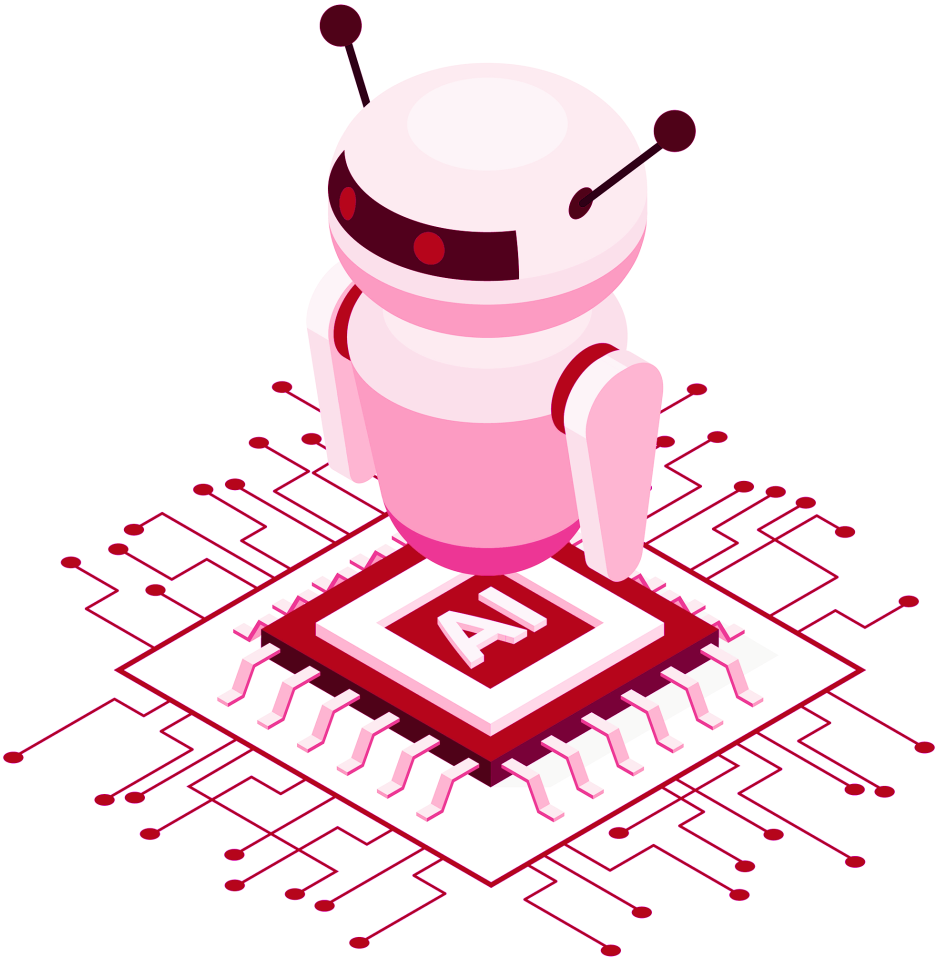

Deep Reinforcement Learning

Professor Mohammad Hossein Rohban

Solution for Homework 12:

Offline Methods

By:

Ayeen Poostforoushan

401105742

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]

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

In offline reinforcement learning, value estimation suffers from overestimation because of the Bellman update. The deep function approximator for Q-values is trained to minimize the estimated average, but due to sample inefficiency and OOD state-action pairs, it causes overshootings in some specific state/actions. The key problem is the maximization in the policy when selecting actions $a' - \pi_{\text{new}}$ chooses actions with the higher Q-values for the next state s'. This creates a positive feedback loop the Bellman update uses these overestimations, increasing the value function in subsequent updates more. Since there's no environmental feedback to correct errors in offline RL, this overestimation propagates unchecked. The issue gets worse with smaller datasets as mentioned in the lecture, as more sparse buffers have even less feedback that reduce the overestimation.

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]

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

(a) In this objective function, we are choosing the Q function which have the lowest policy that maximizes the expected Q-value of the dataset's state,actions. So in the first term, we are maximizing the expected q-value by choosing the policy which exploits the overshooting Q-values the most. So by minimizing the Q for this expression, we are choosing the Q which has the least overshoot over the whole state actions so there is no policy exploiting it.

(b) Because of the first term, even in the regions of the dataset that there is enough data, we still choose the worst Q values. But we don't need to be pessimistic in those regions, only the low-sample regions are needed. So we add this term to fix this for the higher-sample regions.

(c) This term is added to add entropy for the maximized policy for the Q-function so it chooses a softer max over the overshooted action values.

(d) This is the normal function that makes the Q value to be closer to the sampled target by the reward + old policy on the next state. (Which initially led to overestimation)

3. Rewrite the optimization problem from part 3 as a minimization-only problem. [20-points]

We solve the inner maximization over μ by finding its closed-form solution. The μ-dependent terms are:

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

where $\mathcal{H}(\mu) = -\sum_a \mu(a|s) \log \mu(a|s)$. For each s, we have:

$$\max_{\mu(\cdot|s)} \left\{ \alpha \sum_a \mu(a|s) Q(s,a) + \sum_a \mu(a|s) \log \mu(a|s) \right\}$$

Take the derivative w.r.t. $\mu(a|s)$ (and lagrange multipliers for $\sum_a \mu(a|s) = 1$):

$$\frac{\partial}{\partial \mu(a|s)} \left[\alpha \mu(a|s) Q(s,a) + \mu(a|s) \log \mu(a|s) + \lambda \left(1 - \sum_{a'} \mu(a'|s) \right) \right] = 0$$

So we have:

$$\alpha Q(s,a) + \log \mu(a|s) + 1 + \lambda = 0$$

Then we rearrange it:

$$\mu(a|s) = \exp\left(-\alpha Q(s,a) - 1 - \lambda\right)$$

Then we normalize to get a distribution:

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

The maximum value is this

$$\log \sum_a \exp(\alpha Q(s,a))$$

Which is the log-sum-exp. We put it back into the objective and we have:

$$\hat{Q}^T = \arg\min_Q \left\{ \mathbb{E}_{s\sim D} \left[\log \sum_a \exp(\alpha Q(s,a)) \right] - \alpha \mathbb{E}_{(s,a)\sim D}[Q(s,a)] \right.$$
$$\left. + \mathbb{E}_{(s,a,s')\sim D} \left[\left(Q(s,a) - (r(s,a) + \mathbb{E}_{a'\sim\pi_{old}}[Q(s',a')]) \right)^2 \right] \right\}$$

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]

   To use Conservative Q-Learning (CQL) for model-based reinforcement learning, the objective function must use the a learned dynamics model (translation and reward) to generate states for conservative regularization. Instead of regularizing only on dataset states D, we use states from model rollouts M, making sure that Q-values don't overshoot even for OOD states predicted by the model. The Bellman error term remains only for the real dataset D to maintain the Q-values of real transitions.

   The modified objective function is:

$$\hat{Q}^T = \arg\min_Q \left\{ \mathbb{E}_{s\sim M} \left[\log \sum_a \exp(\alpha Q(s,a)) \right] - \alpha \mathbb{E}_{(s,a)\sim D}[Q(s,a)] \right.$$
$$\left. + \mathbb{E}_{(s,a,s')\sim D} \left[\left(Q(s,a) - (r(s,a) + \mathbb{E}_{a'\sim\pi_{old}}[Q(s',a')]) \right)^2 \right] \right\}$$

   This way of adding our learned model to the objective is from the COMBO paper.

References

[1] Cover image designed by freepik