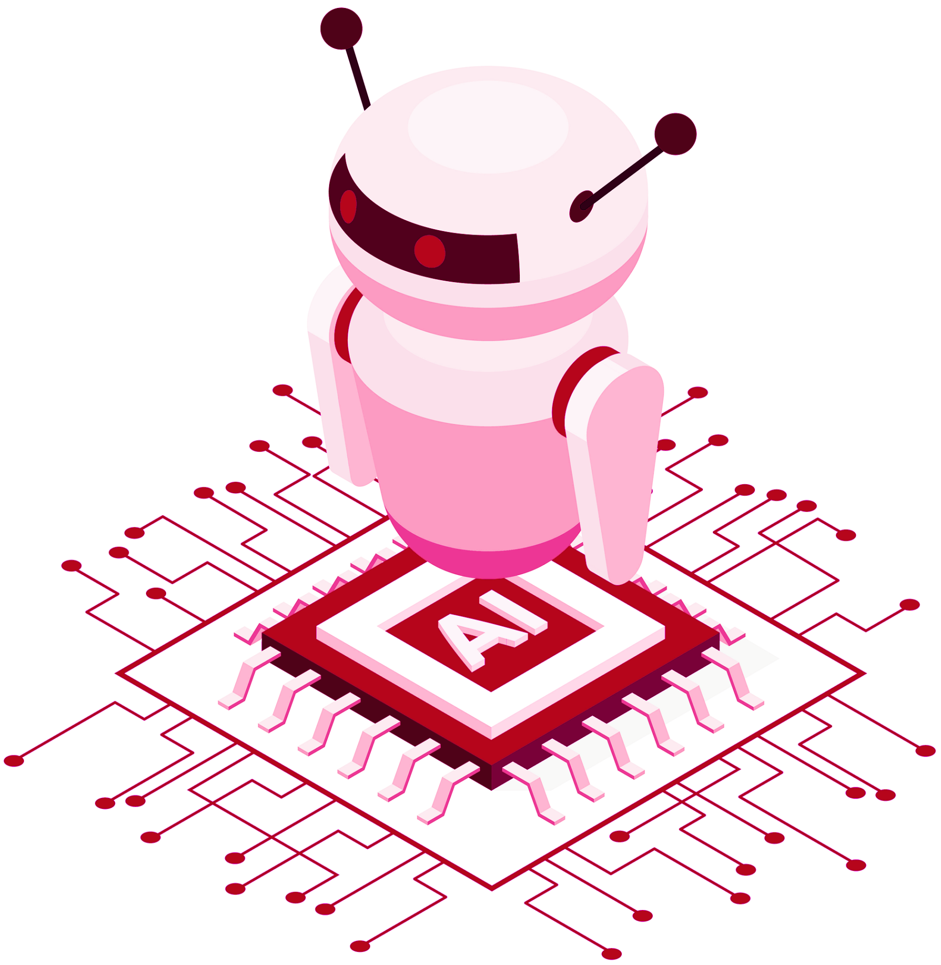

Deep Reinforcement Learning

Professor Mohammad Hossein Rohban

Solution for Homework 12:

Offline Methods

By:

Payam Taebi

400104867

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]

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

ANSWER

In the offline RL setting, we perform Bellman updates of the form

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

where r_{new} is some policy (possibly greedy) derived from our current Q-function. Overestimation arises due to two main factors:

(a) **Maximization bias / distributional shift:** Offline RL uses a fixed dataset $\mathcal{D} = \{(s, a, r, s')\}$ collected by some behavior policy μ. When we bootstrap, we sample a' from a derived policy r_{new}, which typically places more mass on actions with high estimated Q-values (e.g. greedy or near-greedy). Because our estimates $\hat{Q}$ contain approximation errors, the $\arg\max$ (or high-probability region) of $\hat{Q}$ tends to select actions whose errors are positively biased. Concretely:

$$\max_{a'} \hat{Q}(s', a') = \max_{a'}\big[Q^*(s', a') + \underbrace{\epsilon(s', a')}_{\text{approx. error}}\big] \approx Q^*(s', a^*) + \max_{a'} \epsilon(s', a'),$$

and $\max_{a'} \epsilon(s', a') \geq 0$ in expectation if errors are zero-mean but with variance. Hence, bootstrapping via a "max" (or near-max) induces an upward bias.

(b) **Lack of corrective feedback on out-of-distribution actions:** In online RL, if the learner overestimates $Q(s', a')$ for some action a', it will eventually try a', observe the true reward and next state, and correct the error. In offline RL, however, the dataset $\mathcal{D}$ contains no or few samples for these overestimated (out-of-distribution) actions. Thus, the erroneous high values for $Q(s', a')$ are never suppressed by real data, and the overestimation compounds over multiple Bellman backups.

Putting these together, each Bellman update

$$Q_{k+1}(s, a) \;=\; r(s, a) + \mathbb{E}_{a' \sim r_{\text{new}}}\big[Q_k(s', a')\big]$$

systematically injects a positive bias whenever $\mathbb{E}_{a'}[Q_k(s', a')]$ is computed over actions with uncertain (and upward-biased) estimates, and those biases are never corrected offline. Over iterations, this leads to a growing overestimation of the true Q^*-values.

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]

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

ANSWER

The Conservative Q-Learning (CQL) objective is:

$$\hat{Q}^T = \arg\min_Q \max_\mu \Big[\underbrace{\alpha\,\mathbb{E}_{s\sim D,\,a\sim\mu(\cdot|s)}\big[Q(s,a)\big]}_{\text{(1) Increase values on policy-sampled actions}} - \underbrace{\alpha\,\mathbb{E}_{(s,a)\sim D}\big[Q(s,a)\big]}_{\text{(2) Decrease values on dataset actions}}$$
$$- \underbrace{\mathbb{E}_{s\sim D}\big[\mathcal{H}(\mu(\cdot|s))\big]}_{\text{(3) Entropy regularization of auxiliary policy}} + \underbrace{\mathbb{E}_{(s,a,s')\sim D}\Big[\Big(Q(s,a) - \big(r(s,a) + \mathbb{E}_{a'\sim r_{\text{new}}}\big[Q(s',a')\big]\big)\Big)^2\Big]}_{\text{(4) Bellman-error (TD-consistency)}}\Big].$$

Below is a detailed explanation of each term:

(1) $\alpha\,\mathbb{E}_{s\sim D,\,a\sim\mu}[Q(s,a)]$**:**

- *Purpose:* Penalize high Q-values on actions that the learned policy μ might choose, especially those not well supported by the dataset.
- *Mechanism:* During the inner maximization, μ will concentrate on actions with large $Q(s,a)$. By *increasing* the objective when $Q(s,a)$ is large under μ, the outer minimization pushes Q downward on those potentially overestimated actions.
- *Effect on overestimation:* Discourages the learner from assigning artificially high value to out-of-distribution (OOD) actions, since those actions become "expensive" in the optimization.
- *Role of α:* Controls the strength of conservatism—the larger α, the stronger the penalty on OOD actions.

(2) $-\alpha\,\mathbb{E}_{(s,a)\sim D}[Q(s,a)]$**:**

- *Purpose:* Provide a counter-balance so that not all Q-values trivially collapse to the same constant.
- *Mechanism:* Subtracts the average Q-value over the dataset D. If Q were uniformly high, this term would also be large (negative), discouraging such collapse.
- *Effect on conservatism:* Together with term (1), enforces that dataset actions have relatively higher Q-values than OOD actions, instilling a lower bound on the value of seen actions.

(3) $-\mathbb{E}_{s\sim D}\big[\mathcal{H}(\mu(\cdot|s))\big]$**:**

- *Purpose:* Prevent the auxiliary policy μ from becoming too narrow (deterministic) and only selecting the single highest-valued action.
- *Mechanism:* The entropy $\mathcal{H}(\mu(\cdot|s))$ is maximized when μ is uniform; subtracting its expectation encourages μ to maintain randomness.
- *Effect on training:* Ensures the maximization over μ explores a variety of actions, so the conservatism penalty (term 1) applies broadly to all OOD actions rather than just one.

(4) $\mathbb{E}_{(s,a,s')\sim D}\big[\big(Q(s,a) - \big(r(s,a) + \mathbb{E}_{a'\sim r_{\text{new}}}[Q(s',a')]\big)\big)^2\big]$**:**

- *Purpose:* Anchor the learned Q to the actual rewards and transitions observed in the dataset.

- *Mechanism:* This is the standard squared Temporal-Difference (TD) error. It forces Q to satisfy the Bellman equation with respect to the behavior/new policy r_{new}.
- *Effect on learning:* Guarantees that, despite the conservative regularization, Q still fits the observed data; without this term, Q could drift arbitrarily.

Combined effect: - Terms(1) and (2) form a conservatism regularizer that *widens* the gap between dataset actions and potential OOD actions, directly counteracting overestimation. - Term(3) ensures that this penalty covers a *diverse* set of actions rather than collapsing to a single one. - Term(4) keeps the learned Q values *grounded* in the actual offline data, preserving Bellman consistency.

Together, these four terms yield a Q-function that is both *conservative* against extrapolation error and *accurate* with respect to the offline dataset.

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

ANSWER

We begin with the original saddle-point formulation from part 3:

$$\hat{Q}^T = \arg\min_{Q} \max_{\mu} L(Q,\mu),$$

where

$$L(Q,\mu) = \underbrace{\alpha\,\mathbb{E}_{s\sim D,\, a\sim\mu(\cdot|s)}\big[Q(s,a)\big]}_{(1)} - \underbrace{\alpha\,\mathbb{E}_{(s,a)\sim D}\big[Q(s,a)\big]}_{(2)}$$
$$- \underbrace{\mathbb{E}_{s\sim D}\big[\mathcal{H}(\mu(\cdot|s))\big]}_{(3)} + \underbrace{\mathbb{E}_{(s,a,s')\sim D}\Big[\delta(s,a,s')^2\Big]}_{(4)},$$

and we write the Bellman-error term as

$$\delta(s,a,s') = Q(s,a) - \Big(r(s,a) + \mathbb{E}_{a'\sim r_{\text{new}}}\big[Q(s',a')\big]\Big).$$

—

Converting the max–min to a joint minimization

1. **Flip the inner maximization** Recall that

$$\max_{\mu} L(Q,\mu) = -\min_{\mu}\big[-L(Q,\mu)\big].$$

Hence the original problem becomes

$$\hat{Q}^T = \arg\min_{Q}\Big(\max_{\mu} L(Q,\mu)\Big) = \arg\min_{Q}\Big[-\min_{\mu}\big(-L(Q,\mu)\big)\Big].$$

Since $\min_Q[-\min_\mu(\cdot)] = \min_{Q,\mu}[-L(Q,\mu)]$, we have

$$\hat{Q}^T = \arg\min_{Q,\mu}\big[-L(Q,\mu)\big].$$

2. **Write out $-L(Q,\mu)$** Substituting term by term,

$$\begin{aligned}-L(Q,\mu) &= -\big[(1)\big] + \big[(2)\big] + \big[(3)\big] - \big[(4)\big] \\ &= -\,\alpha\,\mathbb{E}_{s\sim D,\,a\sim\mu(\cdot|s)}\big[Q(s,a)\big] \\ &\quad+\,\alpha\,\mathbb{E}_{(s,a)\sim D}\big[Q(s,a)\big] \\ &\quad+\,\mathbb{E}_{s\sim D}\big[\mathcal{H}(\mu(\cdot|s))\big] \\ &\quad-\,\mathbb{E}_{(s,a,s')\sim D}\Big[\delta(s,a,s')^2\Big].\end{aligned}$$

3. **Define the joint objective** Let

$$\begin{aligned}J(Q,\mu) \;=\; &-\alpha\,\mathbb{E}_{s\sim D,\,a\sim\mu(\cdot|s)}\big[Q(s,a)\big] \\ &+\alpha\,\mathbb{E}_{(s,a)\sim D}\big[Q(s,a)\big] \\ &+\mathbb{E}_{s\sim D}\big[\mathcal{H}(\mu(\cdot|s))\big] \\ &-\mathbb{E}_{(s,a,s')\sim D}\Big[\delta(s,a,s')^2\Big].\end{aligned}$$

Then the minimization-only form is simply

$$\boxed{\hat{Q}^T \;=\; \arg\min_{Q,\mu}\; J(Q,\mu).}$$

—

Interpretation of the minimization-only form

- **First term** $-\alpha\,\mathbb{E}_{s\sim D,a\sim\mu}[Q(s,a)]$ now appears as a *negative* expectation, so minimizing it pushes Q-values *down* on actions the auxiliary policy μ prefers.

- **Second term** $+\alpha\,\mathbb{E}_{(s,a)\sim D}[Q(s,a)]$ still raises Q on dataset actions, preventing collapse to a trivial solution.

- **Third term** $+\mathbb{E}_{s\sim D}[\mathcal{H}(\mu(\cdot|s))]$ remains an entropy bonus, encouraging μ to stay stochastic.

- **Fourth term** $-\mathbb{E}_{(s,a,s')\sim D}[\delta(s,a,s')^2]$ becomes a *negative* Bellman-error. Since we minimize J, this forces the squared TD error δ^2 to be *small*, preserving Bellman consistency.

By combining all four into one joint minimization, we remove the inner maximization over μ and arrive at a single objective $J(Q,\mu)$ to optimize.

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]

ANSWER

To turn CQL into a model-based variant, we replace all uses of the empirical transition/reward in the Bellman-error term with samples from a learned dynamics model $\hat{P}(s'|s,a)$ and reward model $\hat{r}(s,a)$. Concretely, the only change is in term (4): instead of

$$\mathbb{E}_{(s,a,s')\sim D}\Big[\big(Q(s,a) - \big(r(s,a) + \mathbb{E}_{a'\sim\mu}[Q(s',a')]\big)\big)^2\Big],$$

we use

$$\mathbb{E}_{(s,a)\sim D}\, \mathbb{E}_{s'\sim\hat{P}(\cdot|s,a)}\Big[\big(Q(s,a) - \big(\hat{r}(s,a) + \gamma\, \mathbb{E}_{a'\sim\mu}[Q(s',a')]\big)\big)^2\Big].$$

Putting it all together, the model-based CQL objective becomes

$$\begin{aligned}\hat{Q}^T = \arg\min_Q \max_\mu \Big[&\underbrace{\alpha\, \mathbb{E}_{s\sim D,\, a\sim\mu(\cdot|s)}\big[Q(s,a)\big]}_{(1)} - \underbrace{\alpha\, \mathbb{E}_{(s,a)\sim D}\big[Q(s,a)\big]}_{(2)} \\ &- \underbrace{\mathbb{E}_{s\sim D}\big[\mathcal{H}(\mu(\cdot|s))\big]}_{(3)} \\ &+ \underbrace{\mathbb{E}_{(s,a)\sim D}\, \mathbb{E}_{s'\sim\hat{P}(\cdot|s,a)}\Big[\big(Q(s,a) - \big(\hat{r}(s,a) + \gamma\, \mathbb{E}_{a'\sim\mu}[Q(s',a')]\big)\big)^2\Big]}_{(4)'}\Big].\end{aligned}$$

Here:

- Terms (1)–(3) remain unchanged.
- Term (4)′ uses model-generated next-states $s' \sim \hat{P}$ and rewards $\hat{r}(s,a)$, and includes the discount factor γ.

References

[1] Cover image designed by freepik