Aniket Sanjay Gujarathi
1007629979

Assignment 2

CSC 2626
October 21, 2022

Conservative Q-Learning [1]

Motivation

One of the main issues in offline RL is the problem of overestimation of values due to the distributional shift between the dataset and the learned policy. This over-estimation could be the cause of out-of-distribution actions, finite sample error, and function approximation error. This results in poor policies that do not help the model to work well due to bootstrapping from out-of-distribution actions and overfitting. In [1], the authors propose a Q-learning method that addresses this problem of distributional shift by learning a conservative Q-function such that the expected value of the policy under this Q-function lower bounds its true value.

A safe strategy to handle this distributional shift is to be conservative i.e. if we explicitly assign a low value to unseen actions, then the estimated value of the policy for unseen behaviors would be low. Thus, the optimization will prevent the policy from executing unseen actions and work reliably. The key idea is to minimize the values under an appropriately chosen distribution over state-action tuples and then further tighten the bound by adding a maximization term over the data distribution.

Algorithm Summary

Conservative Off-Policy Evaluation

The main aim of the algorithm is to estimate the value $V^\pi(s)$ of a target policy π given access to a dataset D, generated by following an expert policy $\pi_\beta(a|s)$. But as we are trying to learn a conservative Q-function, the cql framework learns the Q-function using a sum of two objective functions - a standard Bellman error objective and a regularizer that minimizes the Q-values on unseen actions and maximizes the expected Q-value on the dataset.

$$\hat{Q}^{k+1} \leftarrow \arg\min_{Q} \; \alpha \cdot \left(\mathbb{E}_{\mathbf{s}\sim\mathcal{D},\mathbf{a}\sim\mu(\mathbf{a}|\mathbf{s})}\left[Q(\mathbf{s},\mathbf{a})\right] - \mathbb{E}_{\mathbf{s}\sim\mathcal{D},\mathbf{a}\sim\hat{\pi}_\beta(\mathbf{a}|\mathbf{s})}\left[Q(\mathbf{s},\mathbf{a})\right]\right) + \frac{1}{2}\,\mathbb{E}_{\mathbf{s},\mathbf{a},\mathbf{s}'\sim\mathcal{D}}\left[\left(Q(\mathbf{s},\mathbf{a}) - \hat{\mathcal{B}}^\pi \hat{Q}^k(\mathbf{s},\mathbf{a})\right)^2\right]$$

The second term in the equation above, refers to the standard Bellmann error and the first term consists of the minimization term for the unseen actions (colored black) and the maximization term (colored red) that assures a tighter bound.

Aniket Sanjay Gujarathi
1007629979

Assignment 2

CSC 2626
October 21, 2022

Conservative Q-Learning for offline RL

A general policy for off-policy learning is discussed here. Since the policy $\hat{\pi}^k$ is derived from the Q-function, we could instead choose $\mu(a|s)$ to approximate the policy that would maximize the current Q-function iterate.

$$\min_Q \max_\mu \; \alpha \left(\mathbb{E}_{\mathbf{s}\sim\mathcal{D},\mathbf{a}\sim\mu(\mathbf{a}|\mathbf{s})} \left[Q(\mathbf{s},\mathbf{a}) \right] - \mathbb{E}_{\mathbf{s}\sim\mathcal{D},\mathbf{a}\sim\hat{\pi}_\beta(\mathbf{a}|\mathbf{s})} \left[Q(\mathbf{s},\mathbf{a}) \right] \right) + \frac{1}{2} \mathbb{E}_{\mathbf{s},\mathbf{a},\mathbf{s}'\sim\mathcal{D}} \left[\left(Q(\mathbf{s},\mathbf{a}) - \hat{\mathcal{B}}^{\pi_k} \hat{Q}^k(\mathbf{s},\mathbf{a}) \right)^2 \right] + \mathcal{R}(\mu) \quad (\text{CQL}(\mathcal{R}))$$

(**Q1**) Which equation in the paper was used to implement the CQL update rule in the reference implementation?

The reference implementation uses a variant of CQL for generality. Particularly, the following equation has been used:

$$\min_Q \; \alpha \mathbb{E}_{\mathbf{s}\sim\mathcal{D}} \left[\log \sum_{\mathbf{a}} \exp(Q(\mathbf{s},\mathbf{a})) - \mathbb{E}_{\mathbf{a}\sim\hat{\pi}_\beta(\mathbf{a}|\mathbf{s})} \left[Q(\mathbf{s},\mathbf{a}) \right] \right] + \frac{1}{2} \mathbb{E}_{\mathbf{s},\mathbf{a},\mathbf{s}'\sim\mathcal{D}} \left[\left(Q - \hat{\mathcal{B}}^{\pi_k} \hat{Q}^k \right)^2 \right]$$

(**Q2**) How was this equation derived in the paper?

For the derivation, as explained in Appendix A of [1], we substitute $R = H(\mu)$ and solve the optimization over μ in a closed form over the given Q-function. The optimal solution is of the form $\mu^*(x) = \frac{1}{Z} exp(f(x))$ where Z is the normalization factor.

(**Q3**) How is the equation above implemented for discrete systems and continuous systems?

For discrete action spaces, $CQL(H)$ uses log-sum-exp in the objective for training the Q-function. The standard $tf.reduce\_logsumexp()$ (or $torch.logsumexp()$) functions provided by autodiff libraries provide an easy way to compute this. In continuous action tasks, $CQL(H)$ uses importance sampling.

(**Q4**) Do the CQL lower bounds hold upon convergence of the CQL update rule, or also during each Q iteration?

CQL provides a lower bound if α is larger than some threshold, so as to overcome the sampling error or the over-sampling error. According to proof of theorem 3.1 in [1], it is proved that at each iterate the next Q-value is underestimated. Hence CQL lower bounds hold true for convergence as well as the during each Q-iteration.