Proceedings of Machine Learning Research vol 236:499–513, 2024 3rd Conference on Causal Learning and Reasoning

Implicit and Explicit Policy Constraints for Offline Reinforcement Learning

Yang Liu LIUYANG1123@CONNECT.HKU.HK
Marius Hofert MHOFERT@HKU.HK
Department of Statistics and Actuarial Science, The University of Hong Kong, Hong Kong, China

Editors: Francesco Locatello and Vanessa Didelez

Abstract

Offline reinforcement learning (RL) aims to improve the target policy over the behavior policy based on historical data. A major problem of offline RL is the distribution shift that causes overestimation of the Q-value due to out-of-distribution actions. Most existing works focus on either behavioral cloning (BC) or maximizing Q-Learning methods to suppress distribution shift. BC methods try to mitigate the shift by constraining the target policy to be close to the offline data, but it makes the learned policy highly conservative. On the other hand, maximizing Q-Learning methods adopt pessimism mechanism to generate actions by maximizing Q-value and penalizing Q-value according to the uncertainty of actions. However, the generated actions might be arbitrary, resulting in the predicted Q-values highly uncertain, which will in turn misguide the policy to generate the next action. To alleviate the adverse effect of the distribution shift, we propose to constrain the policy implicitly and explicitly by unifying Q-Learning and behavior cloning to tackle the exploration and exploitation dilemma. For the implicit constraint approach, we propose to unify the action space by generative adversarial networks that dedicate to make the actions of the target policy and behavior policy indistinguishable. For the explicit constraint approach, we propose multiple importance sampling (MIS) to learn an advantage weight for each state-action pair which is then used to suppress or make full use of each state-action pair. Extensive experiments on the D4RL dataset indicate that our approaches can achieve superior performance. The results on the Maze2D data indicate that MIS addresses heterogeneous data better than single importance sampling. We also found that MIS can stabilize the reward curve effectively.

Keywords: Q-Learning, behavior cloning, pessimism mechanism, multiple importance sampling.

1. Introduction

Offline RL requires the agent to learn from historical data without interaction with the environment when trial-and-error actions are expensive or dangerous, e.g. autonomous driving and medical experiments. A long-standing problem of offline RL is the distribution shift (Kumar et al., 2020). This means that the action space of the target policy deviates from the offline data. Such actions are called out-of-distribution (OOD) actions. Without online interactions with the environment to correct the agent policy, the bootstrapping (extrapolation) error of the predicted Q-value caused by OOD actions will accumulate over time (Kumar et al., 2019). When the horizon is long, this becomes a considerable problem and the performance of the agent may be affected severely. To tackle this problem, existing work focuses on either behavior cloning (BC) or maximizing Q-Learning (Mnih et al., 2015). BC methods try to mitigate the distribution shift by constraining the target policy to be close to the offline data, thus reducing the overestimation of Q-values and the accumulation of the bootstrapping errors. A common practice is to add a Kullback–Leibler divergence to the policy loss to shrink the distributional discrepancy between the target policy and the behavior policy. Although

BC methods can reduce the extrapolation error incurred by OOD actions, they make the learned policy highly conservative, i.e. the learned policy tends to be a copy of the behavior policy. Besides, the limited exploration of the action space exacerbates the deployment of the learned policy in the online environment.

Maximizing Q-Learning methods prompt the agent to explore the action space by maximizing the Q function without policy regularization, assuming the Q function can accurately predict the Q-value of the state-action pair. However, without policy regularization, the generated actions based on the current states may be arbitrary, resulting in highly uncertain Q-values. Since the Q function is trained on the offline data, it supports actions near the data well but wildly extrapolates on actions deviating far from the data. Overestimated Q functions will, in turn, misguide the policy for generating the next action. To solve this problem, Wu et al. (2021); An et al. (2021); Bai et al. (2022) adopt maximizing Q-Learning methods without explicitly constraining the target policy to be close to the behavior policy. They use a pessimistic mechanism to penalize the OOD actions, with degree of penalization proportional to the uncertainty of the Q-values. As a consequence, the penalization will favour in-distribution actions and suppress OOD actions, so the target policy's action space can stay within the offline data manifold.

Besides the distribution shift, another problem of offline RL is the heterogeneity of the data. Existing algorithms assume the offline data are homogeneous and use a single distribution to fit the data. But in practice, the data may be collected under various scenarios and are thus a mixture of diverse distributions, such as the Maze2D data in D4RL (Fu et al., 2020). Some data may be a mixture of human demonstration, hand-designed controllers and expert/random policies, resulting in multi-modal data. What's worse, the data trajectories may also be non-Markovian, e.g. if the action decisions are made based on external human knowledge, which makes it difficult to model the trajectories by a Markovian decision process. Since the agent knows nothing about the external knowledge when learning from offline data, it is hard to fit the behavior policy. As a result, the target policy is under- or overestimated when we use importance sampling to weigh the biased behavior policy (Chen et al., 2023). This bias is aggravated when the data is high-dimensional and continuous where offline data only covers a minority of the whole feature space. To handle the heterogeneous trajectories, we propose multiple importance sampling to alleviate the discrepancy between the learned policy and the data. Inspired by Gaussian mixture models, we use multiple functions to fit the target policy for heterogeneous data.

Maximizing Q-Learning methods can prompt exploration since the action-generating process is guided by maximizing the Q function without cloning the behavior of offline data. In contrast, BC methods can be seen as exploitation since they try to imitate the offline data and make full use of them. On the one hand, it is critical for the target policy to stay close to the behavior policy to circumvent the extrapolation error caused by OOD actions. On the other hand, it is also important to make the agent fully exploring the environment, i.e. trying new OOD actions so that the target policy can improve over the behavior policy. But OOD actions will cause overestimation of Q-values, even though the pessimistic mechanism that penalizes highly uncertain Q-values can mitigate this problem to a degree. So we still need to constrain the policy to enforce the generated actions to be in-distribution. Therefore, it is necessary to carefully trade off exploitation and exploration for different experimental environments.

With this motivation in mind, we propose to unify maximizing Q-Learning with BC in an implicit and an explicit way to balance exploration and exploitation. For the implicit approach, termed ImBC, we adopt generative adversarial networks (GANs) as the BC item to unify the action space

of the target policy and offline data, making them indistinguishable with the help of a discriminator. GANs thus implicitly make the action space of the target policy and the behavior policy to stay on the same manifold. We conduct extensive experiments on the locomotion data MuJoCo (Fu et al., 2020) and the results show that ImBC exceeds previous state of the art (SOTA) performance obtained by EDAC (An et al., 2021). For the explicit approach, termed ExBC, we combine maximizing Q-Learning with advantage-weighted BC methods where the weight for each state-action pair is learned via MIS. Experiments on the heterogeneous navigation data Maze2D (Fu et al., 2020) show that MIS stabilizes reward curves and handles heterogeneous data better than single importance sampling.

To summarize, the main contributions of our work are: (1) we proposed ImBC based on GANs; (2) ExBC based on advantage-weighted BC where the weights are learned via MIS; (3) we propose the topK loss for ensemble Q-values to stabilize reward curves.

2. Related Work

Maximizing Q Learning. For Offline RL (Levine et al., 2020), the policy optimization can be roughly divided into two categories: maximizing Q-Learning and behavior cloning. The first method learns a Q function using offline data, then utilizes the Q function to guide the policy to generate optimal actions that maximize the Q-values. A common approach of maximizing Q-Learning is to adopt the pessimistic mechanism that penalizes Q-values of OOD actions. BCQ (Fujimoto et al., 2019) uses a VAE model to generate actions that are close to the data, then trains the policy to choose an action from a neighborhood of generated actions, thus restricting the action space to be close to the data while maximizing the Q-value. CQL (Kumar et al., 2020) learns the lower bound of Q-values by maximizing Q-values of in-distribution actions and minimizing the Q-values of OOD actions. There are some publications that try to enforce generated actions to be within the support of the behavioral policy while maximizing the Q-value (Kumar et al., 2019; Wang et al., 2020). These approaches use KL-divergence, maximum mean discrepancy (MMD) or MSE (Fujimoto and Gu, 2021) to explicitly regularize the policy to reduce the extrapolation error incurred by OOD actions. Another line of research doesn't reduce the OOD actions, but penalizes the Q-values of OOD actions, which is referred to as pessimistic mechanism. Pessimistic methods penalize the Q-value of state-action pairs with high uncertainty. EDAC (An et al., 2021) also uses ensemble Q networks to penalize Q-values with high uncertainty but they found that the performance of the target policy degrades significantly when the ensemble Q networks share similar parameters, so they diversify the gradients of different Q networks and use the minimum of the Q-values to guide the policy optimization.

Behavior Cloning. For BC (also called weighted regression) methods, the target policy tries to learn selectively from offline data to improve over the behavior policy. A common practice is to learn an importance weight such as IQL (Kostrikov et al., 2022), for each state-action pair to filter low quality state-action pairs and to make full use of high quality state-action pairs. Pure BC methods like IQL avoid overestimation of Q-values arising from unseen actions. But a noticeable flaw is that BC depends on expert data. It would be difficult for a pure BC method to improve the target policy for low quality data such as the dataset `halfcheetah-random` of D4RL. IQL (Kostrikov et al., 2022) uses the estimated advantage of actions to optimize the weighted behavior policy without querying OOD actions. The advantage is estimated by the difference of Q-values and state values learned from the expectile regression. The latent-variable generative model VAE

(Kingma and Welling, 2014), (which first encodes the state into a latent variable enforced to be close to a standard Gaussian noise and then decodes it into an action) has been used to replace Gaussian policies (Wu et al., 2022; Chen et al., 2022). In SPOT (Wu et al., 2022), the coefficient of the VAE loss is a fixed value. In LAPO (Chen et al., 2022), the VAE loss is weighted by the advantage of the action. LAPO learns the importance weight similar to IQL, the difference is that LAPO replaces the BC item with a VAE and adds a maximizing Q-value item. It is demonstrated that VAEs can better represent multiple modalities of the data. Transformer architectures have also been used to model distributions of the trajectories (Janner et al., 2021; Chen et al., 2021).

3. Preliminaries

We formulate the RL sequential decision-making problem as a discounted Markov Decision Process (MDP) specified by a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{R}, p, \rho_0, \gamma \rangle$, where $\mathcal{S}$ and $\mathcal{A}$ are the state and action spaces, $p(s'|s,a)$ is the transition dynamics, $\mathcal{R} : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$ is the reward function at a transition, ρ_0 is the initial state distribution and $\gamma \in [0,1)$ is the discount factor. The agent performs an action with a policy $\pi : \mathcal{S} \mapsto \mathcal{A}$, then receives a reward $r(s,a)$ and moves to the next state s'. Performing actions repeatedly, the agent will generate a trajectory $\tau : (s_t, a_t, r_t)_{t \geq 0}$. The goal of RL is to obtain a policy $\pi(a|s)$ that maximizes the discounted cumulative reward $\max_{\pi} J(\pi) = \mathbb{E}_{s_0 \sim \rho_0, a_t \sim \pi(\cdot|s_t), s_{t+1} \sim p(\cdot|s_t,a_t)} \sum_{t=0}^{\infty} \gamma^t r_t(s_t, a_t)$. Since, at time t_0, we don't know future rewards r_t, $t > t_0$, we cannot obtain the cumulative reward $J(\pi)$ directly. Instead, a value function $Q(s,a)$ is used to estimate the discounted cumulative reward. Starting from time t, we want $Q(s_t, a_t) = \sum_{k=t}^{\infty} \gamma^{k-t} r_k$ to hold. If the Q function can predict the cumulative reward accurately, it would satisfy the Bellman equation $Q(s_t, a_t) = r_t + \gamma Q(s_{t+1}, a_{t+1})$. To that end, the objective of the parameterized $Q_\phi(s,a)$ is the temporal difference (TD) loss (Sutton, 1988),

$$\mathcal{L}(\phi) = (r + \gamma Q_{\phi'}(s', a') - Q_\phi(s,a))^2, \tag{1}$$

where s' is the next state, a' is the next action and ϕ' is the target parameter of the Q function. After the Q function has been learned, it is used in the policy optimization to guide the policy π to maximize the Q-value. The objective of the parameterized policy π_θ is

$$\max_{\theta} \mathbb{E}_{s \sim \mathcal{D}, a \sim \pi_\theta(\cdot|s)} Q_\phi(s,a), \tag{2}$$

where $\mathcal{D}$ represents the offline data. The above procedure describes the routine of maximizing Q-Learning methods. These methods involve bootstrapping and function approximation to obtain accurate Q functions. Otherwise, the policy might be misguided by the Q function. In practice, the policy-generated actions are typically OOD, which inevitably causes the overestimation problem of Q functions (Kumar et al., 2019).

4. Methodology

We propose two kinds of behavior cloning, implicit BC and explicit BC, and combine them with maximizing Q-Learning where the Q function is penalized by the uncertainty. Since state-action observations are usually limited in the offline data, it is common for the Q function to encounter extrapolation errors from OOD actions, making the Q-value estimator easily fall into a suboptimal area (Kumar et al., 2019, 2020). We estimate the uncertainty of the Q-value estimator by ensemble

neural networks to reduce the impact of OOD actions. We don't compute the variance explicitly. Instead, we use *topK* (the smallest K-many values) or the minimum of Q-values which has the effect of mean minus variance. The Q-values for state-action pairs with high uncertainty will be smaller than those with low uncertainty. This mechanism prompts a pessimistic policy against OOD actions.

4.1. Implicit BC

Implicit behavioral cloning (ImBC) is implemented by adversarial training (Goodfellow et al., 2014). Concretely, the actions generated by the policy are forced to be close to the actions of the data by the discriminator of the GAN. This ImBC approach favors generating actions supported by the offline data, thus alleviating the extrapolation error problem of the Q function. The Q function is an ensemble of multiple Q networks penalized by the uncertainty of the state-action pair. The corresponding Bellman equation is

$$Q_{\phi_i}(s,a) = r + \gamma(\min_{1\le j\le N} Q_{\phi'_j}(s',a') - \log \pi_\theta(a'|s')), \tag{3}$$

where N is the ensemble size, s' is the state of the next time step, a' is the action generated by the policy based on s', ϕ_j and ϕ'_j are parameters of the j-th Q function and target Q function and $-\log \pi_\theta(a'|s')$ is the entropy regularization item of the soft actor critic policy (Haarnoja et al., 2018a). The final Q-value is the minimum of the N ensemble Q-values, which encourages pessimism towards those state-action pairs with high uncertainty. Suppose $Q_{\phi'_j}$ follows $N(\mu,\sigma)$. By using the minimum of the Q-values, without computing the variances explicitly, we can penalize each state-action pair by its uncertainty (Royston et al., 1982; An et al., 2021):

$$\mathbb{E}[\min_{1\le j\le N} Q_{\phi'_j}(s,a)] \approx \mu(s,a) - \Phi^{-1}\left(\frac{N-\pi/8}{N-\pi/4+1}\right)\sigma(s,a). \tag{4}$$

It can be seen that the expectation of the minimum quantifies the Q-value of (s,a) which is penalized by the uncertainty $\sigma(s,a)$. State-action pairs with high uncertainty like OOD actions would have smaller Q-values than those with low uncertainty.

Policy Optimization. For the policy, a discriminator loss is introduced to enforce the generated actions to stay close to the behavior policy. We adopt Wasserstein GANs (Arjovsky et al., 2017) as the behavior cloning module with the discriminator D loss being

$$\mathcal{L}_D = \mathop{\mathbb{E}}_{s,a\sim\mathcal{D}}[D(s,a)] - \mathop{\mathbb{E}}_{\substack{s\sim\mathcal{D}\\ \hat{a}\sim\pi_\theta(s)}}[D(s,\hat{a})]. \tag{5}$$

In the process of maximizing the ensemble Q-values, we propose the topK loss to stabilize the learning process of the Q-value. As mentioned before, topK means the K smallest values of the N Q-values, with the minimum resulting for $K=1$. The final policy objective is

$$\mathcal{L}(\theta) = \lambda \mathop{\mathbb{E}}_{\substack{s\sim\mathcal{D}\\ \hat{a}\sim\pi_\theta(s)}}\left[\alpha \log \pi_\theta(\hat{a}|s) - \frac{1}{K} topK(\{Q_{\phi_j}(s,\hat{a})\}_{j=1}^N)\right] + (1-\lambda)\mathop{\mathbb{E}}_{\substack{s\sim\mathcal{D}\\ \hat{a}\sim\pi_\theta(s)}}[D(s,\hat{a})], \tag{6}$$

where $\log \pi_\theta(\hat{a}|s) = -\mathcal{H}(\pi_\theta)$ and $\mathcal{H}$ means entropy. $\lambda \in [0,1]$ is used to control the degree of conservatism. Minimizing the loss in (6) means we want to maximize not only the cumulative reward $Q(s,\hat{a})$ but also the entropy $\mathcal{H}(\pi_\theta)$. Maximizing the entropy $\mathcal{H}(\pi_\theta)$ enforces the policy

to be as stochastic as possible, thus improving the exploratory ability of the policy. α is called temperature parameter that determines the relative importance of the entropy term in comparison to the action return, hence α controls the stochasticity. Larger α makes the policy more sensitive to the arbitrariness of actions. It can be optimized proportional to the entropy (Haarnoja et al., 2018b). The objective function for α is

$$\mathcal{L}(\alpha) = \mathbb{E}_{\hat{a}\sim\pi_\theta(s)}\left[-\alpha \log \pi(\hat{a}|s) - \alpha\bar{\mathcal{H}}\right]. \tag{7}$$

The average entropy $\bar{\mathcal{H}}$ is usually assigned to be the negative action dimension, e.g. $\bar{\mathcal{H}}$ would be -6 for the `halfcheetah` task. The dynamically tuned α allows the policy to explore more in regions where actions are uncertain, but to remain more deterministic in states with a clear distinction between good and bad actions.

We use the K smallest Q-values that can stabilize the reward curve. Besides, the performance of topK is also slightly better than that of the minimum value as is shown in Figure 1. The minimum of Q-values is too sparse when the ensemble size N is large. Based on the minimum alone, only the gradients of a single value (the minimum) of the N Q-values is back-propagated, while most Q networks stay unoptimized, thus the reward curve tends to be unstable. The average method also has a drawback since all the Q-values will be optimized including the largest ones, which should be avoided according to the pessimistic mechanism. Therefore, we trade off the minimum and average by only optimizing the top K smallest Q-values.

As for the discriminator term $\mathbb{E}_{\hat{a}\sim\pi_\theta(s)}[D(\hat{a})]$ in (6), it can be seen as an action regularization item that suppresses the deviation of the generated actions $\hat{a}$ from the data actions a, thus alleviating overestimation of the Q-values. The discriminator D tries its best to discriminate the actions a and the policy-generated actions $\hat{a}$. The policy optimizer optimizes the generator parameters to produce actions to deceive the discriminator, i.e. generate actions $\hat{a}$ that can't be distinguished from a by the discriminator. The GAN regularization is much less restrictive than mapping states to actions directly as in ExBC, making the learned policy more exploratory than ExBC.

Q Function Optimization. In the policy loss, the Q function is used to guide the policy optimization. The parameters of the Q function are updated by the observed data (s, a, r). We adopt the pessimistic-mechanism-like equation (4) to penalize $Q(s, a)$ by the uncertainty, which can reduce the extrapolation errors of the Q-values. Besides, for objective (6), we have the overall policy gradient decomposed as $\frac{\partial Q}{\partial \theta} = \frac{\partial Q}{\partial \hat{a}}\frac{\partial \hat{a}}{\partial \theta}$, where $\hat{a} = \pi_\theta(s)$, $\frac{\partial Q}{\partial \hat{a}}$ represents the gradient w.r.t. the Q function parameters ϕ and $\frac{\partial \hat{a}}{\partial \theta}$ represents the policy gradient w.r.t. the policy parameters θ. In order to prevent $\frac{\partial Q}{\partial \hat{a}}$ from dominating the gradient, we add a regularization term $\left|\frac{\partial Q}{\partial \hat{a}}\right|^2$ to the Q-loss, thus the policy gradient $\frac{\partial \hat{a}}{\partial \theta}$ will dominate the overall gradient so as to better optimize the policy parameters θ. Finally, the objective for the Q function is

$$\mathcal{L}(\phi_i) = \mathop{\mathbb{E}}_{\substack{(s,a,s')\sim\mathcal{D}\\ \hat{a}\sim\pi_\theta}}\left[\left(r + \gamma\Big(\min_{1\le j\le N} Q_{\phi'_j}(s', \hat{a}) - \log \pi_\theta(\hat{a}|s')\Big) - Q_{\phi_i}(s, a)\right)^2 + \left|\frac{\partial Q_{\phi_i}}{\partial \hat{a}}\right|^2\right], \tag{8}$$

where ϕ_i is the parameter of the i-th Q function and the total loss is a summation of $\mathcal{L}(\phi_i)$, $1 \le i \le N$ where N is the ensemble size.

4.2. Explicit BC

For explicit unification, we combine maximizing Q-Learning with advantage-weighted regression neural networks where the weights $w(s,a)$ are learned via MIS so that contributions to the objective of actions with high advantage will be up-weighted. The target is to maximize the cumulative reward and simultaneously constrain the target distribution $d^\pi(s,a)$ to be close to the behavior distribution $d^D(s,a)$. The policy objective is

$$\max_{d^\pi} \mathop{\mathbb{E}}_{(s,a)\sim d^\pi}[R(s,a)] - \alpha D_f(d^\pi||d^D) \tag{9}$$

$$s.t. \sum_a d^\pi(s,a) = (1-\gamma)\mu_0 + \gamma \mathcal{T}_* d(s),\ \forall s \in S, \tag{10}$$

where $D_f(d^\pi||d^D)$ is f-divergence between d^π and d^D, f is convex and usually set to be the function $f(x) = x\log(x)$ underlying KL-divergence; here $\mathcal{T}_* d(s) = \sum_{\bar{s},\bar{a}} T(s|\bar{s},\bar{a})d(\bar{s},\bar{a})$ is the transposed Bellman operator, μ_0 is the distribution of the initial state. The constrained equation (10) guarantees that $d^\pi(s,a)$ is the occupancy distribution of the target policy. In (9), the first term maximizes the reward on the target policy $\pi(a|s)$ while the second term constrains the target policy to be close to the behavior policy. α controls the strength of the constraint. We reformulate (9) and (10) using a Lagrangian multiplier $V(s)$:

$$\max_{d^\pi} \min_{V(s)} \mathop{\mathbb{E}}_{s,a\sim d^\pi}\left[R(s,a) - \alpha D_f(d^\pi||d^D) + \sum_s V(s)\Big((1-\gamma)\mu_0 + \gamma\mathcal{T}_* d(s) - \sum_a d^\pi(s,a)\Big)\right]. \tag{11}$$

Note that $\mathcal{T}_*$ is adjoint to $\mathcal{T}$, i.e. $\sum_s V(s)\cdot \mathcal{T}_* d(s) = \sum_{s,a} d(s,a)\mathcal{T}V(s,a)$. So we have

$$\begin{aligned}
&\mathbb{E}_{s,a\sim d^\pi}[R(s,a)] - \alpha D_f(d^\pi||d^D) + \sum_s V(s)((1-\gamma)\mu_0 + \gamma\mathcal{T}_* d(s) - \sum_a d^\pi(s,a)) \\
&= \mathbb{E}_{s,a\sim d^\pi}[R(s,a)] - \alpha E_{s,a\sim d^D}\left[f\left(\frac{d^\pi(s,a)}{d^D(s,a)}\right)\right] \\
&\quad + \mathbb{E}_{s,a\sim d^\pi}[\gamma\mathcal{T}V(s,a) - V(s)] + (1-\gamma)\mathbb{E}_{s\sim\mu_0}[V(s)] \\
&= \mathbb{E}_{s,a\sim d^D}\left[\frac{d^\pi(s,a)}{d^D(s,a)}A(s,a) - \alpha f\left(\frac{d^\pi(s,a)}{d^D(s,a)}\right)\right] + (1-\gamma)E_{s\sim\mu_0}[V(s)],
\end{aligned} \tag{12}$$

where $A(s,a) = R(s,a) + \gamma\mathcal{T}V(s,a) - V(s)$ is the advantage of the (s,a) pair.

Offline data can be complicated and multi-modal. It is thus natural to assume that the data is heterogeneous and it is more appropriate to model the data by weighing multiple distributions (similar to Gaussian mixture models). Our motivation for MIS is that if the original data (generated from the behavior policy) is heterogeneous and follows multiple distributions, then the target policy should also follow multiple distributions. In addition, multiple target policies allow the agent to perform diverse actions under similar states s, which improves the agent's exploratory ability. Assuming there are M distributions $(d^{\pi_1}, d^{\pi_2}, ..., d^{\pi_M})$ for the target policy, each d^{π_k} has its own Q-value and state value function. The objective (12) becomes

$$\mathop{\mathbb{E}}_{(s,a)\sim d^D} \sum_{k=1}^{M}\Big[\beta_k\big(\omega_k A_k(s,a) - \alpha f(\omega_k)\big) + (1-\gamma)\beta_k E_{s\sim\mu_0}V_k(s)\Big], \tag{13}$$

Table 1: Comparison of our methods ImBC and ExBC with baselines for the MuJoCo data.

	Task Name	CQL	ATAC	IQL	SPOT	EDAC	ImBC	ExBC
halfcheetah	medium-expert	62.4	95.5	86.7	86.9 ±4.3	106.3± 1.9	**113.3± 2.4**	96.3± 1.5
	medium-replay	46.2	49.5	44.2	52.2 ±1.2	61.3± 1.9	**64.0± 1.3**	46.7± 0.8
	medium	44.4	54.3	47.4	58.4 ±1.0	65.9± 0.6	**72.0± 0.4**	50.3± 0.6
	random	**35.4**	4.8	-	-	28.4± 1.0	30.4 ± 1.9	6.3± 2.5
hopper	medium-expert	111.0	112.6	91.5	99.3± 7.1	110.7± 0.1	92.7± 1.6	**113.1± 1.3**
	medium-replay	48.6	102.8	94.7	100.2± 1.9	101.0± 0.5	**104.9± 1.8**	103.6± 0.7
	medium	58.0	**102.8**	66.2	86.0± 8.7	101.6± 0.6	92.3± 4.2	65.7± 2.9
	random	10.8	31.8	-	-	25.3± 10.4	**32.1± 3.6**	10.8± 3.4
walker2d	medium-expert	98.7	116.3	109.6	112.0± 0.5	114.7± 0.9	**118.1± 1.2**	112.2± 0.8
	medium-replay	32.6	**94.1**	73.8	91.6± 2.8	87.1± 2.3	92.8± 1.3	86.1± 0.7
	medium	79.2	91.0	78.3	86.4± 2.7	92.5± 0.8	**101.3± 0.7**	96.7± 0.3
	random	7.1	8.0	-	-	16.6± 7.0	**24.2± 3.1**	6.8± 2.2

where $\omega_k = \frac{d^{\pi_k}(s,a)}{d^D(s,a)}$ is the importance weight of (s,a), $A_k(s,a) = R(s,a) + \gamma \mathcal{T} V_k(s,a) - V_k(s)$ is the advantage of action a and $\beta_k = 1/M$. When maximizing (13), deriving from the differentiation of (13) with regard to ω_k, a closed-form solution for ω_k is

$$\omega_k = (f')^{-1}\left(\frac{A_k(s,a)}{\alpha}\right) = f'_*\left(\frac{A_k(s,a)}{\alpha}\right), \tag{14}$$

where f_* is the convex conjugate of f, it is defined as $f_*(y) = \sup_x \{xy - f(x)\}$ (Rockafellar, 1970). If f is the function underlying KL-divergence, then $(f')^{-1}(x) = f'_*(x) = e^{x-1}$. Finally, the overall weight ω is the mean of the weights $w_1, w_2, ..., w_K$. We will use ω to weigh different state-action pairs. Substituting all the ω_k into (13), the objective (13) becomes

$$L(\omega) = \mathop{\mathbb{E}}_{(s,a)\sim d^D} \sum_{k=1}^{M} \Big[\beta_k f_*\left(A_k(s,a)/\alpha\right) + (1-\gamma)\beta_k E_{s\sim\mu_0} V_k(s)\Big]. \tag{15}$$

We will demonstrate the effect of MIS on heterogeneous data in Section 5.2.

Policy Extraction. Weighted by the $\omega(s,a)$ learned from (14), the policy extraction objective is

$$\max_\theta \mathop{\mathbb{E}}_{(s,a)\sim d^\pi} \log \pi_\theta(a|s) = \max_\theta \mathop{\mathbb{E}}_{(s,a)\sim d^D} [\omega(s,a) \log \pi_\theta(a|s)], \tag{16}$$

where $\omega(s,a) = \frac{1}{M}\sum_{k=1}^{M} \omega_k$. For each state-action pair (s,a) a specific weight $\omega(s,a)$ controls the contribution of (s,a) to the policy loss. The weight $\omega(s,a)$ aims to make full use of actions with high advantage and to reduce the impact of actions with low advantage. Considering (14) and (17), for a state-action pair that has a large advantage, it will get a large weight and contribute more to the policy gradient. For a state-action pair that has a small or negative advantage, it will contribute

less to the policy gradient. Finally, the overall policy objective for ExBC is

$$\begin{aligned}\mathcal{L}(\theta) = \lambda \mathop{\mathbb{E}}_{\substack{s\sim\mathcal{D}\\ \hat{a}\sim\pi_\theta}} \left[\alpha \log \pi_\theta(\hat{a}|s) - \frac{1}{K} topK\left(\{Q_{\phi_j}(s,\hat{a})\}_{j=1}^{N}\right)\right] \\ - (1-\lambda) \mathop{\mathbb{E}}_{s,a\sim\mathcal{D}} \left[\omega(s,a) \log \pi_\theta(a|s)\right], \quad \hat{a} = \pi_\theta(s)\end{aligned} \tag{17}$$

Different from ImBC in (6), here we map states to actions explicitly and assign different weights learned from MIS in order to select highly advantageous state-action pairs and suppress less valuable ones.

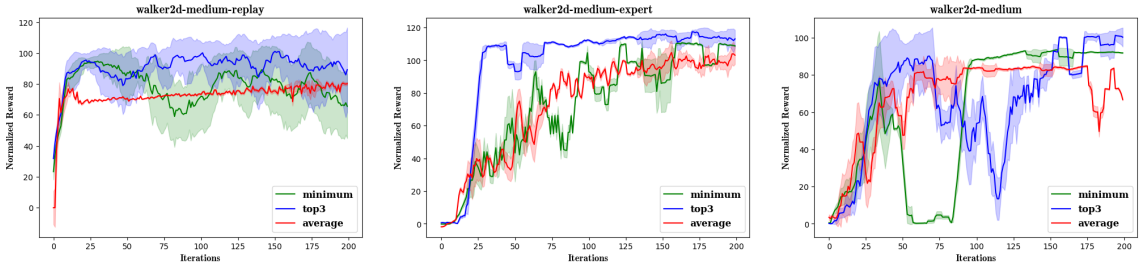

Figure 1: The effect of the topK loss. Minimum, top3 and average loss correspond to K=1,3 and N where N is set to be 10. Shadow area means the variance of the normalized reward.

5. Experiments

We evaluate our proposed approaches against prior approaches on the D4RL benchmark (Fu et al., 2020), with various continuous environments. Following previous publications, we use the normalized average reward as the evaluation metric. λ is chosen as 0.5 and the ensemble size as 10. The results are obtained from ten different seeds with two million training time steps.

Table 2: Comparison of our methods with baselines for Maze2D, '-' means no results reported in that publication.

Task Name	QDT	ROMI-BCQ	SfBC	OptiDICE	ImBC	ExBC
maze2d-large-v1	35.0 ± 24.2	83.1 ± 22.1	74.4 ± 1.7	155.7	**191.6± 32.8**	162.5 ± 11.2
maze2d-medium-v1	10.0 ± 4.7	82.4 ± 15.2	73.8 ± 2.9	145.2	**176.1± 14.3**	152.3 ± 10.4
maze2d-umaze-v1	54.2 ± 9.5	**139.5 ± 3.6**	73.9± 6.6	111.0	84.4 ± 19.2	117.6 ± 8.4
maze2d-large-dense-v1	64.5 ± 6.6	124.0 ± 1.3	-	-	**135.8± 24.3**	128.6 ± 14.2
maze2d-medium-dense-v1	40.5 ± 9.4	102.6± 32.4	-	-	**132.3± 12.4**	104.3 ± 11.2
maze2d-umaze-dense-v1	58.7 ± 1.9	98.3 ± 2.5	-	-	**110.4± 13.2**	102.9 ± 10.4

5.1. Performance on the MuJoCo dataset

The MuJoCo dataset of D4RL contains three environments: `halfcheetah`, `hopper` and `walker2d`. Each environment includes *expert*, *medium-expert*, *medium-replay*, *medium* and *random* domains. In particular, the *expert* dataset contains samples from a fully trained online SAC policy (Haarnoja

et al., 2018a). The *medium* data contains samples generated from the early-stopping SAC policy with samples collected before the policy converges. The *medium-expert* domain means mixing equal amounts of samples generated from the medium and expert policy. The *medium-replay* domain contains samples from the replay buffer when the policy reaches medium level. The *random* domain contains samples from a random policy. Each dataset contains 1M to 2M samples and each sample includes state, action, reward, next state and terminal signal.

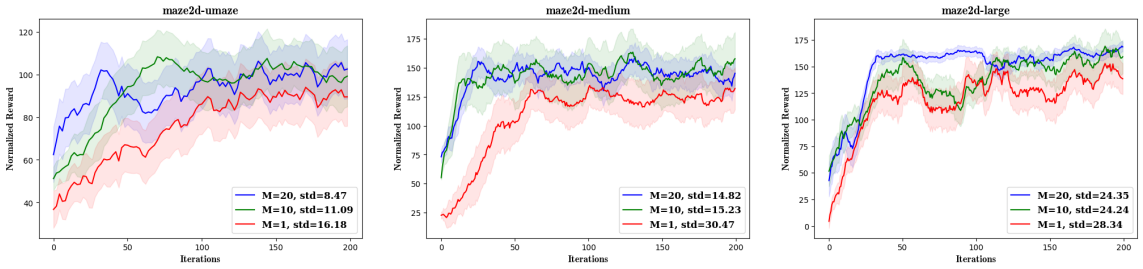

Figure 2: The impact of MIS (M=10,20) in comparison to single importance sampling (M=1) on the heterogeneous data Maze2D.

Baselines. We compare our approach with the following baselines: EDAC (An et al., 2021), which achieved the SOTA performance on the D4RL locomotion dataset and belongs to the maximizing Q-Learning methods. It adopts an ensemble-diversified actor-critic algorithm to minimize the pairwise alignment (cosine distance) of the gradients with regard to actions, thus reducing the ensemble size of the Q functions. The BC method IQL (Kostrikov et al., 2022) uses a state value function and a Q-value function to learn the advantage for each state-action pair. It avoids querying OOD actions and extracts the target policy via advantage-weighted behavior cloning. SPOT (Wu et al., 2022) is a method that combines maximizing Q-Learning and VAEs to model the behavior policy by latent variables. ATAC (Cheng et al., 2022) treats the target policy and behavior policy as two players of a Stackelberg game (Rajeswaran et al., 2020) to trade off between imitation learning and offline RL.

The results of our approaches are shown in Table 1. We can see that **ImBC** outperforms existing methods in most D4RL environments. Our ImBC surpasses the SOTA method EDAC consistently in the `halfcheetah` and `walker2d` environments. The main difference to EDAC is that ImBC has a BC item that consists of a GAN to constrain the generated actions implicitly. This GAN item appropriately mitigates the extrapolation error of the Q function caused by unseen actions. Although the GAN imposes restrictions on the generated actions, it still allows the agent to explore the environment to some degree since the GAN only enforces the generated actions to be indistinguishable from actions in the data instead of enforcing them to be pointwise similar. Further, our method outperforms the pure advantage-weighted BC method IQL in all domains of MuJoCo by a large margin. Compared with our method, IQL focuses on cloning the behavior of the offline data without maximizing the Q-value in the policy optimization, making it difficult to improve substantially over the behavior policy. In Figure 3, our results indicate that it is crucial to have the maximizing Q-value item. Without this item the agent only imitates the data behavior and has no exploratory ability. When the data is generated by a random policy, it would be difficult to improve the target policy by imitating such a random behavior policy. Although alleviating the extrapolation error from OOD actions is important, the agent still needs to explore the unknown environment in order

to improve the target policy over the behavior policy, and maximizing Q-Learning methods help us to do just that.

topK Loss. Figure 1 shows the results of our experiments with the topK loss. We compare the topK loss with the minimum loss for `walker2d` with ensemble size $N = 10$. From Figure 1, the topK loss performs better than the minimum loss. The interpretation is that when the ensemble size N is large, the minimum loss makes the gradients sparse as most Q networks cannot be optimized, while the topK loss can alleviate this adverse effect. We also tried the average loss, i.e. using the average of $\{Q_{\phi_j}\}_{j=1}^{N}$ as the loss. In Figure 1, we can see that the corresponding performance is even worse than for the minimum loss despite having smallest variance. The average loss optimizes not only the lower bound but also the upper bound of the N Q-values, which contradicts the pessimism paradigm. Therefore, it is necessary to trade off the average and the minimum loss, which is what our topK loss does.

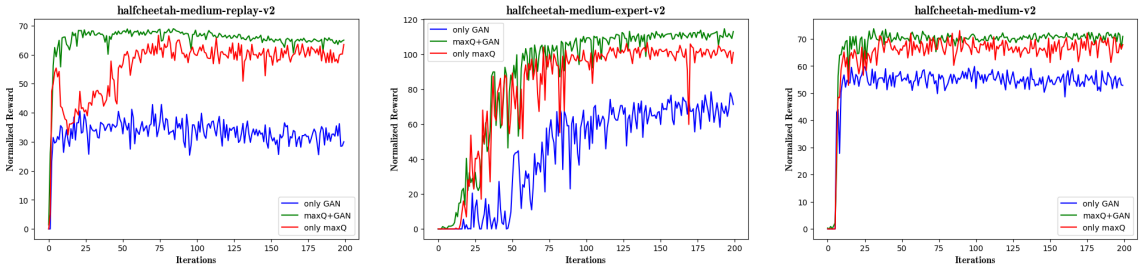

Figure 3: The impact of the BC item and the maxQ item on the policy loss.

5.2. Performance on Maze2D

We evaluate our methods on the navigation task Maze2D (Fu et al., 2020) which is different from the MuJoCo task. Maze2D requires moving a force-actuated ball to a target goal along a certain path from a random starting point. This task mainly tests the agent to stitch together different trajectories from offline data to find the best path to the goal. Since the trajectories were generated by policies of different optimization levels, this makes the data heterogeneous. There are three layouts (umaze, medium, large) in this dataset, with increasing complexity. The agent needs to make full use of expert trajectories while leveraging suboptimal trajectories selectively. Our results are shown in Table 2. We compared our method with SfBC (Chen et al., 2023), ROMI-BCQ (Wang et al., 2021), QDT (Yamagata et al., 2022) and OptiDICE (Lee et al., 2021), which represent the latest algorithms for the Maze2D data. The Maze2D data is mainly used to evaluate the ExBC with MIS which is designed specifically for heterogeneous data. It is worth noting that our ExBC approach can be seen as the MIS-version of OptiDICE; OptiDICE learns the weight for each state-action pair by single importance sampling while ExBC learns the weight by MIS. From Table 2, we can see that ExBC outperforms OptiDICE in all three layouts, which demonstrates the effectiveness of MIS. MIS overcomes the heterogeneity of the data and composes different trajectories to learn the advantage weight better. Although ExBC outperforms the baselines, we can also see its inferiority to the ImBC approach. The potential interpretation is that ExBC limits the policy by mapping states to actions directly, causing the learned policy to be highly conservative, i.e. strictly imitating the behavior of the offline data. However, the constraint of ImBC in terms of GANs is more flexible and allows the agent to explore the action space better than ExBC. This is because based on the same state s, the

GAN only guarantees that $\hat{a}$ and a are distributionally indistinguishable for the discriminator, which is less restrictive than the direct mapping in ExBC that forces $\hat{a}$ to be close to a. The performance gap of ImBC/ExBC indicates the importance of adjusting the strength of BC, i.e. the strength of exploitation vs exploration.

Multiple Importance Sampling. We compare MIS with single importance sampling for the heterogeneous dataset Maze2D. We set the number of distributions M to 1, 10 and 20, respectively. From Figure 2, the overall variance of the normalized reward of MIS is much smaller than that of single importance sampling, which demonstrates that MIS can stabilize the reward curve. Simultaneously, we can see that MIS outperforms single importance sampling slightly in the three layouts of Maze2D, which verifies our conjecture that MIS adapts to heterogeneous data better than single importance sampling. It is worth noting that although the number of parameters of MIS is N times as large as that of single importance sampling, their training time is roughly the same, because we combine multiple neural networks (NNs) into a single big neural network. In ensemble NNs, there is no for-loop to compute each Q-value. For a plain NN, the shape of the weight matrix is (input_dim, output_dim). For ensemble NNs, the shape is (ensemble_size, input_dim, output_dim), which can be seen as a larger NN that has a three-dimensional input. The 3D matrix multiplication can be done with operators like `torch.einsum`. For both $M = 1$ and $M = 10$, the training takes 9 to 10 hours on a Tesla V100-SXM2 GPU card for the `halfcheetah-medium-expert` environment based on 2M training steps.

5.3. Ablation Experiments

We investigate how the BC item and maximizing Q-Learning (maxQ) item of the policy loss affect the performance. Figure 3 shows the results of our approach with and without the BC item in (6). When $\lambda = 1$, the algorithm degrades to the pure maximizing Q-Learning method. When $\lambda = 0$, the algorithm degrades to the pure BC method. From Figure 3 we see that maxQ+GAN outperforms maxQ. This shows that the BC item is crucial and verifies our conjecture that the BC item helps to mitigate the extrapolation error. As is shown by the blue curve of Figure 3, when there is only a BC item, the normalized reward curve decreases severely. It indicates that maximizing Q-Learning is a key item in the policy objective. Because pure BC impedes the agent to explore new actions, resulting in the learned policy to have similar performance to the behavior policy. After combining the maxQ item and BC item implemented by a GAN ($\lambda = 0.5$), the policy achieves remarkable performance gains. Note that maxQ and BC represent exploration and exploitation of the agent, respectively. The ablation results demonstrate the importance of unifying maxQ and BC to tackle the exploration and exploitation dilemma.

6. Conclusion

We proposed implicit and explicit constraints by combining the maximizing Q-Learning approach with GANs or advantage-weighted regression as a trade-off between exploitation and exploration. For the implicit constraint, we treat the policy as the generator of the GAN and use a discriminator to distinguish the actions of the data and the policy-generated actions. After the adversarial training reaches an equilibrium, the policy can generate actions indistinguishable from the actions of the data. We then unified the action space of the target policy and the offline data in an implicit way. The policy objective is the sum of the GAN loss and the maximizing Q-Learning loss where the Q-value is penalized by the uncertainty of the state-action pair. For the explicit constraint, we

proposed to learn the advantage weight of each state-action pair by MIS. Then the weight is used to suppress or make full use of the action. Compared with previously suggested methods, the explicit unification approach achieves excellent performance and stabilizes the reward curve on the heterogeneous navigation data Maze2D, which is specifically designed to test the ability of stitching together multi-modal trajectories. Besides the implicit and explicit unifications, we also proposed the topK loss for policy optimization. Compared with the minimum and the average of the ensemble Q-values, we find that the topK loss can stabilize the reward curve and improve the performance. We have carried out extensive experiments on the D4RL data, the results show that ImBC exceeds previous SOTA approach and ExBC stabilizes the reward curves effectively.

References

G. An, S. Moon, J. H. Kim, and H. O. Song. Uncertainty-based offline reinforcement learning with diversified q-ensemble. In *Proceedings of the 34th Advances in neural information processing systems*, pages 7436–7447, 2021.

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR, 2017.

Chenjia Bai, Lingxiao Wang, Zhuoran Yang, Zhi-Hong Deng, Animesh Garg, Peng Liu, and Zhaoran Wang. Pessimistic bootstrapping for uncertainty-driven offline reinforcement learning. In *International Conference on Learning Representations*, 2022.

Huayu Chen, Cheng Lu, Chengyang Ying, Hang Su, and Jun Zhu. Offline reinforcement learning via high-fidelity generative behavior modeling. In *International Conference on Learning Representations*, 2023.

Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Misha Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling. In *Advances in neural information processing systems*, volume 34, pages 15084–15097, 2021.

Xi Chen, Ali Ghadirzadeh, Tianhe Yu, Jianhao Wang, Yuan Gao, Wenzhe Li, Liang Bin, Chelsea Finn, and Chongjie Zhang. LAPO: Latent-variable advantage-weighted policy optimization for offline reinforcement learning. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, volume 35, 2022.

C. A. Cheng, T. Xie, N. Jiang, and A. Agarwal. Adversarially trained actor critic for offline reinforcement learning. In *International conference on machine learning*, pages 3852–3878, 2022.

Justin Fu, A. Kumar, O. Nachum, G. Tucker, and S. Levine. D4rl: Datasets for deep data-driven reinforcement learning. In *arXiv preprint arXiv:2004.07219*, 2020.

Scott Fujimoto and Shixiang Shane Gu. A minimalist approach to offline reinforcement learning. In *Advances in neural information processing systems*, volume 34, pages 20132–20145, 2021.

Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In *International conference on machine learning*, pages 2052–2062. PMLR, 2019.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, volume 27, 2014.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870, 2018a.

Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, et al. Soft actor-critic algorithms and applications. In *arXiv preprint arXiv:1812.05905*, 2018b.

Michael Janner, Qiyang Li, and Sergey Levine. Offline reinforcement learning as one big sequence modeling problem. In *Advances in neural information processing systems*, volume 34, pages 1273–1286, 2021.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations*, 2014.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. In *International Conference on Learning Representations*, 2022.

Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey Levine. Stabilizing off-policy q-learning via bootstrapping error reduction. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 33, pages 1179–1191, 2020.

Jongmin Lee, Wonseok Jeon, Byungjun Lee, Joelle Pineau, and Kee-Eung Kim. Optidice: Offline policy optimization via stationary distribution correction estimation. In *International Conference on Machine Learning*, pages 6120–6130. PMLR, 2021.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. In *arXiv preprint arXiv:2005.01643*, 2020.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

Aravind Rajeswaran, Igor Mordatch, and Vikash Kumar. A game theoretic framework for model based reinforcement learning. In *International conference on machine learning*, pages 7953–7963. PMLR, 2020.

RT Rockafellar. Convex analysis. princeton university press, princeton, 1970.

JP Royston et al. Expected normal order statistics (exact and approximate). *Journal of the Royal Statistical Society Series C (Applied Statistics)*, 31(2):161–165, 1982.

Richard S Sutton. Learning to predict by the methods of temporal differences. *Machine learning*, 3:9–44, 1988.

Jianhao Wang, Wenzhe Li, Haozhe Jiang, Guangxiang Zhu, Siyuan Li, and Chongjie Zhang. Offline reinforcement learning with reverse model-based imagination. In *Advances in Neural Information Processing Systems*, volume 34, pages 29420–29432, 2021.

Ziyu Wang, Alexander Novikov, Konrad Zolna, and Merel. Critic regularized regression. In *Advances in Neural Information Processing Systems*, volume 33, pages 7768–7778, 2020.

Jialong Wu, Haixu Wu, Zihan Qiu, Jianmin Wang, and Mingsheng Long. Supported policy optimization for offline reinforcement learning. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, volume 35, 2022.

Yue Wu, Shuangfei Zhai, Nitish Srivastava, Joshua M Susskind, Jian Zhang, Ruslan Salakhutdinov, and Hanlin Goh. Uncertainty weighted actor-critic for offline reinforcement learning. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139, pages 11319–11328, 2021.

Taku Yamagata, Ahmed Khalil, and Raul Santos-Rodriguez. Q-learning decision transformer: Leveraging dynamic programming for conditional sequence modelling in offline rl. In *arXiv preprint arXiv:2209.03993*, 2022.