DCRL: Dataset-Constrained Reinforcement Learning for Safe In-Distribution Exploration

Ziang Zheng[*]
School of Vehicle and Mobility,
Tsinghua University
`ziang_zheng@foxmail.com`

Abstract

In offline reinforcement learning (RL), addressing out-of-distribution (OOD) actions is essential for safe policy learning, as such actions often lead to overestimated values and risky behaviors. Existing methods primarily tackle this issue through regularization or counterfactual reasoning but often lack a principled approach to guarantee safe exploration within dataset constraints. This paper presents a novel approach that incorporates safe RL theory into offline RL by introducing the Dataset Feasibility Function (DFF), enabling policy learning that respects dataset boundaries while managing OOD risks. Our proposed Dataset-Constrained Reinforcement Learning (DCRL) framework employs two mechanisms: Dataset Feasibility Guidance (DFG), which serves as a regularization term to keep the policy aligned with the dataset distribution, and Dataset Feasibility Indication (DFI), which acts as an OOD detection tool. DFI enables safe out-of-distribution exploration by leveraging model rollouts constrained within feasible zones identified by a larger tolerance threshold. This approach uniquely blends safety constraints with both regularization and counterfactual reasoning to advance performance and robustness in offline RL. Empirical evaluations on benchmark datasets validate that DCRL outperforms existing methods, achieving superior safety and efficacy in constrained offline tasks.

1 Introduction

Reinforcement Learning (RL) has emerged as a pivotal technology in artificial intelligence[1], driving advancements in fields such as industrial automation[2] [3] and autonomous driving [4]. However, in many real-world applications, safety concerns limit agents' ability to explore their environments freely, necessitating the use of offline RL. In this paradigm, agents must derive policies from a fixed dataset of experiences without real-time interactions, which presents unique challenges.

A significant issue in offline RL is the handling of out-of-distribution (OOD) actions. These arise when an agent selects actions that are underrepresented or absent from the training dataset, often leading to value overestimation, suboptimal policies, and poor generalization to new situations. Traditional offline RL methods, including behavior cloning, policy regularization, and pessimistic value-based approaches, aim to mitigate these OOD risks. However, they grapple with a persistent trade-off between conservatism and flexibility.[5] Overly conservative methods, such as Conservative Q-Learning (CQL)[6] and Behavior Regularized Actor-Critic (BEAR)[7], can hinder agents from discovering optimal actions within the dataset. In contrast, more flexible methods, such as Behavioral Cloning with Q-function (BCQ)[8] and Implicit Q-Learning (IQL)[9], still risk selecting suboptimal OOD actions. Consequently, these approaches often either over-constrain the learned policy, limiting its potential, or allow risky explorations that could lead to unsafe actions.

[*]Corresponding author. Stu.ID: 2024210305

Preprint. Under review.

Rather than confining policy learning to the available data, a novel strategy in offline RL is to enhance exploration through accurate policy evaluation beyond the data support, enabling OOD adaptation. Recent strategies also suggest a more controlled approach: selectively "budgeting" counterfactual decisions to strategically determine where OOD actions may enhance policy outcomes without increasing error risk. [10] Additionally, in cross-domain offline RL, a novel representation-based approach mitigates dynamics mismatches by recovering mutual information between transitions across domains without full domain alignment.[11] This selective cross-domain data sharing significantly enhances learning performance in the target domain with limited data requirements, making it a promising method for safe and adaptive OOD exploration.

Safe reinforcement learning (RL) focuses on optimizing policies while adhering to predefined constraints to ensure agent safety, especially in risk-sensitive environments. Reachability Constrained Reinforcement Learning (RCRL)[12] and Feasible Reachable Policy Iteration (FRPI)[13] are significant developments in this area, where the policy iteratively considers both reward maximization and reachability constraints, ensuring feasible state trajectories that meet safety criteria. Research, such as [14] has outlined methods to incorporate feasibility checks within RL algorithms, highlighting how these approaches can be applied to manage constraints effectively.

Obviously, the core idea of OOD techniques lies in constraining or detecting the policy within the dataset distribution. But what if we could enhance this framework by incorporating safe reinforcement learning strategies to prohibit constraint violations or improve feasibility identification? In this work, we introduce Dataset-Constrained Reinforcement Learning (DCRL), the first approach to integrate safe RL principles directly into the offline reinforcement learning (RL) OOD problem. Our approach centers on the Dataset Feasibility Function (DFF), a novel mechanism that assesses the feasibility of state-action pairs within dataset constraints. By embedding DFF as a regularization term in the policy objective, DCRL provides a structured balance between conservatism and flexibility, guiding the policy to stay within the dataset distribution while selectively permitting exploration beyond it. This dual mechanism, encompassing Dataset Feasibility Guidance (DFG) and Dataset Feasibility Indication (DFI), enables safe, data-compliant exploration while addressing the unique challenges of offline RL. Extensive experiments demonstrate that DCRL not only outperforms existing offline RL approaches in both safety and performance but also represents a significant step forward in leveraging dataset constraints to achieve robust policy learning in constrained environments.

Our contributions are as follows:

- **Introduction of Dataset Constraints in Offline RL:** We are the first to apply safe RL theory to the offline RL setting by introducing a dataset constraint framework, providing a structured approach to mitigate OOD risks and promote safe policy learning.

- **Dataset Feasibility Function (DFF) for Feasibility Evaluation:** We propose the DFF as a mechanism to assess the feasibility of state-action pairs within dataset limits, establishing a principled method to guide safe exploration while maximizing performance within the dataset boundaries.

- **Dual Mechanisms for Enhanced OOD Management:** We develop two complementary strategies—Dataset Feasibility Guidance (DFG) and Dataset Feasibility Indication (DFI). DFG serves as a regularization term that aligns the policy with the dataset distribution, while DFI functions as an OOD detection mechanism, enabling controlled policy optimization through counterfactual reasoning.

- **Empirical Validation on Benchmarks and Real-World Applications:** Extensive experiments on D4RL benchmarks and a legged robot platform demonstrate that our approach outperforms state-of-the-art methods, providing superior safety, adaptability, and robustness in challenging offline RL environments.

2 Proposal Details

Proposal Details please refer to the Appendix.

References

[1] Teng Liu, Xiaosong Hu, Shengbo Eben Li, and Dongpu Cao. Reinforcement learning optimized look-ahead energy management of a parallel hybrid electric vehicle. *IEEE/ASME transactions on mechatronics*, 22(4):1497–1507, 2017.

[2] Deyu Zhang, Wang Sun, Zi-Ang Zheng, Wenxin Chen, and Shiwen He. Adaptive device sampling and deadline determination for cloud-based heterogeneous federated learning. *Journal of Cloud Computing*, 12(1):153, 2023.

[3] Zhongyu Li, Xuxin Cheng, Xue Bin Peng, Pieter Abbeel, Sergey Levine, Glen Berseth, and Koushil Sreenath. Reinforcement learning for robust parameterized locomotion control of bipedal robots. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2811–2817. IEEE, 2021.

[4] Guofa Li, Yifan Yang, Shen Li, Xingda Qu, Nengchao Lyu, and Shengbo Eben Li. Decision making of autonomous vehicles in lane change scenarios: Deep reinforcement learning approaches with risk awareness. *Transportation research part C: emerging technologies*, 134: 103452, 2022.

[5] Ying Jin, Zhuoran Yang, and Zhaoran Wang. Is pessimism provably efficient for offline rl? In *International Conference on Machine Learning*, pages 5084–5096. PMLR, 2021.

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

[7] Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey Levine. Stabilizing off-policy q-learning via bootstrapping error reduction. *Advances in neural information processing systems*, 32, 2019.

[8] 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.

[9] Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. *arXiv preprint arXiv:2110.06169*, 2021.

[10] Yao Liu, Pratik Chaudhari, and Rasool Fakoor. Budgeting counterfactual for offline rl. *Advances in Neural Information Processing Systems*, 36, 2024.

[11] Xiaoyu Wen, Chenjia Bai, Kang Xu, Xudong Yu, Yang Zhang, Xuelong Li, and Zhen Wang. Contrastive representation for data filtering in cross-domain offline reinforcement learning. *arXiv preprint arXiv:2405.06192*, 2024.

[12] Dongjie Yu, Haitong Ma, Shengbo Li, and Jianyu Chen. Reachability constrained reinforcement learning. In *International conference on machine learning*, pages 25636–25655. PMLR, 2022.

[13] Shentao Qin, Yujie Yang, Yao Mu, Jie Li, Wenjun Zou, Jingliang Duan, and Shengbo Eben Li. Feasible reachable policy iteration. In *Forty-first International Conference on Machine Learning*, 2024.

[14] Yujie Yang, Zhilong Zheng, Shengbo Eben Li, Masayoshi Tomizuka, and Changliu Liu. The feasibility of constrained reinforcement learning algorithms: A tutorial study. *arXiv preprint arXiv:2404.10064*, 2024.

[15] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. *Advances in neural information processing systems*, 28, 2015.

[16] Scott Fujimoto and Shixiang Shane Gu. A minimalist approach to offline reinforcement learning. *Advances in neural information processing systems*, 34:20132–20145, 2021.

[17] Dean A Pomerleau. Efficient training of artificial neural networks for autonomous navigation. *Neural computation*, 3(1):88–97, 1991.

[18] Jianxiong Li, Xianyuan Zhan, Haoran Xu, Xiangyu Zhu, Jingjing Liu, and Ya-Qin Zhang. Distance-sensitive offline reinforcement learning. *arXiv preprint arXiv:2205.11027*, 3, 2022.

[19] Jialong Wu, Haixu Wu, Zihan Qiu, Jianmin Wang, and Mingsheng Long. Supported policy optimization for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 35:31278–31291, 2022.

[20] Eitan Altman. *Constrained Markov decision processes*. Routledge, 2021.

[21] Yinan Zheng, Jianxiong Li, Dongjie Yu, Yujie Yang, Shengbo Eben Li, Xianyuan Zhan, and Jingjing Liu. Safe offline reinforcement learning with feasibility-guided diffusion model. *arXiv preprint arXiv:2401.10700*, 2024.

[22] Shengbo Eben Li. *Reinforcement Learning for Sequential Decision and Optimal Control*. Springer Verlag, Singapore, 2023.

[23] Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

[24] Satinder Singh, Tommi Jaakkola, Michael L Littman, and Csaba Szepesvári. Convergence results for single-step on-policy reinforcement-learning algorithms. *Machine learning*, 38: 287–308, 2000.

[25] Yifan Wu, George Tucker, and Ofir Nachum. Behavior regularized offline reinforcement learning. *arXiv preprint arXiv:1911.11361*, 2019.

[26] Guojian Zhan, Ziang Zheng, and Shengbo Eben Li. Canonical form of datatic description in control systems. *arXiv preprint arXiv:2403.01768*, 2024.

A Related Works

Offline Reinforcement Learning (RL) focuses on learning from fixed datasets without real-time interaction, presenting challenges such as distribution shift, which can lead to out-of-distribution (OOD) actions that may result in erroneous value estimations. Existing methods can be broadly categorized into behavior cloning, pessimistic value-based, and regularized policy-based approaches.

Behavior Cloning (BC) is one of the earliest offline RL methods, directly regressing actions from the dataset. While effective, BC's limitation lies in its tendency to mimic suboptimal actions, making it sensitive to dataset quality. Extensions like BCQ (Batch-Constrained Q-learning)[8] use a conditional variational autoencoder[15] to improve flexibility but can still be overly conservative. TD3+BC[16] combines behavior cloning[17] with actor-critic methods, using a BC regularization term, yet it retains the rigidity of traditional BC methods.

Pessimistic approaches such as Conservative Q-Learning (CQL)[6] train the Q-function to underestimate the value of OOD actions, discouraging their selection but potentially stifling exploration of optimal actions. Implicit Q-Learning (IQL)[9] employs expectile regression to avoid evaluating OOD actions directly, reducing complications but limiting the agent's exploration of high-value actions.

Regularized policy-based methods like BEAR (Batch Ensemble Actor-Critic with Regularization) [7] apply Maximum Mean Discrepancy (MMD) to keep the learned policy close to the behavior policy, allowing some exploration while remaining conservative. Similarly, DOGE (Dataset-conditioned Offline RL via Geometric Exploration)[18] and SPOT (Support-Constrained Offline Policy Optimization)[19] use distance functions and density-based regularization to constrain learning within the dataset's convex hull. While these approaches are effective, they often incur high computational costs due to their reliance on density estimation.

B Preliminary

B.1 Safe RL

Safe RL is typically formulated as a Constrained Markov Decision Process [20], which is specified by a tuple $\mathcal{M} := (\mathcal{S}, \mathcal{A}, T, r, h, c, \mathcal{A}mma)$. $\mathcal{S}$ and $\mathcal{A}$ represent the state and action space; $T : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ is transition dynamics; $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function; $h : \mathcal{S} \to \mathbb{R}$ is the constraint violation function; $c : \mathcal{S} \to [0, C_{\max}]$ is cost function; and $\mathcal{A}mma \in (0, 1)$ is the discount factor. Typically, $c(s) = \max(h(s), 0)$, which means that it takes on the value of $h(s)$ when the state constraint is violated ($h(s) > 0$), and zero otherwise ($h(s) \leq 0$). [21]

B.2 Offline RL

In reinforcement learning (RL), the interaction between an agent and its environment is typically modeled as a Markov Decision Process (MDP), denoted as $M = \{S, \mathcal{A}, P, R, \gamma, d_0\}$. Here, S represents the state space, and $\mathcal{A}$ is the action space. The environment dynamics are governed by a transition probability function $P(s' \mid s, a)$, which defines the probability of transitioning from state s to state s' when action a is taken. The reward function $R : S \times \mathcal{A} \to \mathbb{R}$ maps each state-action pair to a scalar reward, and $\gamma \in [0, 1)$ is the discount factor, controlling the importance of future rewards. Finally, d_0 represents the initial state distribution [22].

The goal in RL is to learn a policy $\pi_\theta(a \mid s)$, parameterized by θ, that maximizes the cumulative discounted reward $\mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)\right]$, where $r(s_t, a_t)$ is the reward obtained at time step t. The action-value function (or Q-value) of a policy π is defined as:

$$Q^\pi(s_t, a_t) = \mathbb{E}_{a_{t+1}, a_{t+2}, \cdots \sim \pi}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)\right]. \tag{1}$$

In the offline RL setting, rather than interacting with the environment, the agent is provided with a static dataset $\mathcal{D} \triangleq \{(s, a, r, s')\}$, collected by a behavior policy π_b [23]. Offline RL algorithms aim to learn a policy entirely from this dataset $\mathcal{D}$, without requiring additional online interactions.

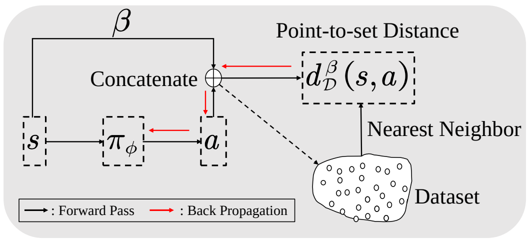

Figure 1: Illustration of the forward calculation and back propaga-tion of our proposed policy regularization with dataset constraint.

B.3 Value Overestimation Issue of Offline RL

We often use the following *one-step* Temporal Difference (TD) update [22] to approximate Q^π,

$$\hat{Q}^\pi(s,a) \leftarrow \hat{Q}^\pi(s,a) + \eta\delta_t, \tag{2}$$

where $\delta_t = \Big[r(s) + \gamma(1-d)\hat{Q}^\pi(s',a') - \hat{Q}^\pi(s,a)\Big]$, $a' = \pi(s')$ and η is a hyper-parameter controlling the step size. With sufficiently enough samples, $\hat{Q}^\pi$ will converge to Q^π [24]. But in offline RL, the dataset $\mathcal{D}$ is limited, with partial coverage of the state-action space. Thus, (s',a') may not exist in $\mathcal{D}$ (a.k.a., distribution shift) because a' is predicted by the learned policy π, not the behavior policy μ. If $\hat{Q}^\pi(s',a')$ is overestimated, the error will continuously backpropagate to the updates of $\hat{Q}^\pi$, eventually causing $\hat{Q}^\pi$ to have overly large outputs for any input state-action. It is known as the value overestimation issue, with which policy regularization [25] has been proven to be effective to deal. And we can categorize existing works on policy regularization into distribution constraint and support constraint (A).

C Dataset Constrained Reinforcement Learning

In this section, we present our approach to Dataset Constrained Reinforcement Learning (DCRL), which focuses on learning optimal policies within the constraints of a fixed dataset collected from various behavior policies. We begin by defining the dataset constraints, followed by an exploration of in-distribution feasibility, the dataset feasibility function, and the framework for policy iteration.

C.1 Dataset Constraint

The basic motivation of *Dataset Constraint (DC)* is to allow the policy π to choose optimal actions from all actions in the offline dataset $\mathcal{D}$. Since either distribution constraint or support constraint regularizes π by only selecting actions from the same state in the dataset, DC empowers a better generalization ability on π.

Definition C.1 (Point-to-set distance)**.** Given the offline dataset $\mathcal{D}$, for any state-action pair $(s,a) \in \mathcal{S} \times \mathcal{A}$, we define its point-to-set distance to $\mathcal{D}$ as

$$d_{\mathcal{D}}^{\beta}(s,a) = \min_{(\hat{s},\hat{a})\in\mathcal{D}} \|(\beta s) \oplus a - (\beta\hat{s}) \oplus \hat{a}\|,$$

where $\oplus$ denotes the vector concatenation operation and β is a hyper-parameter trading off the differences in s and a.

Definition C.2 (Dataset Constraint Signal)**.** We define the constraint signal, denoted as the cost c, based on a distance metric between a given sample and its nearest neighbor in the dataset. This constraint can be formulated in two different ways for practical purposes:

- **Value-based format:** The cost is a continuous value determined by:

$$c = d_{\mathcal{D}}^{\beta}(s,a) - \epsilon \tag{3}$$

  where $d_{\mathcal{D}}^{\beta}(s,a)$ represents the distance between the state-action pair (s,a) and its closest neighbor in the dataset $\mathcal{D}$, with ϵ as a predefined tolerance threshold.

- **Binary format:** The cost is represented as a binary value indicating whether the distance exceeds the tolerance threshold:

$$c = \left(d^{\beta}_{\mathcal{D}}(s,a) - \epsilon\right) > 0 \tag{4}$$

  In this case, the cost is 1 if the distance is greater than ϵ, and 0 otherwise.

C.2 Dataset Feasibility Function

The Dataset Feasibility Function (DFF) is designed to evaluate whether a given state-action pair (s, a) is feasible within the constraints imposed by the dataset, specifically under the context of offline reinforcement learning (RL). Unlike traditional Q-functions that estimate the expected cumulative reward, DFF focuses on estimating the likelihood of a state-action pair being within the dataset's support. By doing so, DFF plays a crucial role in guiding policy learning away from out-of-distribution (OOD) regions, ensuring that the learned policy remains within the feasible regions defined by the dataset.

The DFF can be formally defined in the context of constrained decision-making as follows:

$$F_h(s,a) = \mathbb{E}\left[\max_{t\in\mathbb{N}} h(s_t) \mid s_0 = s, a_0 = a\right] \tag{5}$$

Where $h(s_t)$ indicates whether a state s_t is within the feasible region defined by the dataset constraints (binary or continuous feasibility measure). $F_h(s, a)$ is the dataset feasibility action-value function, which evaluates the feasibility of taking action a in state s based on dataset support.

This formulation allows the DFF to serve as a regularization term that penalizes policies for selecting OOD actions. The lower the DFF value for a given state-action pair, the more likely it is that the action is OOD, and vice versa. This makes the DFF an essential tool for controlling policy exploration, particularly in scenarios where dataset coverage is limited.

C.2.1 Constrained MDPs

To formalize the problem of avoiding OOD actions, we model the offline RL problem as a Constrained Markov Decision Process (CMDP). In a CMDP, the agent must optimize its expected return while satisfying a set of constraints that reflect real-world limitations or, in our case, dataset support constraints.

The CMDP can be defined by the tuple $\langle \mathcal{S}, \mathcal{A}, P, R, C, \gamma\rangle$, where $\mathcal{S}$ is the state space. $\mathcal{A}$ is the action space. $P(s' \mid s, a)$ is the transition probability from state s to state s' after taking action a. $R(s, a)$ is the reward function. $C(s, a)$ is the cost function that encodes the dataset feasibility. Here, $C(s, a) = 0$ if (s, a) is within the dataset support and $C(s, a) > 0$ otherwise. γ is the discount factor.

In this framework, the DFF acts as the constraint function within the CMDP. The agent's objective is to maximize the expected cumulative reward while minimizing the cumulative cost associated with OOD actions:

$$\max_{\pi} \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)\right], \quad \text{s.t.} \quad \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t C(s_t, a_t)\right] \le \epsilon \tag{6}$$

Where ϵ is a threshold for acceptable OOD behavior. This formulation ensures that the policy avoids actions leading to unsafe or unsupported states while still exploring within the feasible dataset region.

Remark C.3 (DFF as an OOD Indicator). The key to proving DFF as an OOD indicator lies in its ability to estimate the feasibility of state-action pairs under the dataset constraints. Since the DFF evaluates whether a state-action pair belongs to the support of the dataset, it naturally acts as an indicator for OOD actions. If the DFF value for a given (s, a) is low, it indicates that the pair lies outside the dataset's feasible region. On the other hand, a high DFF value suggests that the pair is within the dataset's support.

Remark C.4 (Implementing DFF as a Network). We implement DFF as a neural network to estimate the feasibility of state-action pairs. The DFF network takes as input a state s and an action a, and outputs a scalar value representing the feasibility score for that pair. The architecture of the DFF network can be designed similarly to a value network in traditional RL.

Formally, the loss for the DFF network is given by:

$$\mathcal{L}_{\text{DFF}} = \mathbb{E}_{(s,a)\sim\mathcal{D}}\left[(F_h(s,a) - h(s))^2\right] \tag{7}$$

Where $F_h(s,a)$ is the predicted feasibility score from the DFF network, and $h(s)$ is the ground truth feasibility measure.

C.3 Dataset Feasibility Function (DFF) Update

In our method, the Dataset Feasibility Function (DFF) is a key component that helps to evaluate the feasibility of state-action pairs based on dataset constraints. Inspired by the structure of TD3's dual value network, we introduce a similar dual update mechanism for DFF.

We define the optimal feasible state-value function $C_h^*(s)$ and the optimal feasible action-value function $F_h^*(s,a)$ as follows:

Definition C.5 (Optimal Dataset Feasible Value Function)**.** The optimal feasible state-value function $C_h^*(s)$ is defined as:

$$\begin{aligned} C_h^*(s) &:= \min_\pi C_h^\pi(s) \\ &:= \min_\pi \max_{t\in\mathbb{N}} h(s_t), \\ \text{s.t.} \quad & s_0 = s, a_t \sim \pi(\cdot \mid s_t), \end{aligned} \tag{8}$$

where $h(s_t)$ denotes the feasibility measure (binary or continuous) indicating whether the state s_t is feasible under the dataset constraints.

Similarly, the optimal feasible action-value function $F_h^*(s,a)$ is defined as:

$$\begin{aligned} F_h^*(s,a) &:= \min_\pi F_h^\pi(s,a) \\ &:= \min_\pi \max_{t\in\mathbb{N}} h(s_t), \\ \text{s.t.} \quad & s_0 = s, a_0 = a, a_{t+1} \sim \pi(\cdot \mid s_{t+1}), \end{aligned} \tag{9}$$

Theorem C.6 (DFF Update Rule)**.** *The DFF update rule follows a similar structure to TD3's Q-value update, incorporating both feasible state-value and action-value functions. For each state s, the update for the feasible state-value function C_h is given by:*

$$C_h'(s) = (1-\gamma)h(s) + \gamma \max\left(C_h(s'), F_h(s,a)\right), \tag{10}$$

where γ is the discount factor applied in the infinite horizon setting, and s' is the next state sampled according to the dynamics model or dataset.

For action-value updates, we define the objective function in terms of state and action feasibilities:

$$F_h'(s,a) = \mathbb{E}\left[\max\left(h(s), \gamma F_h(s',a')\right)\right], \tag{11}$$

where the next action a' is sampled according to the policy π.

C.4 Dataset Constrained Reinforcement Learning Framework

We propose Dataset-Constrained Reinforcement Learning (DCRL), a novel framework that introduces the Dataset Feasibility Function (DFF), a mechanism for evaluating the feasibility of state-action pairs under dataset constraints. We leverage DFF in two key innovations: (1) Dataset Feasibility Guidance (DFG), where DFF serves as a regularization term in the optimization objective to guide policy learning within the dataset's distribution, and (2) Dataset Feasibility Indication (DFI), which identifies feasible zones and allows state-wise policy optimization with distinct objectives. If a state-action pair is deemed in-distribution, we perform traditional optimization; if out-of-distribution, we employ counterfactual reasoning to enable safe exploration beyond the dataset.

C.4.1 Dataset Feasibility Guidance

To ensure the policy avoids out-of-distribution (OOD) actions, we incorporate DFF as a regularization term directly into the policy optimization objective. This guides the policy toward maximizing feasible actions while minimizing the risk of OOD actions. The objective for the policy update can be written as:

$$\pi' = \arg\max_{\pi} \mathbb{E}_{s,a\sim\pi}\left[Q^{\pi}(s,a) - \lambda F_h(s,a)\right], \tag{12}$$

where λ is a regularization parameter controlling the trade-off between reward maximization and feasibility preservation.

Algorithm 1 DCRL-DFG

```
1: Input: Initial policy parameters $\phi$, Q-function parameters $\theta_1$, $\theta_2$, C-function parameters $\omega_1$, $\omega_2$,
   offline dataset $\mathcal{D}$, hyper-parameters $\alpha, \beta, \tau$
2: Set $\theta_1' \leftarrow \theta_1$, $\theta_2' \leftarrow \theta_2$, $\phi' \leftarrow \phi$
3: for step $t = 1$ to $T$ do
4:     Sample a mini-batch of transitions $\{(s, a, r, s', d)\}$ from $\mathcal{D}$
5:     Update $\theta_i,\ i \in \{1, 2\}$ using gradient descent with
6:     Use KD-Tree to find the nearest neighbor in $\mathcal{D}$ of every $(s, \pi_\phi(s))$
7:     Calculate the necessary values
8:     Update $\phi$ using gradient descent with
9:     Update target networks:
10:        $\theta_1' \leftarrow \tau\theta_1 + (1-\tau)\theta_1'$;    $\theta_2' \leftarrow \tau\theta_2 + (1-\tau)\theta_2'$
11:        $\omega_1' \leftarrow \tau\omega_1 + (1-\tau)\omega_1'$;    $\omega_2' \leftarrow \tau\omega_2 + (1-\tau)\omega_2'$
12:        $\phi' \leftarrow \tau\phi + (1-\tau)\phi'$
13: end for
```

Proposition C.7 (Policy Evaluation). *Consider an initial $Q_0 : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ with $|\mathcal{A}| < \infty$. The Q-value iterates defined by $Q_{k+1} = \mathcal{T}_c^{\pi} Q_k$ will converge to a fixed point Q^{π} as $k \to \infty$.*

Proposition C.8 (Policy Improvement). *Let π_k be the policy at iteration k, and π_{k+1} be the updated policy (maximizing the objective function). Then for all $(s,a) \in \mathcal{S} \times \mathcal{A}$ with $|\mathcal{A}| < \infty$, we have $Q^{\pi_{k+1}}(s,a) \geq Q^{\pi_k}(s,a)$.*

C.4.2 Dataset Feasibility for OOD Identification (Proposal)

To extend the Dataset Feasibility Function (DFF) for Out-of-Distribution (OOD) identification, we propose a dual-threshold approach. The key idea is to maintain two thresholds, ϵ_1 and ϵ_2, corresponding to two different levels of feasibility for state-action pairs. This framework enables us to better manage policy optimization both within and beyond the dataset's feasible region, allowing safe exploration of OOD states through model rollouts.

In this proposal, we define two DFFs—one for in-distribution (ID) actions and one for actions that are slightly out-of-distribution (OOD). These functions are governed by two separate feasibility thresholds:

- ϵ_1: A strict threshold that defines the primary dataset's feasible region. State-action pairs with feasibility scores above this threshold are considered safe and in-distribution.
- ϵ_2: A more relaxed threshold that allows exploration into a secondary zone where state-action pairs are feasible but riskier, and model rollouts can be applied for optimization.

Theorem C.9 (Policy Optimization under Dual DFF Constraints). *Once the rollouts are generated, policy optimization proceeds with the following structure:*

- *In-distribution optimization: When $DFF_1(s,a) \geq \epsilon_1$, the agent optimizes its policy using traditional techniques, such as actor-critic methods, relying on actual dataset transitions.*
- *Out-of-distribution optimization: When $\epsilon_1 > DFF_1(s,a) \geq \epsilon_2$, optimization is based on the model-generated rollouts, and the policy is updated subject to the relaxed dataset feasibility constraint $DFF_2(s,a) \geq \epsilon_2$. [10]*

In our next phase of analysis, we will focus on:

1. Formalizing the DFI Framework: Further refining the mathematical framework for dual DFFs, including determining optimal values for ϵ_1 and ϵ_2 based on the dataset and environment dynamics.
2. Designing the World Model: Developing a robust world model capable of accurately predicting environment dynamics for OOD rollouts, potentially using techniques like normalizing flows or variational autoencoders for dynamic modeling.
3. Integration with Policy Optimization: Implementing a two-stage optimization pipeline where in-distribution and OOD optimizations are handled in parallel, ensuring smooth transitions between the two modes.

C.5 Convergence Analysis

Theorem C.10 (Convergence of DCRL)**.** *The proposed DCRL algorithm converges to a policy* π^* *that optimally balances performance and dataset constraints, achieving* $\max_\pi J(\pi)$ *subject to the feasibility conditions defined by the dataset.*

Definition C.11 (Lipschitz Function)**.** A function f from $S \subset \mathbb{R}^m$ into $\mathbb{R}^n$ is called a Lipschitz function if there exists a constant $K \geq 0$ such that

$$\|f(x) - f(y)\| \leq K\|x - y\|, \tag{13}$$

for all $x, y \in S$. Here, K is known as the Lipschitz constant. We denote the L2 norm as $\|\cdot\|$ unless stated otherwise.

D Experiment Proposal

D.1 Feasibility Zone Validation with Synthetic Dataset

To begin, we will evaluate the effectiveness of our Dataset Feasibility Function (DFF) in identifying feasible zones through a controlled experiment using a synthetic dataset. This dataset will be generated by us, designed to include both in-distribution (ID) and out-of-distribution (OOD) state-action pairs. By visualizing the learned feasible zones, we aim to confirm that our DFF can accurately detect OOD actions and guide the policy toward safe, in-distribution decisions. This step will allow us to validate the feasibility zone boundaries in a straightforward setting and provide an intuitive understanding of the DFF's behavior.

Definition D.1 (Dataset Reachable Zone)**.** The data collected by U enables the reach of various states through trajectories. We denote the Dataset Reachable Zone as:

$$S_{\mathcal{D}R} = \{s \mid \sum_{i=0}^{K} d_{\mu_i}(s) > 0, s \in S\}, \tag{14}$$

where S is the state space, and $S_{\mathcal{D}R}$ is the Dataset Reachable Zone. This zone consists of all states with occupancy by U.

Definition D.2 (Feasible Region)**.** The feasible region for policy π and the largest feasible region are defined as:

$$\mathcal{S}_f^\pi := \{s \mid V_h^\pi(s) \leq 0\}, \tag{15}$$

$$\mathcal{S}_f^* := \{s \mid V_h^*(s) \leq 0\}. \tag{16}$$

D.2 Toy Car Tracking Environment

Next, we will apply our method to a toy car tracking environment to further explore the performance of DFF under different dataset configurations. We will vary the composition of the dataset, testing settings such as:

- Positive Sample Proportion: Varying the percentage of positive samples to assess the impact on policy learning and generalization.

- Goal-Reaching Data: Including or excluding data where the toy car successfully reaches its goal to evaluate how this affects the feasibility zone and policy optimization.

These experiments will allow us to assess how the distribution of samples within the dataset influences the DFF's ability to guide the policy toward optimal actions.[26]

D.3 D4RL Benchmark Evaluation

Following the initial experiments, we will conduct extensive tests using the D4RL (Deep Offline RL) benchmark, which includes a variety of standard tasks for offline RL evaluation. We will compare our method, Dataset-Constrained Reinforcement Learning (DCRL), against state-of-the-art methods such as CQL, BEAR, and BCQ. The datasets will cover diverse environments, including tasks with varying difficulty levels and data quality.

We have already gathered preliminary results, which are summarized in the following table:

Table 1: Average normalized score over the final 10 evaluations and 5 seeds. Scores with the highest mean are highlighted.

Task Name	BC	BCQ	BEAR	AWAC	CQL	IQL	TD3+BC	DOGE	SPOT	PRDC	DCRL(DFG)
halfcheetah-random	0.2	8.8	15.1	—	20.0	11.2	11.0	17.8	—	26.9	**26.9** ± 1.0
hopper-random	4.9	7.1	14.2	—	8.3	7.9	8.5	21.1	—	26.8	**26.8** ± 9.3
walker2d-random	1.7	6.5	**10.7**	—	8.3	5.9	1.6	0.9	—	5.0	5.0 ± 1.2
halfcheetah-medium	42.6	47.0	41.0	43.5	44.0	47.4	48.3	45.3	58.4	63.5	**65.8** ± 0.9
hopper-medium	52.9	56.7	51.9	57.0	58.5	66.2	59.3	98.6	86.0	100.3	**101.3** ± 0.2
walker2d-medium	75.3	72.6	80.9	72.4	72.5	78.3	83.7	86.8	86.4	85.2	**87.1** ± 0.4
halfcheetah-medium-replay	36.6	40.4	29.7	40.5	45.5	44.2	44.6	42.8	52.2	55.0	**55.0** ± 1.1
hopper-medium-replay	18.1	53.3	37.3	37.2	95.0	94.7	60.9	76.2	100.2	100.1	**101.4** ± 1.6
walker2d-medium-replay	26.0	52.1	18.5	27.0	77.2	73.8	81.8	87.3	91.6	92.0	**93.1** ± 1.6
halfcheetah-medium-expert	55.2	89.1	38.9	42.8	91.6	86.7	90.7	78.7	86.9	94.5	**102.9** ± 0.5
hopper-medium-expert	52.5	81.8	17.7	55.8	105.4	91.5	98.0	102.7	99.3	109.2	**112.9** ± 4.0
walker2d-medium-expert	107.5	109.5	95.4	74.5	108.8	109.6	110.1	110.4	**112.0**	111.2	111.2 ± 0.6

These comparisons will highlight the strengths and weaknesses of our approach in relation to other popular offline RL methods.

D.4 Real-World Application on Unitree GO1 Legged Robot

Finally, we will transition our experiments to a real-world application by testing DCRL on a Unitree GO1 legged robot. The dataset for this task will be collected from real-world interactions where the robot navigates different terrains and performs locomotion tasks. We will measure the performance of our algorithm in terms of:

- Stability: How well the robot maintains balance and control across diverse scenarios.
- Policy Effectiveness: The ability of the learned policy to perform tasks safely while staying within the dataset constraints.
- Real-World Generalization: Testing whether our approach can generalize from offline data to real-world situations without encountering unsafe behavior.

This final experiment will demonstrate the practical application and robustness of DCRL in real-world robotics, providing insights into its performance in complex, dynamic environments.