Safe Decision Transformer with Learning-based Constraints

Ruhan Wang RUHWANG@IU.EDU
Indiana University

Dongruo Zhou DZ13@IU.EDU
Indiana University

Editors: N. Ozay, L. Balzano, D. Panagou, A. Abate

Abstract

In the field of safe offline reinforcement learning (RL), the objective is to utilize offline data to train a policy that maximizes long-term rewards while adhering to safety constraints. Recent work, such as the Constrained Decision Transformer (CDT) (Liu et al., 2023b), has utilized the Transformer (Vaswani, 2017) architecture to build a safe RL agent that is capable of dynamically adjusting the balance between safety and task rewards. However, it often lacks the stitching ability to output policies that are better than those existing in the offline dataset, similar to other Transformer-based RL agents like the Decision Transformer (DT) (Chen et al., 2021). We introduce the Constrained Q-learning Decision Transformer (CQDT) to address this issue. At the core of our approach is a novel trajectory relabeling scheme that utilizes learned value functions, with careful consideration of the trade-off between safety and cumulative rewards. Experimental results show that our proposed algorithm outperforms several baselines across a variety of safe offline RL benchmarks.

Keywords: safe reinforcement learning; decision transformer

1. Introduction

Recent studies have demonstrated the Transformer's (Vaswani, 2017) state-of-the-art performance across a range of applications, including natural language processing (Wolf et al., 2020; Beltagy et al., 2020) and computer vision (Liu et al., 2021; Arnab et al., 2021). When applied to the domain of Reinforcement Learning (RL), a recent trend is to treat the decision-making problem as a sequence modeling problem, using autoregressive models such as Transformer which maps the history information directly to the action (Chen et al., 2021) or the next state (Janner et al., 2021). Notably, the Decision Transformer (DT) (Chen et al., 2021) effectively extends the Transformer architecture to offline RL tasks, showcasing strong performance, particularly in sequential modeling. However, it is worth noting that while DT excels in maximizing long-term rewards, it may not always align with the complexities of real-world tasks. In practice, many tasks cannot be simplified solely into optimizing a single scalar reward function, and the presence of various constraints significantly narrows the spectrum of feasible solutions (Garcıa and Fernández, 2015). Such a setting is called safe RL, which has been studied in lots of safety-related decision-making problems. For instance, it is crucial that robots interacting with humans in human-machine environments prioritize human safety and avoid causing harm. In the realm of recommender systems, it is imperative to avoid recommending false or racially discriminatory information to users. Similarly, when self-driving cars operate in real-world environments, ensuring safety is paramount (Shalev-Shwartz et al., 2016; Gu et al., 2022; Pecka and Svoboda, 2014).

Constrained Decision Transformer (CDT) (Liu et al., 2023b) serves as a pioneering work which extends the Transformer-based RL to the safe RL regime, which builds upon the foundation of the DT

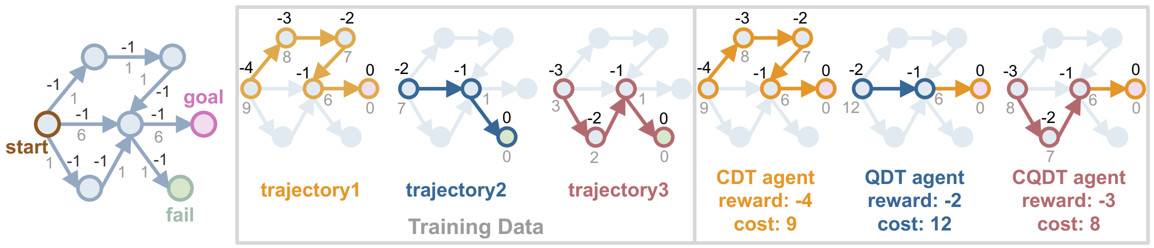

Figure 1: The toy example is presented where arrows denote the node connections. Here we use black number to denote rewards and gray number to denote costs. The demonstration example shows that cost return-based method (CDT) fails to find the shortest path to the goal since it lacks the stitching ability. In contrast, Q-learning-based DT (QDT) finds the shortest path, while it violates the safety constraints. Our proposed CQDT enjoys the superiority of both methods.

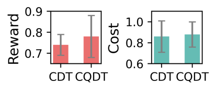

Figure 2: Performance comparison between CDT and CQDT.

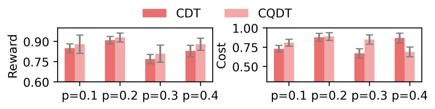

Figure 3: Stitching ability comparison between CDT and CQDT on reward-suboptimal datasets.

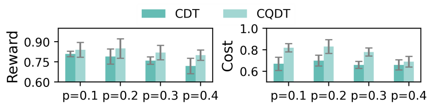

Figure 4: Stitching ability comparison between CDT and CQDT on cost-suboptimal datasets.

while incorporating essential safety constraints. CDT's core objective is to acquire a safe policy from an offline dataset. Its distinguishing feature is the ability to maintain zero-shot adaptability across a range of constraint thresholds, rendering it highly suitable for real-world RL applications burdened by constraints. While CDT demonstrates exceptional competitiveness in safe offline RL tasks, it shares a common limitation with DT—an absence of the 'stitching' capability. This crucial ability involves integrating sub-optimal trajectory segments to form a near-optimal trajectory, a pivotal characteristic highly desired in offline RL agents. What is more challenging is that for RL with safety constraints, the agent not only needs to stitch trajectories to achieve better reward feedback but also needs to guarantee that the obtained policy is still *safe* after stitching, not being affected by unsafe trajectories in the offline dataset. Thus, we raise the following question:

Can we design DT-based algorithms that output safe policies with the stitching ability?

We answer this question affirmatively. To better demonstrate our algorithm design, we propose a toy example involving finding the path that maximizes reward under cost constraints, as illustrated in Figure 1. The task's objective is to identify a path with the highest reward while adhering to a cost constraint (cost limitation = 10). The training data covers segments of the optimal path, but none of the training data trajectories encompasses the entire optimal path. The agent must synthesize these fragmented trajectories to determine the optimal path within the cost constraint. For the existing DT-based RL algorithms such as Q-learning Decision Transformer (QDT) (Yamagata et al., 2023), they are able to stitch suboptimal trajectories into the optimal ones, but they are unable to maintain safety during the stitching phase. For the existing DT-based safe RL methods such as CDT, they can only obtain suboptimal policies while satisfying the safety guarantee, due to the lack of stitching ability. Thus, we propose the **Constrained Q-learning Decision Transformer (CQDT)** method to address the issues in existing methods, as shown in Figure 1. The main contributions are listed as follows.

- CQDT seeks to improve the quality of the training dataset by utilizing cost and reward critic functions from the Constraints Penalized Q-learning (CPQ) framework (Xu et al., 2022) to relabel the return-to-go (RTG) and cost-to-go (CTG) values in the training trajectories. This relabeled dataset is subsequently used to train a Decision Transformer-based safe RL method, such as CDT.

- We provide a comparative analysis of our CQDT against various safe RL benchmarks across different RL task settings. The results are summarized in Figure 2, demonstrate that CQDT consistently outperforms all existing benchmarks in several offline safe RL tasks.

- We also show that our proposed CQDT enjoys better stitching ability compared with CDT, which suggests that CQDT can better utilize suboptimal trajectories in the offline dataset. The results are summarized in Figure 3 and Figure 4.

2. Related Work

Offline Reinforcement Learning. Offline RL refers to a data-driven approach to the classic RL problem (Levine et al., 2020). It focuses on deriving policies from pre-existing data without requiring further interaction with the environment. The practical applications of offline RL are extensive, spanning domains such as healthcare (Gottesman et al., 2018) and the acquisition of robotic manipulation skills (Kalashnikov et al., 2018). However, its inherent challenge lies in the potential disparity between the learned policy and the behavior that generated the initial offline data (Kumar et al., 2019). Addressing this challenge has spurred the development of various methodologies, including constraining the learned policy close from the behavior policy (Fujimoto et al., 2019; Kumar et al., 2019, 2020; Yu et al., 2021; Janner et al., 2019; Kidambi et al., 2020; Wang et al., 2018; Peng et al., 2019; Janner et al., 2021). There is a recent line of work aiming at providing a Transformer-based policy without explicitly constraining the distribution shift issue (Chen et al., 2021). Our work falls into that regime, which uses a Transformer as the policy model focusing on the safe RL setting.
Offline Safe Reinforcement Learning. Offline Safe RL aims to acquire constrained policies from pre-collected datasets, ensuring adherence to safety requirements throughout the learning process. This approach amalgamates techniques from both offline RL and safe RL, leveraging methodologies from both domains (Le et al., 2019). Certain methods tackle the constrained optimization problem through stationary distribution correction, employing Lagrangian constraints to ensure safe learning (Xu et al., 2022; Polosky et al., 2022). Our work does not take the Lagrangian approach to learn a safe policy. Instead, our method explicitly treats the safe policy learning as a sequence modeling problem, similar to the previous CDT approach. Such an approach enjoys the simplicity regarding the algorithm design, as well as the robustness to the algorithm performance.

3. Preliminaries

Offline Safe RL. We formulate the environment as a Constrained Markov Decision Process (CMDP), a mathematical framework for addressing the safe RL problem (Altman, 1998). A CMDP comprises a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, r, c, \mu_0)$, where $\mathcal{S}$ denotes the state space, $\mathcal{A}$ signifies the action space, $\mathcal{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ represents the transition function, $r : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ stands for the reward function, and $\mu_0 : \mathcal{S} \to [0, 1]$ indicates the initial state distribution. In CMDP, $c : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, C_{\max}]$ quantifies the cost incurred for violating constraints, where $C_{\max}$ denotes the maximum cost allowable.

We denote the policy by $\pi : \mathcal{S} \times \mathcal{A} \to [0, 1]$, while $\tau = \{s_1, a_1, r_1, c_1, \ldots, s_T, a_T, r_T, c_T\}$ delineates the trajectory containing state, action, reward, and cost information throughout the maximum episode length T. We use $\tau.s_t, \tau.a_t, \tau.r_t, \tau.c_t$ to denote the t-th state, action, reward and cost in trajectory τ. For each time step t, the action a_t is drawn following the distribution $\pi(s_t, \cdot)$, and the next state s_{t+1} is drawn following the transition function $\mathcal{P}(s_t, a_t, \cdot)$. The cumulative reward and cost for a trajectory τ are represented as $R(\tau) = \sum_{t=1}^{T} r_t$ and $C(\tau) = \sum_{t=1}^{T} c_t$. We also denote $R_t = \sum_{i=t}^{T} r_i$ by the RTG at t-th step, $C_t = \sum_{i=t}^{T} c_i$ by the CTG at t-th step as well.

For simplicity, we define $Q_r^\pi(s, a) = \mathbb{E}_{\tau\sim\pi}[R(\tau)|\tau.s_1 = s, \tau.a_1 = a]$ as the expected return of the policy starting from initial state s and action a. Similarly, we denote $Q_c^\pi(s, a) = \mathbb{E}_{\tau\sim\pi}[C(\tau)|\tau.s_1 = s, \tau.a_1 = a]$ as the expected cost. The agent's objective is to determine a policy π^κ that maximizes the reward while ensuring that the cumulative cost for constraint violation remains below the threshold κ:

$$\pi^\kappa = \arg\max_{\pi} \mathbb{E}_{\tau\sim\pi,\tau.s_1\sim\mu_0}[R(\tau)], \text{s.t. } \mathbb{E}_{\tau\sim\pi,\tau.s_1\sim\mu_0}[C(\tau)] \leq \kappa. \tag{1}$$

For safe offline RL, the agent learns the optimal safe policy purely from a static dataset $\mathcal{T}$ that is previously collected with an unknown behavior policy (or policies). Specifically, $\mathcal{T}$ consists of m episodes τ_i with the maximum length T, which is $\mathcal{T} := \{\tau_1, \ldots, \tau_m\}$.

Constrained Decision Transformer. Our algorithm builds upon the CDT framework. CDT utilizes the return-conditioned sequential modeling framework to accommodate varying constraint thresholds during deployment, ensuring both safety and high return. CDT employs a stochastic Gaussian policy representation to generate the action at t-th time step, i.e., $a_t \sim \pi_\theta(\cdot|o_t) = \mathcal{N}(\mu_\theta(o_t), \Sigma_\theta(o_t))$, where $o_t = \{R_{t-K:t}, C_{t-K:t}, s_{t-K:t}, a_{t-K:t-1}\}$ represents the truncated history from step $t-K$ to t, $K \in \{1, \ldots, t-1\}$ indicates the context length and θ denotes the CDT policy parameters. During the training phase, CDT generates the training set $\{(o_t, a_t)\}$ by splitting trajectories in $\mathcal{T}$ into shorter contexts with length K, then it trains the policy by minimizing the prediction loss between a_t and $\pi_\theta(\cdot|o_t)$. During the inference phase, CDT selects the initial RTG R_1 as well as the CTG C_1, then it selects the action a_t based on the current R_t, C_t. It is worth noting that CDT enables the agent to dynamically adjust its constraint threshold during deployment without using a Lagrangian-type update.

Constraints Penalized Q-learning. Our algorithm relies on a component that delivers precise value estimates for the state. We take a state-of-the-art method CPQ as our primary approach and discuss its details as follows. CPQ adopts the actor-critic framework, which maintains four components at each iteration. They are a policy $\pi : \mathcal{S} \times \mathcal{A} \to [0, 1]$, a discriminator $\nu : \mathcal{S} \times \mathcal{A} \to \{0, 1\}$ that decides whether a given state-action pair (s, a) are out-of-distribution (OOD) of the behavior policy of the offline dataset; a reward critic function $Q_r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ and a cost critic function $Q_c : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$. During the training phase, CPQ first pre-trains ν by a Conditional Variational Autoencoder (CVAE) (Ivanovic et al., 2020; Yang et al., 2021). At the t-th step, CPQ maintains its current policy π. Then it updates the cost critic first, following

$$Q_c = \arg\min_{Q_c} -\alpha\mathbb{E}_{s,a\sim\mathcal{T}}[Q_c(s, a)\nu(s, a)] + \mathbb{E}_{s,a,s',c\sim\mathcal{T}}[(Q_c(s, a) - c - \gamma\mathbb{E}_{a'\sim\pi(\cdot|s')}[Q_c(s', a')])^2] \tag{2}$$

where $0 < \gamma < 1, \alpha$ are tunable parameters. Next, CPQ updates the reward critic by optimizing the following cost-penalized Bellman equation, which is

$$Q_r = \arg\min_{Q_r} \mathbb{E}_{s,a,s',r\sim\mathcal{T}}[(Q_r(s, a) - r - \gamma\mathbb{E}_{a'\sim\pi}[\mathbb{I}(Q_c(s', a') \leq \kappa)Q_r(s', a')])^2], \tag{3}$$

Algorithm 1 Relabeling trajectory set

```
Require: Trajectories dataset $\mathcal{T}$, cost limitation list $\mathcal{K} = [\kappa_1, \kappa_2, ..., \kappa_m]$, reward critic functions
    $\mathbb{Q}_r = \emptyset$, cost critic functions $\mathbb{Q}_c = \emptyset$
 1: //Run CPQ under various cost limitations to derive distinct reward critic and cost critic functions.
 2: for $\kappa_i$ in $\mathcal{K}$ do
 3:     Run CPQ(Algorithm 3) with the constraint limit $\kappa_i$, obtain the reward and cost critic $Q_r$, $Q_c$
 4:     Set $\mathbb{Q}_r[\kappa_i] = Q_r; \mathbb{Q}_c[\kappa_i] = Q_c$
 5: end for
 6: //Relabel trajectories in the dataset.
 7: for $\tau$ in $\mathcal{T}$ do
 8:     Select $\kappa^\tau = \arg\min_{\kappa_i \in \mathcal{K}} |\mathbb{Q}_c[\kappa_i](\tau.s_0, \tau.a_0) - C_0|$
 9:     Set $Q_r^\tau = \mathbb{Q}_r[\kappa^\tau]; Q_c^\tau = \mathbb{Q}_c[\kappa^\tau]$
10:     Take $\tau'$ as the output of Algorithm 2, based on $\tau$, $Q_r^\tau$ and $Q_c^\tau$, $\mathcal{T} \leftarrow \mathcal{T} \cup \{\tau'\}$
11: end for
12: Use Pareto-Frontier augmentation method in Algorithm 4 to augment $\mathcal{T}$
Ensure: New trajectory dataset $\mathcal{T}$
```

where κ is the cost constraint defined in Equation (1). Finally, given Q_r and Q_c, CPQ performs any policy optimization method to obtain π', which maximizes the following constrained optimization problem:

$$\pi' = \arg\max_{\pi} \mathbb{E}_{s\sim\mathcal{T}}\mathbb{E}_{a\sim\pi(\cdot|s)}[\mathbb{I}(Q_c(s,a) \le \kappa)Q_r(s,a)]. \tag{4}$$

4. Method

Algorithm Overview. We present a framework called CQDT, which exploits the Constrained Dynamic Programming approach, specifically CPQ, to overcome the limitations of CDT. The algorithm details are summarized in Algorithm 1. Intuitively speaking, CQDT takes a trajectory dataset $\mathcal{T}$ as its input and outputs an augmented dataset $\mathcal{T}'$ based on $\mathcal{T}$. It then trains CDT over the augmented dataset $\mathcal{T}'$. Next, we describe how to augment $\mathcal{T}$ in steps. Due to space limitations, we refer readers to the appendix in the supplementary material for a more detailed description.

First Step: Training CPQ to Obtain Value Functions. The objective of this step is to train CPQ to obtain precise estimates of action value functions across different cost limit κ settings. We maintain a list of cost limitations, $\mathcal{K}$, which includes m different cost limits, $\kappa_1, \ldots, \kappa_m$, and run CPQ m times to obtain corresponding Q_r and Q_c functions, denoted as $\mathbb{Q}_r[\kappa_i]$ and $\mathbb{Q}_c[\kappa_i]$, respectively, as outlined in line 4 of Algorithm 1. In the subsequent steps, the $\mathbb{Q}_r$ and $\mathbb{Q}_c$ lists are utilized to relabel the RTG and CTG for each trajectory.

Second Step: Relabeling Trajectories. Now we describe how to relabel a given trajectory τ using $\mathbb{Q}_r$ and $\mathbb{Q}_c$ in detail. The steps are summarized in Algorithm 2. To demonstrate that, recall that our goal is to learn π^κ that maximizes the expected return under the κ constraint. Assume that for the trajectory τ, the policy π that generates τ satisfies the constraint κ. Therefore, in order to further push the agent to learn π^κ instead of only learning π, we generate a new trajectory τ' identical to τ, with different $\tau'.R_i, \tau'.C_i$, to make τ' similar to a trajectory generated by π^κ. Our strategy is simple: we first replace the last RTG and CTG of τ' as 0. At the t-th step of τ', we would like to calculate

Algorithm 2 Relabeling one trajectory

```
Require: Trajectory $\tau$, reward critic $Q_r^\tau$ and cost critic $Q_c^\tau$
1: Set $T$ as the length of $\tau$, the new trajectory $\tau' = \tau$, $\tau'.R_{T+1} = \tau'.C_{T+1} = 0$
2: for $t = T+1, \ldots, 2$ do
3:     Set $V_r^\tau(\tau.s_t) = Q_r^\tau(\tau.s_t, \tau.a_t)$ , $V_c^\tau(\tau.s_t) = Q_c^\tau(\tau.s_t, \tau.a_t)$
4:     $\tau'.R_{t-1} \leftarrow \tau.r_{t-1} + \tau'.R_t$, $\tau'.C_{t-1} \leftarrow \tau.c_{t-1} + \tau'.C_t$
5:     if $V_c^\tau(\tau.s_t) \leq \tau'.C_t$ and $V_r^\tau(\tau.s_t) \geq \tau'.R_t$ then
6:         $\tau'.R_{t-1} \leftarrow \tau.r_{t-1} + V_r^\tau(\tau.s_t)$, $\tau'.C_{t-1} \leftarrow \tau.c_{t-1} + V_c^\tau(\tau.s_t)$
7:     end if
8: end for
Ensure: Relabeled trajectory $\tau'$
```

$\tau'.R_{t-1}$ and $\tau'.C_{t-1}$. Then we either to use the existing RTG ($\tau'.R_t$) and CTG ($\tau'.C_t$) to update $\tau'.R_{t-1}$ and $\tau'.C_{t-1}$ (line 4 in Algorithm 2), or we use the learned reward critic and cost critic to update $\tau'.R_{t-1}$ and $\tau'.C_{t-1}$ (line 6 in Algorithm 2), if the reward critic and cost critic provide a more "aggressive" approximation, i.e., $V_c^\tau(\tau.s_t) \leq \tau'.C_t$ and $V_r^\tau(\tau.s_t) \geq \tau'.R_t$ (line 5 in Algorithm 2). Here V_r^τ and V_c^τ are learned reward and cost critics, and they are selected from $\mathbb{Q}_r$ and $\mathbb{Q}_c$ to make sure that the cost constraint estimation Q_c^τ is close to the true CTG C_0 (line 8 in Algorithm 2).

The most notable difference between our relabeling strategy and the previous one for offline RL (Yamagata et al., 2023) is that we relabel RTG and CTG *jointly and simultaneously*. If we only relabel each trajectory based on their RTG and CTG separately, we might obtain unsafe trajectories, which hurts the overall performance of CQDT. Instead, our strategy ensures that, each safe trajectory will still be safe after relabeling, which is crucial for the safe RL setting. Our experimental results in the later section suggest the effectiveness of our relabeling strategy.

Third Step: Post-Processing Steps for the Final Trajectory. Now, we have produced an augmented trajectory dataset $\mathcal{T}$ which consists of the original trajectories τ and the new trajectories τ'. Finally, we introduce some additional post-processing steps over $\mathcal{T}$ from existing works (Liu et al., 2023b; Yamagata et al., 2023) for the further performance improvement of CQDT.

The first post-processing technique we adopt is to resolve the potential conflict between a high RTG and a low CTG. Due to the nature of safe RL, we would like to first guarantee the safety of our learned policy. Following Liu et al. (2023b), we use a Pareto Frontier-based data augmentation technique to further generate new trajectories and add them to $\mathcal{T}$. The second post-processing technique aims to maintain the consistency of the RTG and CTG within the input sequence of CDT.

5. Experiment

In this section, we begin by outlining the fundamental settings of our experiment. We then present the performance of CQDT through a series of experiments, each specifically designed to address a key challenge, as described below. Due to space constraints, detailed descriptions of the experimental settings and comprehensive analyses of the results are provided in the supplementary material.

- How does CQDT compare with CDT and other offline safe RL methods in terms of performance? Additionally, how does the choice of the value function affect CQDT's performance?

- Is CQDT capable of performing effective stitching?

	CarCircle		CarRun		AntRun		DroneCircle		DroneRun		Average	
	reward	cost	reward	cost	reward	cost	reward	cost	reward	cost	reward	cost
BC-Safe*	0.500	0.840	0.940	0.220	0.650	1.090	0.560	0.570	0.280	0.740	0.586	0.692
BCQ-Lag *	0.630	1.890	0.940	0.150	0.760	5.110	0.800	3.070	0.720	5.540	0.770	3.152
BEAR-Lag *	0.740	2.180	0.680	7.780	0.150	0.730	0.780	3.680	0.420	2.470	0.554	3.368
COptiDICE *	0.490	3.140	0.870	0.000	0.610	0.940	0.260	1.020	0.670	4.150	0.580	1.850
CDT*	0.750	0.950	0.990	0.650	0.720	0.910	0.630	0.980	0.630	0.790	0.744	0.856
CPQ *	0.710	0.330	0.950	1.790	0.030	0.020	-0.220	1.280	0.330	3.520	0.360	1.388
CQDT(ours)	$0.767_{\pm 0.081}$	$0.895_{\pm 0.103}$	$0.994_{\pm 0.361}$	$0.886_{\pm 0.138}$	$0.735_{\pm 0.094}$	$0.832_{\pm 0.205}$	$0.753_{\pm 0.075}$	$0.981_{\pm 0.285}$	$0.640_{0.075}$	$0.812_{\pm 0.046}$	$0.778_{\pm 0.137}$	$0.881_{\pm 0.155}$
BCQL-CDT	$0.733_{\pm 0.206}$	$0.893_{\pm 0.002}$	$0.983_{\pm 0.186}$	$1.758_{0.072}$	$0.673_{\pm 0.029}$	$0.793_{\pm 0.038}$	$0.685_{\pm 0.275}$	$0.747_{\pm 0.143}$	$0.621_{\pm 0.101}$	$0.597_{\pm 0.239}$	$0.739_{\pm 0.159}$	$0.958_{\pm 0.099}$
BEARL-CDT	$0.735_{\pm 0.227}$	$0.929_{\pm 0.178}$	$0.999_{\pm 0.059}$	$1.707_{\pm 0.134}$	$0.665_{\pm 0.229}$	$0.719_{\pm 0.162}$	$0.684_{\pm 0.034}$	$0.766_{\pm 0.122}$	$0.597_{\pm 0.266}$	$0.647_{\pm 0.091}$	$0.736_{\pm 0.163}$	$0.953_{\pm 0.137}$
Ablation①	$0.785_{\pm 0.145}$	$0.981_{\pm 0.238}$	$0.977_{\pm 0.296}$	$1.357_{\pm 0.082}$	$0.560_{\pm 0.053}$	$0.313_{\pm 0.218}$	$0.633_{\pm 0.076}$	$0.837_{\pm 0.224}$	$0.636_{\pm 0.094}$	$0.551_{\pm 0.175}$	$0.718_{\pm 0.133}$	$0.808_{\pm 0.187}$
Ablation②	$0.767_{\pm 0.201}$	$0.964_{\pm 0.230}$	$0.982_{\pm 0.186}$	$1.901_{\pm 0.110}$	$0.702_{\pm 0.194}$	$0.874_{\pm 0.076}$	$0.764_{0.263}$	$1.202_{\pm 0.130}$	$0.621_{\pm 0.149}$	$0.776_{\pm 0.051}$	$0.767_{\pm 0.198}$	$1.143_{\pm 0.119}$

Table 1: Evaluation results for normalized reward and cost are provided. **Bold**: Safe agents with a normalized cost smaller than 1. Gray: Unsafe agents. **Blue**: Safe agent achieving the highest reward. Each value is averaged over 3 distinct cost thresholds ($\kappa = 10, 20, 40$), 20 episodes and 3 seeds, following the setting in Liu et al. (2023a). Results for baselines with * are copied from Liu et al. (2023a). BCQL-CDT: Validation of the impact of the value function on algorithm performance: Using the value functions in BCQ-Lag. BEARL-CDT: Validation of the impact of the value function on algorithm performance: Using the value functions in BEAR-Lag. Ablation①: Effect of removing constraints learning by separately considering reward and cost relabeling. Ablation②: Without PF-based augmentation.

- What is the impact of the various components of CQDT on its overall performance?
- How does CQDT perform in a sparse reward environment?

5.1. Basic Experiment Setting

Tasks and Environments. We utilize established safety RL tasks within the **BulletSafetyGym** environment following Liu et al. (2023b). In our experiment, we focus on two tasks: Circle and Run, and train three types of agents: Car, Ant, and Drone.

Dataset. We utilize the offline safe RL dataset from Liu et al. (2023a). For each environment, the dataset is collected by training different implemented algorithms under gradually varied cost thresholds and collecting the generated trajectories. The algorithms employed during this procedure include CPO (Achiam et al., 2017), FOCOPS (Zhang et al., 2020), CVPO (Liu et al., 2022), among others.

Baselines and Parameters Setting. In our selection of baseline models, we have covered a variety of established offline safe RL methods, as thoroughly discussed in Liu et al. (2023a). The baselines encompass **CDT** (Liu et al., 2023b), **Behavior Cloning (BC)** (Xu et al., 2022), **COptiDICE** (Lee et al., 2022), **CPQ** (Xu et al., 2022), **BCQ-Lag** (Fujimoto et al., 2019; Stooke et al., 2020), and **BEAR-Lag**(Kumar et al., 2019; Stooke et al., 2020).

5.2. CQDT Performance Analysis

Evaluation Metrics. We employ the normalized reward return and normalized cost return as our comparison metrics (Fu et al., 2020). Let $R_{\text{max}}(\mathcal{T})$ and $R_{\text{min}}(\mathcal{T})$ denote the maximum and minimum empirical reward returns for task $\mathcal{M}$, respectively. The normalized reward is calculated as $R_{\text{norm}}(\mathcal{T}) = \frac{R_\pi(\mathcal{T}) - R_{\text{min}}(\mathcal{T})}{R_{\text{max}}(\mathcal{T}) - R_{\text{min}}(\mathcal{T})}$, where R_π represents the raw return of policy π. For the cost measure, we use the normalized cost return defined as $C_{\text{norm}}(\mathcal{T}) = \frac{C_\pi}{\kappa + \epsilon}$, where κ denotes the cost limitation of the task and ϵ is a small positive number incorporated to ensure numerical stability. The term C_π

represents the evaluated accumulated cost return of policy π. If the cost return surpasses $\kappa + \epsilon$, the normalized cost return exceeds one; otherwise, it remains within the range of one.

Result Analysis. Table 1 presents a comprehensive summary of the performance of contemporary offline RL strategies across various environments and task configurations. Performance is assessed using reward and cost metrics, with evaluation criteria outlined as follows:

- When $C_{\text{norm}} \leq 1$, a larger R_{norm} indicates superior strategy performance.
- When $C_{\text{norm}} \geq 1$, a smaller C_{norm} signifies enhanced strategy performance.
- Comparing strategies with $C_{\text{norm}} \leq 1$ and $C_{\text{norm}} \geq 1$, preference is given to the policy with $C_{\text{norm}} \leq 1$.

Compared to other baselines, our proposed CQDT method achieves maximum return while ensuring safety. CPQ, BCQ-Lag, and BEAR-Lag, three Q-learning-based safe RL methods, encounter challenges in balancing safety and reward optimization. The BC-Safe method, grounded in imitation learning and trained on provided data, exhibits suboptimal performance during the test phase. This phenomenon may be attributed to the scarcity of safe data in the training dataset, indicating a lack of robustness. COptiDICE employs novel estimators to evaluate policy constraints and achieves suboptimal rewards, while facing challenges related to adhering to strict safety constraints.

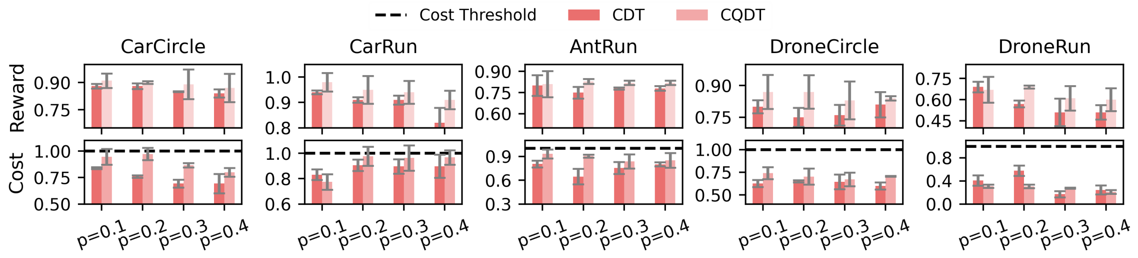

Figure 5: Verification of stitching-reward ability with $p = 0.x$ values representing various suboptimal datasets. Suboptimal datasets were generated by removing safe trajectories that fall within the top x% of cumulative rewards. Higher p-values indicate the removal of more high-reward safe paths, degrading dataset quality. The first row illustrates cumulative rewards obtained by CQDT and CDT trained on these datasets for different tasks, while the second row shows the corresponding cumulative costs. The black dashed line denotes the cost limitation.

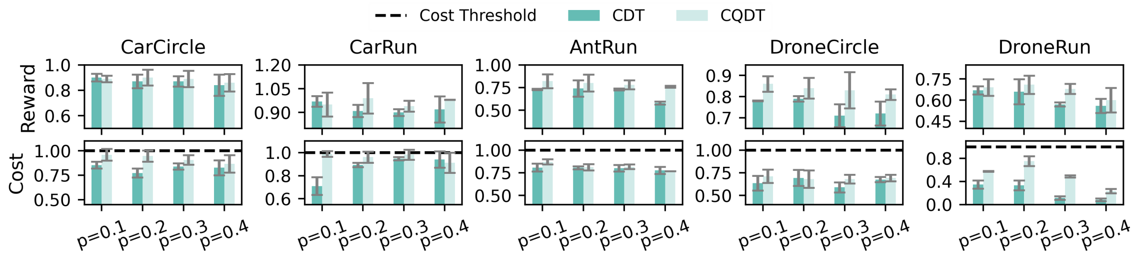

Figure 6: Verification of stitching-cost ability with $p = 0.x$ values corresponding to various suboptimal datasets. These datasets were created by excluding trajectories with low cumulative costs. As the p-value increases, fewer trajectories with small cumulative costs are retained, resulting in increasingly suboptimal datasets.

The CQDT method presented in our work leverages additional value functions to relabel trajectories, enhancing the model's stitching capability and enabling it to achieve state-of-the-art performance.

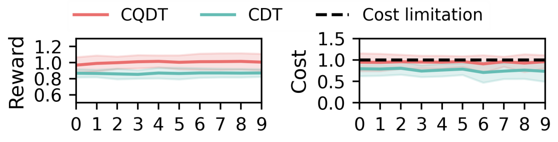

Figure 7: Comparison of CQDT and CDT performance in the Sparse-Reward DroneCircle environment with varying target return and fixed target cost.

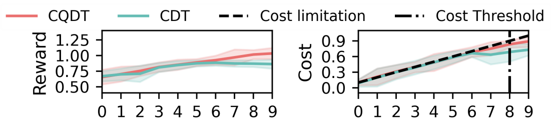

Figure 8: Comparison of CQDT and CDT performance in the Sparse-Reward DroneCircle environment with varying target costs and fixed target return.

To demonstrate the effectiveness of the CQDT method, we conduct a comprehensive comparative analysis examining the impact of various value functions on algorithmic performance. BCQ-Lag and BEAR-Lag, both grounded in offline safe RL and based on Q-Learning, are employed for this investigation. The Q_r and Q_c functions in these methodologies are employed to estimate the RTG and CTG values of the original trajectory. To signify the enhanced versions of these methods, we specifically refer to them as **BCQL-CDT** and **BEARL-CDT**, respectively.

Our experimental results, detailed in Table 1, indicate that BCQL-CDT and BEARL-CDT perform similarly to CDT in the selected tasks, although they do not reach the performance of CQDT. The performance discrepancy between BCQ-Lag and BEAR-Lag compared to CPQ suggests suboptimality in relabeling the original trajectories using their respective value functions. This contributes to the varied performance among BCQL-CDT, BEARL-CDT, and CQDT.

We also conduct two ablation studies, Ablation ① and Ablation ②, to assess the impact of various components within CQDT. The results of these experiments are presented in Table 1. In addition, we evaluate the zero-shot adaptation capability and robustness of CQDT, with further information included in the supplementary material.

5.3. The Stitching Capability of CQDT

We conduct an evaluation of CQDT's stitching capabilities for reward and cost by creating various suboptimal datasets for five tasks and comparing the performance of CQDT and CDT across these datasets.

To evaluate the reward stitching ability, we remove the top X% of trajectories with the highest RTG from batches of trajectories. As the value of X increases, more high-return trajectories are excluded from the dataset. To create a suboptimal dataset for evaluating cost stitching capability, we group trajectories based on their RTG, then we remove the trajectories with lowest CTG in each group. In detail, we divide the trajectories into $\frac{Max\ Return}{10}$ groups, where Max Return denotes the highest return among all trajectories. Within each group, we remove trajectories with the lowest X% CTG from each group. Such a setup allows us to detect the stitching ability for a safe offline RL agent, as we want her to learn safe and high-return policy from unsafe and low-return trajectories.

We present the performance of CQDT on different suboptimal datasets in Figures 5 and 6. These experiments demonstrate that as the value of X increases, leading to the removal of higher-quality trajectories, the performance of policies generated by CQDT and CDT deteriorates. We can observe that the cumulative reward decreases. However, CQDT consistently outperforms CDT, demonstrating its superior stitching ability. Even when trained with suboptimal datasets, CQDT effectively utilizes these datasets to maximize performance by leveraging its stitching capabilities. Superior performance by CQDT highlights its unique ability to stitch suboptimal trajectories, a capability not present in CDT. This stitching ability enables CQDT to achieve better overall performance.

5.4. Performance of CQDT in Sparse Reward Environment

The experiments in the previous sections demonstrate that CQDT performs well in dense reward environments. In this section, we evaluate and analyze the performance of CQDT in the sparse reward environment. Since there is no publicly available dataset or corresponding environment for sparse reward scenarios in the field of offline safe RL, we build our own environment based on existing datasets. Specifically, we select the existing DroneCircle environment to create a Sparse-Reward DroneCircle environment.

For any trajectory τ in DroneCircle, we aim to create a new trajectory τ' as follows. We consider each subsequence in τ with length 10, i.e., $\tau.r_{10k}, \tau.r_{10k+1}, ..., \tau.r_{10k+9}$. We then set $\tau' = \tau$, while it replaces $\tau'.r_{10k+9}$ with $\tau.r_{10k} + \tau.r_{10k+1} + \cdots + \tau.r_{10k+9}$, and replaces $\tau'.r_{10k+i}, 0 \leq i < 9$ with 0. Such an operation keeps the total reward of τ' unchanged, while it makes τ' a trajectory with sparse rewards. Accordingly, we use the Sparse-Reward DroneCircle Offline Dataset to train CQDT and CDT. During testing, we adopt the similar strategy, where each agent encounters 0 reward in time step $10k, 10k+1, ..., 10k+8$, and she encounters $\tau.r_{10k} + \tau.r_{10k+1} + \cdots + \tau.r_{10k+9}$ at time step $10k+9$. We do not change the cost distribution.

Figures 7 and 8 present a performance comparison between CQDT and CDT under varying target return and target cost settings in the Sparse-Reward DroneCircle environment. The results indicate that CQDT consistently outperforms CDT across various settings of target return and target cost. These experimental results demonstrate that CQDT maintains its superiority even in sparse-reward environments.

6. Conclusion and Future Work

In this work, we proposed the CQDT for safe offline RL. Our approach replaces RTG and CTG in the training data with dynamic programming-based learning-based reward return and cost return, which brings the stitching ability and addresses the weakness of the CDT. Our evaluation shows that our approach is able to outperform existing safe RL baseline algorithms. One potential future direction is to build a theoretical analysis to justify the effectiveness of our learning-based constraint approach to safe RL, similar to previous analyses applied to general goal-based RL algorithms (Brandfonbrener et al., 2022). Please refer to the supplementary material for the complete paper and the corresponding code. The link to the supplementary material is also provided for reference[1].

1. https://drive.google.com/drive/folders/1y8U2NbU7sh2_XPoOPLtK_Txusj633It-?usp=drive_link

References

Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *International conference on machine learning*, pages 22–31. PMLR, 2017.

Eitan Altman. Constrained markov decision processes with total cost criteria: Lagrangian approach and dual linear program. *Mathematical methods of operations research*, 48:387–417, 1998.

Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lučić, and Cordelia Schmid. Vivit: A video vision transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6836–6846, 2021.

Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

David Brandfonbrener, Alberto Bietti, Jacob Buckman, Romain Laroche, and Joan Bruna. When does return-conditioned supervised learning work for offline reinforcement learning? *Advances in Neural Information Processing Systems*, 35:1542–1553, 2022.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

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. *Advances in neural information processing systems*, 34:15084–15097, 2021.

Yinlam Chow, Ofir Nachum, Aleksandra Faust, Edgar Duenez-Guzman, and Mohammad Ghavamzadeh. Lyapunov-based safe policy optimization for continuous control. *arXiv preprint arXiv:1901.10031*, 2019.

Yan Duan, Xi Chen, Rein Houthooft, John Schulman, and Pieter Abbeel. Benchmarking deep reinforcement learning for continuous control. In *International conference on machine learning*, pages 1329–1338. PMLR, 2016.

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.

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.

Javier Garcıa and Fernando Fernández. A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research*, 16(1):1437–1480, 2015.

Omer Gottesman, Fredrik Johansson, Joshua Meier, Jack Dent, Donghun Lee, Srivatsan Srinivasan, Linying Zhang, Yi Ding, David Wihl, Xuefeng Peng, et al. Evaluating reinforcement learning algorithms in observational health settings. *arXiv preprint arXiv:1805.12298*, 2018.

Shangding Gu, Long Yang, Yali Du, Guang Chen, Florian Walter, Jun Wang, Yaodong Yang, and Alois Knoll. A review of safe reinforcement learning: Methods, theory and applications. *arXiv preprint arXiv:2205.10330*, 2022.

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. PMLR, 2018.

Boris Ivanovic, Karen Leung, Edward Schmerling, and Marco Pavone. Multimodal deep generative models for trajectory prediction: A conditional variational autoencoder approach. *IEEE Robotics and Automation Letters*, 6(2):295–302, 2020.

Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to trust your model: Model-based policy optimization. *Advances in neural information processing systems*, 32, 2019.

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

Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, et al. Scalable deep reinforcement learning for vision-based robotic manipulation. In *Conference on robot learning*, pages 651–673. PMLR, 2018.

Rahul Kidambi, Aravind Rajeswaran, Praneeth Netrapalli, and Thorsten Joachims. Morel: Model-based offline reinforcement learning. *Advances in neural information processing systems*, 33: 21810–21823, 2020.

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.

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.

Hoang Le, Cameron Voloshin, and Yisong Yue. Batch policy learning under constraints. In *International Conference on Machine Learning*, pages 3703–3712. PMLR, 2019.

Jongmin Lee, Cosmin Paduraru, Daniel J Mankowitz, Nicolas Heess, Doina Precup, Kee-Eung Kim, and Arthur Guez. Coptidice: Offline constrained reinforcement learning via stationary distribution correction estimation. *arXiv preprint arXiv:2204.08957*, 2022.

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

Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021.

Zuxin Liu, Zhepeng Cen, Vladislav Isenbaev, Wei Liu, Steven Wu, Bo Li, and Ding Zhao. Constrained variational policy optimization for safe reinforcement learning. In *International Conference on Machine Learning*, pages 13644–13668. PMLR, 2022.

Zuxin Liu, Zijian Guo, Haohong Lin, Yihang Yao, Jiacheng Zhu, Zhepeng Cen, Hanjiang Hu, Wenhao Yu, Tingnan Zhang, Jie Tan, et al. Datasets and benchmarks for offline safe reinforcement learning. *arXiv preprint arXiv:2306.09303*, 2023a.

Zuxin Liu, Zijian Guo, Yihang Yao, Zhepeng Cen, Wenhao Yu, Tingnan Zhang, and Ding Zhao. Constrained decision transformer for offline safe reinforcement learning. *arXiv preprint arXiv:2302.07351*, 2023b.

Martin Pecka and Tomas Svoboda. Safe exploration techniques for reinforcement learning–an overview. In *Modelling and Simulation for Autonomous Systems: First International Workshop, MESAS 2014, Rome, Italy, May 5-6, 2014, Revised Selected Papers 1*, pages 357–375. Springer, 2014.

Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. *arXiv preprint arXiv:1910.00177*, 2019.

Nicholas Polosky, Bruno C Da Silva, Madalina Fiterau, and Jithin Jagannath. Constrained offline policy optimization. In *International Conference on Machine Learning*, pages 17801–17810. PMLR, 2022.

Alex Ray, Joshua Achiam, and Dario Amodei. Benchmarking safe exploration in deep reinforcement learning. *arXiv preprint arXiv:1910.01708*, 7(1):2, 2019.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International conference on machine learning*, pages 1889–1897. PMLR, 2015.

Shai Shalev-Shwartz, Shaked Shammah, and Amnon Shashua. Safe, multi-agent, reinforcement learning for autonomous driving. *arXiv preprint arXiv:1610.03295*, 2016.

Adam Stooke, Joshua Achiam, and Pieter Abbeel. Responsive safety in reinforcement learning by pid lagrangian methods. In *International Conference on Machine Learning*, pages 9133–9143. PMLR, 2020.

Ashish Vaswani. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017.

Qing Wang, Jiechao Xiong, Lei Han, Han Liu, Tong Zhang, et al. Exponentially weighted imitation learning for batched historical data. *Advances in Neural Information Processing Systems*, 31, 2018.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pages 38–45, 2020.

Haoran Xu, Xianyuan Zhan, and Xiangyu Zhu. Constraints penalized q-learning for safe offline reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 8753–8760, 2022.

Taku Yamagata, Ahmed Khalil, and Raul Santos-Rodriguez. Q-learning decision transformer: Leveraging dynamic programming for conditional sequence modelling in offline rl. In *International Conference on Machine Learning*, pages 38989–39007. PMLR, 2023.

Jingkang Yang, Kaiyang Zhou, Yixuan Li, and Ziwei Liu. Generalized out-of-distribution detection: A survey. *arXiv preprint arXiv:2110.11334*, 2021.

Tianhe Yu, Aviral Kumar, Rafael Rafailov, Aravind Rajeswaran, Sergey Levine, and Chelsea Finn. Combo: Conservative offline model-based policy optimization. *Advances in neural information processing systems*, 34:28954–28967, 2021.

Yiming Zhang, Quan Vuong, and Keith Ross. First order constrained optimization in policy space. *Advances in Neural Information Processing Systems*, 33:15338–15349, 2020.