

---

# Rebalancing Return Coverage for Conditional Sequence Modeling in Offline Reinforcement Learning

---

Wensong Bai<sup>1</sup>, Chufan Chen<sup>1</sup>, Yichao Fu<sup>1</sup>, Qihang Xu<sup>1</sup>, Chao Zhang<sup>1,2,\*</sup>, Hui Qian<sup>1</sup>

<sup>1</sup>College of Computer Science and Technology, Zhejiang University, Hangzhou, China

<sup>2</sup>Advanced Technology Institute, Zhejiang University, Hangzhou, China

{wensongb, chufan. chen, fuyichao, 22351233, zczju, qianhui}@zju.edu.cn

## Abstract

Recent advancements in offline reinforcement learning (RL) have underscored the capabilities of conditional sequence modeling (CSM), a paradigm that models the action distribution conditioned on both historical trajectories and target returns associated with each state. However, due to the imbalanced return distribution caused by suboptimal datasets, CSM is grappling with a serious distributional shift problem when conditioning on high returns. While recent approaches attempt to empirically tackle this challenge through return rebalancing techniques such as weighted sampling and value-regularized supervision, the relationship between return rebalancing and the performance of CSM methods is not well understood. In this paper, we reveal that both expert-level and full-spectrum return-coverage critically influence the performance and sample efficiency of CSM policies. Building on this finding, we devise a simple yet effective return-coverage rebalancing mechanism that can be seamlessly integrated into common CSM frameworks, including the most widely used one, Decision Transformer (DT). The resulting CSM algorithm, referred to as Return-rebalanced Value-regularized Decision Transformer (RVDT), integrates both implicit and explicit return-coverage rebalancing mechanisms, and achieves state-of-the-art performance in the D4RL experiments.

## 1 Introduction

Offline RL aims to learn effective policies solely from previously collected datasets, without any further environment interaction [40, 17, 25, 30]. Recent advancements in offline RL primarily focus on the use of conditional sequence modeling (CSM), where the core idea is to learn the return-conditioned distribution of actions at each state given historical information [61, 25, 65, 66, 26]. One of the most representative methods in CSM is the decision transformer (DT) [11], wherein historical trajectories composed of multiple return-state-action triplets are exploited to train a Transformer-based model. During inference time, the model predicts a sequence of actions conditioned on a target return, effectively reframing the offline RL problem as a supervised learning task. By enabling the policy to make decisions directly from temporally extended trajectories, this paradigm circumvents the need for recursive value estimation, thereby avoiding both the challenges of Bellman completeness and the bootstrapping errors associated with dynamic programming-based methods [11, 68].

However, CSM-based algorithms encounter substantial difficulties when learning from suboptimal datasets, particularly due to the imbalance between near-optimal and low-return trajectories within datasets [34, 42]. This uneven distribution of trajectory quality in the training dataset may induce a

---

\*Corresponding author: zczju@zju.edu.cn.

significant distributional shift problem at inference time when conditioning on high returns, critically limiting the achievable performance of CSM methods [14, 10]. Recent advances have acknowledged such difficulties and proposed various mitigation strategies by rebalancing return distributions [48, 23, 42, 57, 25]. For instance, DiffStitch [42] employs diffusion models to synthesize high-return trajectories by stitching together optimal sub-segments drawn from multiple suboptimal trajectories. Critic-Guided Decision Transformer (CGDT) [57] and Q-value Regularized Transformer (QT) [25] integrate value-based guidance into the DT framework, implicitly assigning greater importance to near-optimal actions, thereby steering policies toward higher-return regions. These methods achieve enhanced performance through *explicit or implicit rebalancing of the return distribution*, and their efficacy has been empirically validated by achieving state-of-the-art (SOTA) results [42, 57, 25]. Nonetheless, none of these prior works provides a theoretical analysis elucidating the fundamental principles underpinning their empirical successes.

In this paper, we uncover the relationship between the return distribution of the training dataset and the performance of CSM policies at inference. Specifically, two aspects of return-coverage are relevant to the performance: (i) *expert-level return-coverage*, corresponding to the coverage of near-optimal and expert-level returns; (ii) *full-spectrum return-coverage*, which corresponds to the return-coverage over the entire return space<sup>2</sup>. By analyzing the coverage of these two returns within the offline datasets, we derive theoretical bounds quantifying the performance gap between the learned CSM policy (given fixed inference-time conditioning functions) and the optimal policy  $\pi^*$  of the underlying MDP. Our analysis reveals that the performance gap is jointly influenced by both expert-level and full-spectrum return-coverage, providing a principled explanation for why return distribution rebalancing enhances policy performance and mitigates distributional shift at inference.

Building upon this finding, we propose a simple yet effective return-coverage rebalancing mechanism, which formulates rebalancing as a KL regularization between the agent’s policy and an expert policy extracted from near-optimal trajectories in the dataset. The proposed mechanism is explicitly interpretable and seamlessly compatible with most existing CSM algorithms as a plug-in module. Leveraging this mechanism, we further design a practical CSM algorithm named Return-rebalanced Value-regularized Decision Transformer (RVDT), which achieves consistent performance improvements over state-of-the-art methods. Our contributions are summarized as follows:

1. We theoretically show how the performance of CSM-based policies is influenced by the coverage of expert-level and full-spectrum conditional returns in the offline training dataset. Specifically, we show that the gap between the expected return of the learned CSM policy and the optimal return is inversely proportional to the coverage of these two returns, and we derive an upper bound to explicitly characterize the associated sample complexity.
2. We propose an explicit return-coverage rebalancing mechanism as a plug-in module for enhancing existing CSM algorithms. In practice, an independent network is trained via imitation learning on expert-level trajectories to extract expert policies, whose KL divergence with the agent’s policies is used as a regularizer. This regularizer has been shown to perform weighted sampling explicitly over trajectories when the policy is parameterized by a Gaussian distribution.
3. We design RVDT, which utilizes both explicit and implicit rebalancing mechanisms, effectively addressing the difficulties posed by limited expert data through the utilization of an expert KL divergence regularizer and a value-regularized guidance. This combination collectively mitigates inference-time distributional shift, enabling superior performance in challenging scenarios such as sparse-reward environments and low-quality datasets.

The performance of RVDT is evaluated on the D4RL benchmarks [16]. Experimental results demonstrate significant performance gains over representative baselines, establishing RVDT as a strong candidate for advancing the state-of-the-art (SOTA) in offline RL.

## 2 Related Work

Learning from imbalanced datasets, comprising predominantly low-return trajectories and relatively few high-return trajectories, has long been a critical challenge in offline RL [23, 21, 14, 10]. Broadly,

---

<sup>2</sup>The full-spectrum return-coverage is closely related to the runtime returns, i.e., the realistic returns during policy execution, as the actual distribution of runtime conditional returns during execution is typically uncertain.

prevailing methods tackling this challenge can be categorized into two types: *explicit rebalancing* and *implicit rebalancing*. Explicit methods encompass techniques such as dataset selection [12, 48, 32], weighted sampling [44, 21, 23], and data synthesis [62, 34, 42], while implicit methods generally leverage value estimation as additional guidance [38, 15, 17, 9, 57, 25].

**Sub-dataset Selection.** Several methods explicitly select informative subsets from imbalanced datasets, e.g., sample uncertainty [12, 48] or frequency of being forgotten [54] is utilized to prune the datasets. Other studies employ submodular functions that capture the notions of diversity, representation, and coverage to eliminate redundancy [52, 60, 59, 32]. A more straightforward approach, top-% BC, directly selects and fits only the highest-return trajectories within the dataset [39, 11].

**Weighted Sampling.** Another line of research adopts weighted sampling techniques to rebalance trajectory distributions [24, 41, 22, 44]. Hong et al. [21] leverage a Boltzmann distribution based on trajectory returns or advantages to reweight trajectories. Similarly, density-ratio weighting [23] utilizes importance sampling [49] to emulate sampling from other supplementary datasets.

**Data Augmentation and Generation.** Other methods address trajectory imbalance by augmenting or synthesizing data [56, 45, 64, 62, 34, 42]. For instance, QDT [62] relabels return tokens using a learned conservative Q-function, subsequently integrating these augmented trajectories into the training datasets. DiffStitch [42] leverages diffusion models to generate high-return trajectories by stitching optimal sub-segments extracted from multiple suboptimal trajectories.

**Value-regularized Supervised Learning.** Recent works such as BEAR [38], CDC [15], O-RL [7], TD3+BC [17], TD3+RKL [9], CGDT [57], and QT [25] integrate value-based guidance into supervised learning frameworks to prioritize high-return actions. As a result of leveraging transition-level value estimates to prioritize near-optimal actions without explicitly modifying the trajectory return distribution, these algorithms are considered as implicit return rebalancing methods.

### 3 Preliminary

#### 3.1 Offline Reinforcement Learning

We model an agent-environment interaction as a finite horizon Markov Decision Process (MDP)  $(\mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{T}, \mu_0, H)$ , where  $\mathcal{S}$  and  $\mathcal{A}$  denote the (continuous) state and action spaces.  $\mathcal{R} : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$  is the reward function,  $\mathcal{T}$  defines the transition dynamics, i.e.,  $s_{t+1} \sim \mathcal{T}(\cdot | s_t, a_t)$ .  $\mu_0$  is the initial state distribution, i.e.,  $s_1 \sim \mu_0$ , and  $H$  is the time horizon. In the offline RL setup, algorithms are given a static dataset  $\mathcal{D}$  consisting of trajectories  $\tau = \{s_1, a_1, r_1, \dots, s_H, a_H, r_H\}$ , collected by a behavior policy  $\pi_\beta$ . Let  $g(\tau_t) = \sum_{i=t}^H r_i$  (abbreviated as  $g_t$ ) denote the return-to-go (RTG) of a trajectory at time step  $t$ , and  $\mathcal{J}(\pi) = \mathbb{E}_{\tau \sim \pi} [g(\tau)]$  is the expected return of a policy  $\pi$ .

#### 3.2 Return Conditioned Sequence Modeling

Motivated by the wide-ranging applications and remarkable success of transformer models in various domains [55, 8, 11, 27, 29, 36], CSM-based methods formulate policy learning as a return-conditioned supervised learning problem. Specifically, during training, the policy  $\pi$  is optimized by minimizing the empirical negative log-likelihood (NLL) loss over a dataset  $\mathcal{D}$  of trajectories:

$$\mathcal{L}(\pi) = -\sum_{\tau \in \mathcal{D}} \sum_{1 \leq t \leq H} \log \pi(a_t | s_t, g(\tau_t), \bar{\tau}_{t-1}^K), \quad (1)$$

where  $\bar{\tau}_{t-1}^K = \{g_{t-K}, s_{t-K}, a_{t-K}, \dots, g_{t-1}, s_{t-1}, a_{t-1}\}$  denotes the historical input with context length  $K$ . During inference, the learned policy  $\pi$  is combined with a conditioning function  $f(s)$  to form the inference-time policy  $\pi_f$  as  $\pi_f(a|s) := \pi(a|s, f(s), \bar{\tau}_{t-1}^K)$ , where  $s$  denotes the collection of  $s$  and historical triplets. Algorithms such as DT and its variants, e.g., QT [25] and CGDT [57], can be viewed as specific instantiations within this framework.

The NLL loss can be interpreted as approximating the conditional distribution  $P_{\pi_\beta}(a|s, g)$ , where  $P_{\pi_\beta}$  denotes the distribution over states, actions, and returns induced by the behavior policy  $\pi_\beta$ . In the regime of infinite data, it has been shown that the CSM objective effectively reweights the behavior policy based on the distribution of future returns. Under this setting, the optimal return-conditioned policy  $\pi_f^{\text{CSM}}$  for a given conditioning function  $f$  can be expressed as:  $\pi_f^{\text{CSM}}(a|s) =$

$\pi_\beta(a|s) \frac{P_{\pi_\beta}(f(s)|s, a)}{P_{\pi_\beta}(f(s)|s)}$ . With infinite data and a fully expressive policy class, optimizing (1) guarantees that the learned policy  $\pi_f^{\text{CSM}}$  is provably near-optimal [6].

**Theorem 1.** Consider a finite-horizon MDP with horizon  $H$ , a behavior policy  $\pi_\beta$ , and a conditioning function  $f$ . Suppose the following conditions hold: (i) The return is sufficiently covered, i.e.,  $P_{\pi_\beta}(g = f(s_1)|s_1) \geq \alpha_f$  for all initial states  $s_1$ . (ii) The environment is nearly deterministic in the sense that for some functions  $\mathcal{T}$  and  $\mathcal{R}$ , i.e.,  $P(r \neq \mathcal{R}(s, a) \text{ or } s' \neq \mathcal{T}(s, a)|s, a) \leq \epsilon, \forall s, a$ . (iii) The conditioning function  $f$  is consistent such that  $f(s) = f(s') + r, \forall s$ . Then the policy that optimizes the CSM objective (Eq. 1) is guaranteed to be near-optimal:  $\mathbb{E}_{s_1} f(s_1) - J(\pi_f^{\text{CSM}}) \leq (1/\alpha_f + 2) H^2 \epsilon$ .

Theorem 1 establishes theoretical guarantees and success conditions for CSM methods. Brandfonbrener et al. [6] further analyze the performance gap between  $\pi_{f^*}^{\text{CSM}}$  and  $\pi^*$ , where  $f^*$  corresponds to the optimal conditioning function. However, at inference time, the runtime conditional returns, computed as the prior returns minus sampled rewards  $r$ , cannot consistently remain optimal due to the inherent randomness of  $r$  [5, 13]. This discrepancy between theory and practice substantially constrains its practical utility for enhancing the robustness and generalization of CSM algorithms.

## 4 Methodology

This paper focuses on investigating how return distribution of the training dataset impacts the performance of CSM policies through the lens of return-coverage. We theoretically reveal that the performance gap between the MDP's optimal policy  $\pi^*$  and the learned CSM policy is jointly determined by both expert-level and full-spectrum return-coverage. Likewise, we establish that the sample complexity is similarly governed by these two aspects of coverage. The theoretical insights elucidate why previous approaches, utilizing explicit or implicit rebalancing of return distributions, effectively improve performance. Following this notion, we propose a simple yet effective return-coverage rebalancing mechanism, which is explicitly interpretable, and seamlessly compatible as a plug-in module with most existing CSM algorithms. To demonstrate the effectiveness of this mechanism, we integrate it into QT [25], the current SOTA, resulting in a new approach termed RVDT.

### 4.1 Analysis on Return-Coverage Rebalancing for Conditional Sequence Modeling

Due to the imbalance between expert-level and low-return trajectories in offline datasets, the study of Brandfonbrener et al. [6], which focuses on the optimal conditioning function  $f^*$ , becomes impractical for real-world scenarios. Our analysis extends this perspective by considering a more general function class that encompasses both potentially non-optimal runtime conditioning functions and optimal (or near-optimal) conditioning functions, denoted as  $f$  and  $f^*$ , respectively. The runtime conditional return function  $f$  corresponds to arbitrary possible returns that may be encountered during policy execution. Let  $\mathcal{G}$  denote the collection of all possible returns collected by  $\pi \in \Pi$ , then  $f : \mathcal{S} \rightarrow \mathcal{G}$ . The target conditional function  $f^*$  that we aim to find is the RTG under the optimal policy  $\pi^*$ , i.e.,  $f(s) = \max_\pi \mathbb{E}_\pi[g(s)]$ . We define the expert-level return-coverage in the offline dataset generated by behavior policy  $\pi_\beta$  as  $P_{\pi_\beta}(g = f^*(s_1)|s_1)$ , where the  $f^*$  corresponds to the optimal policy  $\pi^*$  of the underlying MDP. Correspondingly, the full-spectrum return-coverage is defined as  $P_{\pi_\beta}(g = f(s_1)|s_1)$ . We first analyze the performance of CSM policies under an idealized setting with infinite data and a fully expressive policy class, establishing an upper bound on the suboptimality of CSM policies and quantifying its dependence on both aspects of return-coverage.

**Theorem 2** (Performance gap with respect to return-coverage). Consider a finite-horizon MDP with horizon  $H$ , behavior policy  $\pi_\beta$ , a runtime conditioning function  $f$ , and the optimal conditioning function for  $\pi^*$  is  $f^*$ . Assume the following assumptions hold: (i) **Return-coverage:**  $P_{\pi_\beta}(g = f(s_1)|s_1) \geq \alpha_f$  and  $P_{\pi_\beta}(g = f^*(s_1)|s_1) \geq \alpha_f^*$  for all initial states  $s_1$ . (ii) **Near determinism:**  $P(r \neq \mathcal{R}(s, a) \text{ or } s' \neq \mathcal{T}(s, a)|s, a) \leq \epsilon$  at all  $(s, a)$  for some  $\mathcal{T}$  and  $\mathcal{R}$ . (iii) **Consistency of  $f$ :**  $f(s) = f(s') + r$  for all  $s$ . Then the following upper bound holds:

$$J(\pi^*) - J(\pi_f^{\text{CSM}}) \leq \left( \frac{1}{\alpha_f^*} + 3 \right) H^2 \epsilon + \left( \frac{1}{\alpha_f} + \frac{1}{\alpha_f^*} \right) H^2 C, \quad (2)$$

where  $C \in (0, 1)$  is a constant. The proof is provided in Appendix A.2.

The result formally characterizes how the effectiveness of policies trained via CSM is constrained by both expert-level and full-spectrum return-coverage. This implies that algorithms are capable of modulating  $\alpha_f^*$  and  $\alpha_f$  through rebalancing the return distribution, thereby facilitating the convergence of  $\pi_f^{\text{CSM}}$  toward the optimal policy  $\pi^*$ . Notably, in the offline RL setting, the relationship  $\alpha_f + \alpha_f^* \leq 1$  holds when  $f \neq f^*$ . In other words, strategies such as sub-dataset selection or weighted sampling that attempt to increase one type of return-coverage without altering the distribution's support tend to reduce the other.

The subsequent analysis examines the sample efficiency of CSM methods under common assumptions as in [6], with finite data and a restricted policy class. Letting  $N$  denote the size of the dataset, Theorem 3 demonstrates that the sample complexity exhibits a structural dependence on the return-coverage analogous to that in Theorem 2. Note that for clarity, we consider the case of a single  $s$  ( $\bar{\tau}_{t-1}^K = \emptyset$ ) here, which can be readily extended to the non-empty  $\bar{\tau}_{t-1}^K$  scenario.

**Theorem 3** (Sample complexity). *To get finite data guarantees, add to the above assumptions in Theorem 2 that (i) bounded occupancy mismatch:  $P_{\pi_f^{\text{CSM}}}(s) \leq C_f \cdot P_{\pi_\beta}(s)$  for all  $s$ ; (ii) the policy class  $\Pi$  is finite; (iii)  $|\log \pi(a|s, g) - \log \pi(a'|s', g')| \leq c$  for any  $(a, s, g, a', s', g')$  and all  $\pi \in \Pi$ ; (iv) the approximation error of  $\Pi$  is bounded by  $\epsilon_{\text{approx}}$ , i.e.,  $\min_{\pi \in \Pi} L(\pi) \leq \epsilon_{\text{approx}}$ .*

Define the expected loss as  $L(\hat{\pi}) = \mathbb{E}_{s \sim P_{\pi_\beta}} \mathbb{E}_{g \sim P_{\pi_\beta}(\cdot|s)} [\text{KL}(P_{\pi_\beta}(\cdot|s, g) \parallel \hat{\pi}(\cdot|s, g))]$ . Then for any estimated CSM policy  $\hat{\pi}_f$  that conditions on  $f$  at inference time, with probability at least  $1 - \delta$ ,

$$J(\pi^*) - J(\hat{\pi}_f) \leq O \left( \left[ \frac{C_f}{\alpha_f} \sqrt{c} \left( \frac{\log |\Pi|/\delta}{N} \right)^{1/4} + \frac{C_f}{\alpha_f} \sqrt{\epsilon_{\text{approx}}} + \frac{\epsilon + C}{\alpha_f^*} + \frac{C}{\alpha_f} \right] H^2 \right). \quad (3)$$

The proof, provided in Appendix A.3, utilizes uniform convergence results from supervised learning theory [50].

Theorem 3 quantifies the sample complexity of CSM methods in terms of expert-level and full-spectrum return-coverage. Together with Theorem 2, our analysis unveils the fundamental principles underpinning the empirical successes of current return-coverage rebalancing strategies. That is, rebalancing  $\alpha_f^*$  and  $\alpha_f$  reduces the gap between the learned CSM policies and  $\pi^*$ .

## 4.2 Plug-in Return-Coverage Rebalancing Mechanism for CSM

Building upon the theoretical insights established in preceding theorems, we propose an explicit rebalancing mechanism based on KL divergence regularization with respect to a learned expert policy  $\pi^e$ . Specifically, a subset of near-optimal trajectories is filtered from the offline dataset, denoted as  $\mathcal{D}_e \subset \mathcal{D}$ , from which  $\pi^e$  is extracted via imitation learning. Given a CSM policy  $\pi_\theta$  parameterized by DT, our proposed rebalancing mechanism operates as a plug-in module appended to the original NLL loss in Eq. 1. We refer to the resulting CSM algorithm as Return-rebalanced Decision Transformer (RDT), which is optimized with the following loss:

$$\mathcal{L}_{\text{RDT}}(\theta) = \mathbb{E}_{\tau \sim \mathcal{D}} \left[ \sum_{i=1}^H -\log \pi_\theta(a_i|s_i, g(\tau_i), \bar{\tau}_{t-1}^K) \right] + \alpha \mathbb{E}_{\tau \sim \mathcal{D}_e} \sum_{i=1}^H \text{KL}[\pi_\theta(\cdot|s_i, g(\tau_i), \bar{\tau}_{t-1}^K) \parallel \pi^e(\cdot|s_i)]. \quad (4)$$

The first term in Eq. (4) aligns the model with the empirical action distribution of  $\pi_\beta$ , while the regularization term guides the policy towards expert-level actions. Notably, the rebalancing module adopts KL divergence instead of MSE, designed to maintain compatibility with broader policy classes in modern CSM methods, such as diffusion models [69, 47].

To illustrate how the expert KL regularizer explicitly rebalances the return-coverage, we demonstrate that under a common stochastic policy parameterization, the regularization process can be equivalently interpreted as trajectory-level reweighting of expert-level samples in the offline dataset.

**Proposition 1** (KL regularization as weighted sampling strategy). *Assume the policy  $\pi_\theta$  is parameterized by a factorized Gaussian distribution with a fixed standard deviation. Then optimizing the RDT objective in (4) is equivalent to optimizing the following weighted NLL loss (proof in A.4):*

$$\arg \min_{\pi_\theta} \mathcal{L}_{\text{RDT}}(\theta) = \arg \min_{\pi_\theta} \mathbb{E}_{\tau \sim \mathcal{D}} \left[ (1 + \alpha \cdot \mathbb{I}[\tau \in \mathcal{D}_e]) \cdot \left( \sum_{i=1}^H -\log \pi_\theta(a_i|s_i, g(\tau_i), \bar{\tau}_{t-1}^K) \right) \right]. \quad (5)$$

Proposition 1 shows that the KL regularization effectively performs a weighted resampling of expert trajectories, practically increasing the expert-level return-coverage within the offline dataset. Though an optimal rebalancing ratio exists, it is generally infeasible to compute explicitly. In practice, our rebalancing mechanism, akin to other rebalancing strategies [17, 9, 21], amplifies the gradient contributions of expert-level behaviors during training.

### 4.3 Return-rebalanced Value-regularized Decision Transformer

We integrate our proposed rebalancing mechanism into the current SOTA method, QT, resulting in a novel algorithm that incorporates both explicit and implicit return-coverage rebalancing mechanisms, referred to as RVDT. By implicitly rebalancing the return-coverage with a Q-value regularizer to DT, the loss function of QT is defined as follows.

$$\mathcal{L}_{\text{QT}}(\theta) = \mathbb{E}_{\tau \sim \mathcal{D}} \left[ \sum_{i=1}^H -\log \pi_\theta(a_i | s_i, g(\tau_i), \bar{\tau}_{t-1}^K) \right] - \eta \mathbb{E}_{\tau \sim \mathcal{D}} \mathbb{E}_{s_i \sim \tau, a_i \sim \pi_\theta} [Q^{\pi_\theta}(s_i, a_i)]. \quad (6)$$

However, accurate Q-value estimation can be prohibitive in complex environments, limiting the effectiveness of QT’s value-based rebalancing mechanism. In contrast, RVDT uniquely combines explicit and implicit rebalancing mechanisms, i.e., an expert KL divergence regularizer and an value-regularized guidance, enhancing its adaptability to challenging offline datasets. The resulting learning objective for RVDT augments policy regularization with value-based guidance:

$$\begin{aligned} \mathcal{L}_{\text{RVDT}}(\theta) = & \mathbb{E}_{\tau \sim \mathcal{D}} \left[ \sum_{i=1}^H -\log \pi_\theta(a_i | s_i, g(\tau_i), \bar{\tau}_{t-1}^K) \right] - \eta \mathbb{E}_{\tau \sim \mathcal{D}} \mathbb{E}_{s_i \sim \tau, a_i \sim \pi_\theta} [Q^{\pi_\theta}(s_i, a_i)] \\ & + \alpha \mathbb{E}_{\tau \sim \mathcal{D}_e} \sum_{i=1}^H \text{KL}[\pi_\theta(\cdot | s_i, g(\tau_i), \bar{\tau}_{t-1}^K) \| \pi^e(\cdot | s_i)]. \end{aligned} \quad (7)$$

We adopt the same parameterization strategy as QT and employ offline DP to iteratively learn the value function  $Q^{\pi_\theta}$ . The coefficient  $\eta$  follows a commonly adopted normalization scheme [17], i.e.,  $\eta = \frac{\bar{\eta}}{\mathbb{E}_{\tau \sim \mathcal{D}} \mathbb{E}_{(s_i, a_i) \sim \tau} [Q_\psi(s_i, a_i)]}$ , where  $\bar{\eta}$  is a tunable hyperparameter. The training and inference details, along with the algorithm pseudocode, are provided in Appendix B.

Compared to QT, RVDT not only inherits its strengths but also benefits from directly aligning its policy distribution with expert-level trajectories, thereby significantly enhancing performance in sparse-reward tasks where accurate Q-value estimation is infeasible. While empirical evidence (Section 5) confirms the superiority of RVDT, we provide a theoretical perspective to substantiate its superiority over related methods under idealized conditions (i.e., infinite data and neglecting approximation errors). The formal statement and proof of Proposition 2 are provided in Appendix A.5.

**Proposition 2 (Informal).** *Let  $\pi_e^*$ ,  $\pi_w^*$ , and  $\pi_{qt}^*$  denote the optimal policies learned from the objectives of RVDT, RDT, and QT, respectively. Given the behavior policy  $\pi_\beta$ , there exists an expert-level subset  $\mathcal{D}_e \subseteq \mathcal{D}$  and an appropriate coefficient  $\alpha \geq 0$  such that the following inequalities hold for all states  $s \sim d_{\pi_\beta}$ :  $\mathbb{E}_s V^{\pi_e^*}(s) \geq \mathbb{E}_s V^{\pi_w^*}(s) \geq \mathbb{E}_s V^{\pi_\beta}(s)$  and  $\mathbb{E}_s V^{\pi_e^*}(s) \geq \mathbb{E}_s V^{\pi_{qt}^*}(s) \geq \mathbb{E}_s V^{\pi_\beta}(s)$ .*

## 5 Experiment

We conduct a comprehensive empirical evaluation of our proposed approach on the D4RL benchmarks [16]. RVDT is compared against a broad range of baselines, mainly composed of *DP-based methods* [37, 19, 40], *value-regularized SL* [38, 7, 17], and *CSM approaches* [11, 51, 62, 28, 57, 63, 25]. To further assess the robustness of RVDT, we design a set of more challenging datasets to compare RVDT with the current SOTA method (QT) [25], focusing on three key desired properties: (i) effective utilization of limited expert data, (ii) learning ability under low-data regimes [46], and (iii) trajectory stitching ability. Finally, we perform carefully designed ablation studies to isolate and examine the contributions of key components in RVDT, including the KL regularization-based return-coverage rebalancing mechanism, the Q-value guidance, and the stochastic policy modeling.

**Benchmarks and Baselines.** We evaluate our method across five diverse domains from the D4RL benchmarks [16]: Gym, Adroit, Kitchen, Maze2D, and AntMaze, encompassing tasks from continuous control to multi-task manipulation and long-horizon planning. Our approach is compared against a broad set of baselines, including CSM-based methods (DT [11], StAR [51], QDT [62], GDT [28], CGDT [57], QT [25]), DP-based methods (IQL [37], BCQ [19], CQL [40]), value-regularized SL methods (BEAR [38], O-RL [7], TD3+BC [17]), and diffusion-based methods (Diffuser [31], Decision Diffuser [2], Diffusion-QL [58]). We further include a model-based approach (MoRel [33]) for broader comparison. QT is reproduced using its official implementation, while other baselines follow their original publications or results reported in QT’s paper, depending on availability. Additional benchmark and implementation details are provided in Appendix C.1 and D.

### 5.1 Main Results on D4RL Benchmarks

The experimental results comparing RVDT against baselines on the D4RL benchmarks are presented in Table 1. RVDT consistently outperforms most baselines across various domains, establishing new SOTA performance on the majority of D4RL tasks. Task-level comparisons are provided below.

Table 1: Performance comparison across D4RL tasks. For each task, RVDT reports the mean and standard error of normalized scores [16] over 30 random rollouts and averaged over 5 random seeds.

Gym Tasks	CQL	IQL	BCQ	TD3+BC	MoRel	BC	DD	DT	StAR	GDT	CGDT	QT	<b>RVDT</b>
halfcheetah-m-e	91.6	86.7	69.6	90.7	53.3	55.2	90.6	86.8	93.7	93.2	93.6	93.2	<b>94.4 ± 0.1</b>
hopper-m-e	105.4	91.5	109.1	98.0	108.7	52.5	111.8	107.6	111.1	111.1	107.6	113.0	<b>113.1 ± 0.5</b>
walker2d-m-e	108.8	109.6	67.3	110.1	95.6	107.5	108.8	108.1	109.0	107.7	109.3	112.0	<b>112.7 ± 1.6</b>
halfcheetah-m	49.2	47.4	41.5	48.4	42.1	42.6	49.1	42.6	42.9	42.9	43.0	51.0	<b>51.9 ± 0.3</b>
hopper-m	69.4	66.3	65.1	59.3	95.4	52.9	79.3	67.6	59.5	77.1	96.9	99.6	<b>100.2 ± 0.1</b>
walker2d-m	83.0	78.3	52.0	83.7	77.8	75.3	82.5	74.0	73.8	76.5	79.1	87.2	<b>90.2 ± 0.1</b>
halfcheetah-m-r	45.5	44.2	34.8	44.6	40.2	36.6	39.3	36.6	36.8	40.5	40.4	48.8	<b>53.8 ± 2.0</b>
hopper-m-r	95.0	94.7	31.1	60.9	93.6	18.1	100.0	82.7	29.2	85.3	93.4	102.1	<b>103.2 ± 1.9</b>
walker2d-m-r	77.2	73.9	13.7	81.8	49.8	32.3	75.0	79.4	39.8	77.5	78.1	97.8	<b>99.3 ± 0.8</b>
Average	80.6	77.0	53.8	75.3	72.9	52.6	81.8	76.2	66.2	79.1	82.4	89.4	<b>91.2</b>
Adroit Tasks	CQL	IQL	BCQ	BEAR	O-RL	BC	DD	D-QL	DT	StAR	GDT	QT	<b>RVDT</b>
pen-human	37.5	71.5	66.9	-1.0	90.7	63.9	66.7	72.8	79.5	77.9	92.5	111.9	<b>127.2 ± 5.5</b>
hammer-human	4.4	1.4	0.9	0.3	0.2	1.2	1.9	0.2	3.7	3.7	5.5	10.4	<b>24.0 ± 1.5</b>
pen-cloned	39.2	37.3	50.9	26.5	60.0	37.0	42.8	57.3	75.8	33.1	86.2	85.8	<b>117.8 ± 8.6</b>
hammer-cloned	2.1	2.1	0.4	0.3	2.0	0.6	1.7	3.1	3.0	0.3	8.9	11.8	<b>21.3 ± 2.7</b>
Average	20.8	28.1	29.8	6.5	38.2	25.7	28.3	33.4	40.5	28.8	48.3	55.0	<b>72.6</b>
Kitchen Tasks	CQL	IQL	BCQ	BEAR	O-RL	BC	DD	D-QL	DT	StAR	GDT	QT	<b>RVDT</b>
kitchen-Comp.	43.8	62.5	8.1	0.0	2.0	65.0	65.0	84.0	50.8	40.8	43.8	81.7	<b>84.5 ± 2.3</b>
kitchen-partial	49.8	46.3	18.9	13.1	35.5	33.8	57.0	60.5	57.9	12.3	73.3	72.5	<b>75.0 ± 2.5</b>
Average	46.8	54.4	13.5	6.6	18.8	49.4	61.0	72.2	54.4	26.6	58.6	77.1	<b>79.8</b>
Maze2D Tasks	CQL	IQL	BCQ	BEAR	TD3+BC	BC	Diffuser	DD	DT	GDT	QDT	QT	<b>RVDT</b>
maze2d-u	94.7	42.1	49.1	65.7	14.8	88.9	113.9	116.2	31.0	50.4	57.3	99.2	<b>145.1 ± 3.8</b>
maze2d-m	41.8	34.9	17.1	25.0	62.1	38.3	121.5	122.3	8.2	7.8	13.3	168.8	<b>183.5 ± 4.5</b>
maze2d-l	49.6	61.7	30.8	81.0	88.6	1.5	123.0	125.9	2.3	0.7	31.0	242.7	<b>254.3 ± 4.6</b>
Average	62.0	46.2	32.3	57.2	55.2	42.9	119.5	121.5	13.8	19.6	33.9	170.2	<b>194.3</b>
AntMaze Tasks	CQL	IQL	BCQ	BEAR	TD3+BC	BC	DD	D-QL	DT	StAR	GDT	QT	<b>RVDT</b>
antmaze-u	74.0	87.5	78.9	73.0	78.6	54.6	73.1	93.4	59.2	51.3	76.0	96.0	<b>98.0 ± 4.0</b>
antmaze-u-d	84.0	62.2	55.0	61.0	71.4	45.6	49.2	66.2	53.0	45.6	69.0	92.0	<b>98.0 ± 4.0</b>
antmaze-m-d	53.7	70.0	0.0	8.0	3.0	0.0	24.6	<b>78.6</b>	0.0	0.0	6.0	24.0	30.0 ± 6.3
antmaze-l-d	14.9	47.5	2.2	0.0	0.0	0.0	7.5	<b>56.6</b>	0.0	0.0	0.0	10.0	10.0 ± 0.0
Average	56.6	66.8	34.0	35.5	38.2	25.0	38.6	<b>73.7</b>	28.0	24.2	37.8	57	59.0

**Gym MuJoCo tasks** are widely employed to evaluate continuous control capabilities under dense reward conditions. From Table 1, we observe that RVDT achieves new SOTA results across all datasets. Notably, scores exceeding 100 on D4RL MuJoCo tasks are already near the theoretical performance ceiling, where boundary effects make even small improvements significant. Moreover, algorithms employing return-coverage rebalancing mechanisms, such as RVDT, QT, CGDT, and TD3+BC, consistently demonstrate superior performance over traditional methods (e.g., BC and DT) across the majority of tasks, providing empirical support for our theoretical analysis.

**Adroit tasks** present substantial challenges for offline RL due to pronounced extrapolation errors stemming from the limited coverage of human demonstrations [16]. Consequently, these tasks critically test robustness against distributional shift and the ability to imitate from sparse expert data. Several observations emerge from our results: (i) except for RVDT and QT, other baselines fail to achieve satisfactory performance; (ii) RVDT and QT significantly outperform DT due to their utilization of rebalancing mechanisms; (iii) RVDT notably surpasses QT with 25% and 105% improvements on pen and hammer, respectively. These findings underscore RVDT’s effectiveness in learning from limited data and highlight the advantage gained from integrating explicit and implicit rebalancing mechanisms, effectively mitigating extrapolation errors and distributional shift problem.

**Kitchen tasks** primarily assess algorithms’ capabilities in multi-task manipulation and generalization to unseen states. Our results indicate that RVDT slightly outperforms current SOTA methods, demonstrating its competence in managing multi-task dependencies and generalizing effectively. Furthermore, RVDT substantially surpasses non-CSM approaches, reinforcing the superiority of CSM methods over DP-based methods, aligning with findings from prior works [11, 25].

**Maze tasks** evaluate the trajectory stitching capacity of CSM algorithms [16, 25]. By integrating expert KL regularization and Q-value guidance into a Transformer-based policy, RVDT substantially

Table 2: Performance comparison under varying proportions of expert trajectory augmentation. Each setting (e.g., 15% RVDT) corresponds to training on a mixed dataset composed of the base dataset (merged  $-\mathbf{m}$  and  $-\mathbf{m}-\mathbf{r}$ ) and top- $k\%$  expert trajectories sampled from the  $-\mathbf{m}-\mathbf{e}$  dataset.

Task	0% Expert			15% Expert			30% Expert			50% Expert		
	BC	QT	RVDT	BC	QT	RVDT	BC	QT	RVDT	BC	QT	RVDT
halfcheetah	42.4	46.9	<b>49.3</b>	46.0	79.5	<b>92.7</b>	73.3	91.3	<b>93.3</b>	71.1	92.8	<b>94.4</b>
hopper	65.4	95.8	<b>100.2</b>	99.0	108.0	<b>113.1</b>	112.3	110.6	<b>113.8</b>	112.8	108.3	<b>113.9</b>
walker2d	81.0	92.3	<b>92.7</b>	107.8	106.3	<b>110.3</b>	108.2	108.0	<b>112.4</b>	108.1	113.6	<b>113.8</b>
<b>Average</b>	62.9	78.3	<b>80.3</b>	84.3	97.9	<b>105.4</b>	97.9	103.4	<b>106.5</b>	97.3	104.9	<b>107.0</b>

Table 3: Performance comparison in low-data regimes. The learning difficulty increases progressively from  $\mathcal{D}_1$  to  $\mathcal{D}_4$ , with  $\mathcal{D}_1/\mathcal{D}_2$  containing 280 trajectories each and  $\mathcal{D}_3/\mathcal{D}_4$  containing 140 each.

Task (sparse R)	$\mathcal{D}_1$		$\mathcal{D}_2$		$\mathcal{D}_3$		$\mathcal{D}_4$	
	QT	RVDT	QT	RVDT	QT	RVDT	QT	RVDT
maze2d-umaze	100.3	<b>171.7</b>	81.8	<b>101.8</b>	73.1	<b>76.9</b>	61.4	<b>100.5</b>
maze2d-medium	137.1	<b>187.4</b>	175.2	<b>190.0</b>	163.2	<b>182.0</b>	98.3	<b>175.3</b>
maze2d-large	109.5	<b>140.4</b>	81.5	<b>90.1</b>	104.4	<b>131.3</b>	<b>100.2</b>	95.1
<b>Average</b>	115.6	<b>166.5</b>	112.8	<b>127.3</b>	113.6	<b>130.1</b>	86.6	<b>123.6</b>

enhances its ability to assemble optimal paths from fragmented sub-trajectories, thereby significantly outperforming all baselines. The AntMaze tasks, characterized by abundant suboptimal and unsuccessful trajectories, pose even greater challenges due to multi-action per state scenarios. While RVDT maintains competitive performance on smaller mazes (antmaze-u), virtually all CSM and value-regularized SL methods fail in larger, more complex mazes (antmaze-m, antmaze-l). In contrast, D-QL, leveraging expressive diffusion policies, achieves the best performance on these challenging tasks. This suggests that highly expressive policies are particularly effective in multi-action per state environments, highlighting the limitations in expressiveness of current CSM methods.

## 5.2 Performance under Imbalanced Trajectory Distributions

**Effectiveness of Expert Data Utilization.** We evaluate whether RVDT, integrating both explicit and implicit return rebalancing mechanisms, can more effectively leverage expert-level data from suboptimal datasets compared to QT and BC. We construct a base dataset by merging the  $-\mathbf{m}$  and  $-\mathbf{m}-\mathbf{r}$  datasets, incrementally augmenting it with top-performing trajectories from the  $-\mathbf{m}-\mathbf{e}$  dataset. Table 2 shows the performance of BC, QT, and RVDT at varying expert ratios. The results indicate: (i) RVDT consistently outperforms QT and BC at all expert ratios, highlighting the advantage of combining explicit and implicit rebalancing mechanisms compared to using only one (QT) or none (BC) of them; (ii) as the proportion of expert trajectories increases, RVDT and QT show steady improvements, while BC plateaus beyond approximately 30% expert data. This underscores the effectiveness of rebalancing mechanisms in enabling CSM policies to effectively utilize expert data.

**Performance in Low-Data Regimes.** To assess RVDT’s performance in low-data scenarios [46], we construct four datasets ( $\mathcal{D}_1$ - $\mathcal{D}_4$ ) by subsampling the original maze2d datasets. The learning difficulty increases progressively from  $\mathcal{D}_1$  to  $\mathcal{D}_4$ , with  $\mathcal{D}_1$  and  $\mathcal{D}_2$  containing 280 trajectories each, and  $\mathcal{D}_3$  and  $\mathcal{D}_4$  containing 140 trajectories. Within each pair,  $\mathcal{D}_2$  includes fewer expert-level trajectories than  $\mathcal{D}_1$ , as does  $\mathcal{D}_4$  compared to  $\mathcal{D}_3$ . Compared to the original 2000-trajectory maze2d dataset, these subsets better reflect algorithmic performance in small-sample conditions (details for  $\mathcal{D}_1$ - $\mathcal{D}_4$  are provided in Appendix F.1). Table 3 shows RVDT consistently outperforming QT, often by a large margin, with average improvements of 45.4%, 14.5%, 14.2%, and 45.6% across  $\mathcal{D}_1$ - $\mathcal{D}_4$ . This substantial performance gain highlights RVDT’s superior ability to utilize useful information from limited and low-quality datasets (detailed comparisons in Appendix F.1).

**Stitching Ability.** Maze2D is commonly used to evaluate the trajectory stitching capability of offline RL algorithms [25], as agents are required to compose successful trajectories by integrating fragmented experience. We test RVDT and several baselines across four Maze2D environments

Table 4: Performance comparison on Maze2D environments with sparse and dense rewards.

	Dataset	CQL	DT	QDT	QT	RVDT
Sparse R	maze2d-open-v0	216.7 $\pm$ 80.7	196.4 $\pm$ 39.6	190.1 $\pm$ 37.8	497.9 $\pm$ 12.3	<b>634.6 <math>\pm</math> 12.3</b>
	maze2d-umaze-v1	94.7 $\pm$ 23.1	31.0 $\pm$ 21.3	57.3 $\pm$ 8.2	105.4 $\pm$ 4.8	<b>145.1 <math>\pm</math> 3.8</b>
	maze2d-medium-v1	41.8 $\pm$ 13.6	8.2 $\pm$ 4.4	13.3 $\pm$ 5.6	172.0 $\pm$ 6.2	<b>183.5 <math>\pm</math> 4.5</b>
	maze2d-large-v1	49.6 $\pm$ 8.4	2.3 $\pm$ 0.9	31.0 $\pm$ 19.8	240.1 $\pm$ 2.5	<b>254.3 <math>\pm</math> 4.6</b>
Dense R	maze2d-open-v0	307.6 $\pm$ 43.5	346.2 $\pm$ 14.3	325.7 $\pm$ 61.4	608.4 $\pm$ 1.9	<b>663.9 <math>\pm</math> 15.9</b>
	maze2d-umaze-v1	72.7 $\pm$ 10.1	-6.8 $\pm$ 10.9	58.6 $\pm$ 3.3	<b>103.1 <math>\pm</math> 7.8</b>	99.5 $\pm$ 4.3
	maze2d-medium-v1	70.9 $\pm$ 9.2	31.5 $\pm$ 3.7	42.3 $\pm$ 7.1	111.9 $\pm$ 1.9	<b>126.9 <math>\pm</math> 8.7</b>
	maze2d-large-v1	90.9 $\pm$ 19.4	45.3 $\pm$ 11.2	62.2 $\pm$ 9.9	177.2 $\pm$ 7.8	<b>197.9 <math>\pm</math> 2.0</b>

Table 5: Component-level breakdown across ablation variants.

Component	DT	DT-Dup	RDT	QT	VDT	RVDT-Dup	RVDT-Determ	RVDT
Explicit Rebal.	None	Dup.	KL	None	None	Dup.	KL	KL
Implicit Rebal.	None	None	None	Q-value	Q-value	Q-value	Q-value	Q-value
Policy Type	Determ.	Stoch.	Stoch.	Determ.	Stoch.	Stoch.	Determ.	Stoch.

Table 6: Performance comparison of ablation variants across MuJoCo tasks.

Task	DT	DT-Dup	RDT	QT	VDT	RVDT-Dup	RVDT-Determ	RVDT
halfcheetah	84.2	90.3	90.5	91.2	89.5	93.4	91.5	<b>94.9</b>
hopper	109.5	112.1	111.9	112.3	112.6	112.1	113.6	<b>113.8</b>
walker2d	108.2	108.8	109.7	113.2	110.3	110.9	113.1	<b>118.7</b>
<b>Average</b>	100.6	103.7	104.0	105.6	104.1	105.5	106.1	<b>109.1</b>

with increasing complexity (open, umaze, medium, large) under both sparse and dense reward settings. Results in Table 4 show RVDT significantly outperforming baselines across almost all configurations. Notably, RVDT achieves significant improvements over QT under the sparse reward setting, where effective trajectory stitching is crucial. Results demonstrate that integrating explicit and implicit return rebalancing mechanisms substantially improves CSM-based policies' stitching capabilities, particularly for tasks requiring long-horizon credit assignment ability.

### 5.3 Ablation Studies

To quantify the contribution of each individual component in RVDT, we perform ablation studies on merged datasets (-m-e, -m-r, and -m) for MuJoCo continuous control tasks. These merged datasets alleviate coverage limitations in state, action, and return spaces, providing a clearer evaluation of component impacts without dataset-boundary biases [6, 68]. The ablation variants focus on three key aspects: *explicit rebalancing mechanisms*, *implicit rebalancing mechanisms*, and *policy types*. Explicit rebalancing mechanisms include expert-policy KL regularization and expert-level data duplication, while the implicit rebalancing mechanism corresponds to Q-value guidance. For policy types, we consider both Gaussian stochastic policies and deterministic policies. The ablation variants are summarized in Table 5. Specifically, the two major regularizers introduced in our framework, the policy regularization term and the value-based guidance term, correspond to the "Explicit Rebal." (Explicit Rebalancing Mechanism) and "Implicit Rebal." (Implicit Rebalancing Mechanism) abbreviated in Table 6, respectively. The variant RDT denotes the ablated version of RVDT without the value-based guidance component, while VDT represents the version without the policy regularization term. For the ablated variant employing deterministic policies (RVDT-Determ), the KL regularization term is replaced with an MSE objective. In variants where data duplication is applied, the top 50% of trajectories are directly duplicated and appended to the original dataset. Details are provided in Appendix C.

Table 6 summarizes the ablation results and reveals several key insights: (i) RVDT consistently achieves top performance across all tasks, demonstrating the synergistic effectiveness of integrating all proposed components. (ii) Both RDT (explicit rebalancing) and QT/VDT (implicit rebalancing) individually outperform DT, confirming the effectiveness of applying return-coverage rebalanc-

ing mechanisms on DT, and indicating that the combination of rebalancing mechanisms in RVDT provides additional performance gains. (iii) RVDT-Dup, which employs direct expert-level data duplication, achieves comparable or superior performance relative to VDT, supporting our theoretical insights regarding the benefits of explicit rebalancing mechanisms. (iv) A comparison between RVDT-Determ (deterministic policy) and RVDT (stochastic policy) shows that RVDT with stochastic policy outperforms its deterministic counterpart, highlighting the importance of policy expressiveness, particularly in multi-action per state scenarios.

## 6 Conclusion

This work establishes that both expert-level and full-spectrum return-coverage are critical determinants of the performance and sample efficiency of CSM policies in offline RL. Theoretical bounds that quantify the impact of these two aspects of return-coverage on policy optimality and sample complexity are derived, offering a principled understanding of the underlying mechanisms that drive the effectiveness of return rebalancing strategies in CSM-based methods. Moreover, we propose RVDT, the first CSM-based approach that elegantly integrates explicit and implicit return rebalancing mechanisms to effectively alleviate challenges posed by suboptimal datasets. Empirical evaluations on D4RL benchmarks demonstrate that RVDT enhances state-of-the-art performance in offline RL, validating the benefits and efficacy of return-coverage rebalancing.

## Acknowledgements

This work was supported in part by the National Natural Science Foundation of China: 62206248.

## References

- [1] Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *International conference on machine learning*, pages 22–31. PMLR, 2017.
- [2] Anurag Ajay, Yilun Du, Abhi Gupta, Joshua Tenenbaum, Tommi Jaakkola, and Pulkit Agrawal. Is conditional generative modeling all you need for decision-making? *arXiv preprint arXiv:2211.15657*, 2022.
- [3] Saurabh Arora and Prashant Doshi. A survey of inverse reinforcement learning: Challenges, methods and progress. *Artificial Intelligence*, 297:103500, 2021.
- [4] Wensong Bai, Chao Zhang, Yichao Fu, Peilin Zhao, and Hui Qian. Pacer: A fully push-forward-based distributional reinforcement learning algorithm. *Neurocomputing*, page 131301, 2025.
- [5] Marc G Bellemare, Will Dabney, and Rémi Munos. A distributional perspective on reinforcement learning. In *International conference on machine learning*, pages 449–458. PMLR, 2017.
- [6] 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.
- [7] David Brandfonbrener, Will Whitney, Rajesh Ranganath, and Joan Bruna. Offline rl without off-policy evaluation. *Advances in neural information processing systems*, 34:4933–4946, 2021.
- [8] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.
- [9] Yuanying Cai, Chuheng Zhang, Li Zhao, Wei Shen, Xuyun Zhang, Lei Song, Jiang Bian, Tao Qin, and Tieyan Liu. Td3 with reverse kl regularizer for offline reinforcement learning from mixed datasets. In *2022 IEEE International Conference on Data Mining (ICDM)*, pages 21–30. IEEE, 2022.

[10] Jiayu Chen, Bhargav Ganguly, Yang Xu, Yongsheng Mei, Tian Lan, and Vaneet Aggarwal. Deep generative models for offline policy learning: Tutorial, survey, and perspectives on future directions. *Transactions on Machine Learning Research*, 2024.

[11] 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.

[12] C Coleman, C Yeh, S Mussmann, B Mirzasoleiman, P Bailis, P Liang, J Leskovec, and M Zaharia. Selection via proxy: Efficient data selection for deep learning. In *International Conference on Learning Representations (ICLR)*, 2020.

[13] Will Dabney, Georg Ostrovski, David Silver, and Rémi Munos. Implicit quantile networks for distributional reinforcement learning. In *International conference on machine learning*, pages 1096–1105. PMLR, 2018.

[14] Scott Emmons, Benjamin Eysenbach, Ilya Kostrikov, and Sergey Levine. Rvs: What is essential for offline rl via supervised learning? In *10th International Conference on Learning Representations, ICLR 2022*, 2022.

[15] Rasool Fakoor, Jonas W Mueller, Kavosh Asadi, Pratik Chaudhari, and Alexander J Smola. Continuous doubly constrained batch reinforcement learning. *Advances in Neural Information Processing Systems*, 34:11260–11273, 2021.

[16] 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.

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

[18] Scott Fujimoto, Herke Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International conference on machine learning*, pages 1587–1596. PMLR, 2018.

[19] 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.

[20] Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. *Advances in neural information processing systems*, 29, 2016.

[21] Zhang-Wei Hong, Pulkit Agrawal, Remi Tachet des Combes, and Romain Laroche. Harnessing mixed offline reinforcement learning datasets via trajectory weighting. In *The Eleventh International Conference on Learning Representations*.

[22] Zhang-Wei Hong, Tao Chen, Yen-Chen Lin, Joni Pajarinen, and Pulkit Agrawal. Topological experience replay. In *International Conference on Learning Representations*. OpenReview.net, 2022.

[23] Zhang-Wei Hong, Aviral Kumar, Sathwik Karnik, Abhishek Bhandwaldar, Akash Srivastava, Joni Pajarinen, Romain Laroche, Abhishek Gupta, and Pulkit Agrawal. Beyond uniform sampling: Offline reinforcement learning with imbalanced datasets. *Advances in Neural Information Processing Systems*, 36:4985–5009, 2023.

[24] Dan Horgan, John Quan, David Budden, Gabriel Barth-Maron, Matteo Hessel, Hado van Hasselt, and David Silver. Distributed prioritized experience replay. In *International Conference on Learning Representations*, 2018.

[25] Shengchao Hu, Ziqing Fan, Chaoqin Huang, Li Shen, Ya Zhang, Yanfeng Wang, and Dacheng Tao. Q-value regularized transformer for offline reinforcement learning. In *International Conference on Machine Learning*, pages 19165–19181. PMLR, 2024.

[26] Shengchao Hu, Ziqing Fan, Li Shen, Ya Zhang, Yanfeng Wang, and Dacheng Tao. Harmodt: Harmony multi-task decision transformer for offline reinforcement learning. In *International Conference on Machine Learning*, pages 19182–19197. PMLR, 2024.

[27] Shengchao Hu, Li Shen, Ya Zhang, and Dacheng Tao. Learning multi-agent communication from graph modeling perspective. In *The Twelfth International Conference on Learning Representations*.

[28] Shengchao Hu, Li Shen, Ya Zhang, and Dacheng Tao. Graph decision transformer. *arXiv preprint arXiv:2303.03747*, 2023.

[29] Shengchao Hu, Li Shen, Ya Zhang, and Dacheng Tao. Prompt-tuning decision transformer with preference ranking. *arXiv preprint arXiv:2305.09648*, 2023.

[30] Shengchao Hu, Li Shen, Ya Zhang, and Dacheng Tao. Graph decision transformer for offline reinforcement learning. *SCIENCE CHINA-INFORMATION SCIENCES*, 68(6), 2025.

[31] Michael Janner, Yilun Du, Joshua Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. In *International Conference on Machine Learning*, pages 9902–9915. PMLR, 2022.

[32] Vishal Kaushal, Rishabh Iyer, Suraj Kothawade, Rohan Mahadev, Khoshrav Doctor, and Ganesh Ramakrishnan. Learning from less data: A unified data subset selection and active learning framework for computer vision. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1289–1299. IEEE, 2019.

[33] 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.

[34] Sungyoon Kim, Yunseon Choi, Daiki E Matsunaga, and Kee-Eung Kim. Stitching sub-trajectories with conditional diffusion model for goal-conditioned offline rl. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 13160–13167, 2024.

[35] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[36] Yilun Kong, Guozheng Ma, Qi Zhao, Haoyu Wang, Li Shen, Xueqian Wang, and Dacheng Tao. Mastering massive multi-task reinforcement learning via mixture-of-expert decision transformer. In *Forty-second International Conference on Machine Learning*.

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

[38] 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.

[39] Aviral Kumar, Joey Hong, Anikait Singh, and Sergey Levine. When should we prefer offline reinforcement learning over behavioral cloning? *arXiv preprint arXiv:2204.05618*, 2022.

[40] 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.

[41] Thibault Lahire, Matthieu Geist, and Emmanuel Rachelson. Large batch experience replay. In *International Conference on Machine Learning*, pages 11790–11813. PMLR, 2022.

[42] Guanghe Li, Yixiang Shan, Zhengbang Zhu, Ting Long, and Weinan Zhang. Diffstitch: Boosting offline reinforcement learning with diffusion-based trajectory stitching. In *International Conference on Machine Learning*, pages 28597–28609. PMLR, 2024.

[43] TP Lillicrap. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

[44] Cong Lu, Philip Ball, Yee Whye Teh, and Jack Parker-Holder. Synthetic experience replay. *Advances in Neural Information Processing Systems*, 36:46323–46344, 2023.

[45] Jiafei Lyu, Xiu Li, and Zongqing Lu. Double check your state before trusting it: Confidence-aware bidirectional offline model-based imagination. *Advances in Neural Information Processing Systems*, 35:38218–38231, 2022.

[46] Michael Moret, Lukas Friedrich, Francesca Grisoni, Daniel Merk, and Gisbert Schneider. Generative molecular design in low data regimes. *Nature Machine Intelligence*, 2(3):171–180, 2020.

[47] Fei Ni, Jianye Hao, Yao Mu, Yifu Yuan, Yan Zheng, Bin Wang, and Zhixuan Liang. Metadif-fuser: Diffusion model as conditional planner for offline meta-rl. In *International Conference on Machine Learning*, pages 26087–26105. PMLR, 2023.

[48] Mansheej Paul, Surya Ganguli, and Gintare Karolina Dziugaite. Deep learning on a data diet: Finding important examples early in training. *Advances in neural information processing systems*, 34:20596–20607, 2021.

[49] Doina Precup, Richard S Sutton, and Satinder Singh. Eligibility traces for off-policy policy evaluation. In *ICML*, volume 2000, pages 759–766. Citeseer, 2000.

[50] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

[51] Jinghuan Shang, Kumara Kahatapitiya, Xiang Li, and Michael S Ryoo. Starformer: Transformer with state-action-reward representations for visual reinforcement learning. In *European conference on computer vision*, pages 462–479. Springer, 2022.

[52] Yusuke Shinohara. A submodular optimization approach to sentence set selection. In *2014 IEEE International conference on acoustics, speech and signal processing (ICASSP)*, pages 4112–4115. IEEE, 2014.

[53] Richard S Sutton, Andrew G Barto, et al. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

[54] Mariya Toneva, Alessandro Sordoni, Remi Tachet des Combes, Adam Trischler, Yoshua Bengio, and Geoffrey J Gordon. An empirical study of example forgetting during deep neural network learning. In *International Conference on Learning Representations*, 2019.

[55] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

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

[57] Yuanfu Wang, Chao Yang, Ying Wen, Yu Liu, and Yu Qiao. Critic-guided decision transformer for offline reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 15706–15714, 2024.

[58] Zhendong Wang, Jonathan J Hunt, and Mingyuan Zhou. Diffusion policies as an expressive policy class for offline reinforcement learning. In *The Eleventh International Conference on Learning Representations*.

[59] Kai Wei, Rishabh Iyer, and Jeff Bilmes. Submodularity in data subset selection and active learning. In *International conference on machine learning*, pages 1954–1963. PMLR, 2015.

[60] Kai Wei, Yuzong Liu, Katrin Kirchhoff, and Jeff Bilmes. Unsupervised submodular subset selection for speech data. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4107–4111. IEEE, 2014.

[61] Yueh-Hua Wu, Xiaolong Wang, and Masashi Hamaya. Elastic decision transformer. *Advances in Neural Information Processing Systems*, 36, 2024.

- [62] 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.
- [63] Teng Yan, Zhendong Ruan, Yaobang Cai, Yu Han, Wenxian Li, and Yang Zhang. Q-value regularized decision conformer for offline reinforcement learning. *arXiv preprint arXiv:2409.08062*, 2024.
- [64] Junjie Zhang, Jiafei Lyu, Xiaoteng Ma, Jiangpeng Yan, Jun Yang, Le Wan, and Xiu Li. Uncertainty-driven trajectory truncation for data augmentation in offline reinforcement learning. In *ECAI 2023*, pages 3018–3025. IOS Press, 2023.
- [65] Hongling Zheng, Li Shen, Yong Luo, Tongliang Liu, Jialie Shen, and Dacheng Tao. Decomposed prompt decision transformer for efficient unseen task generalization. *Advances in Neural Information Processing Systems*, 37:122984–123006, 2024.
- [66] Hongling Zheng, Li Shen, Yong Luo, Deheng Ye, Bo Du, Jialie Shen, and Dacheng Tao. Decision mixer: Integrating long-term and local dependencies via dynamic token selection for decision-making. In *Forty-second International Conference on Machine Learning*.
- [67] Qinling Zheng, Amy Zhang, and Aditya Grover. Online decision transformer. In *international conference on machine learning*, pages 27042–27059. PMLR, 2022.
- [68] Zhaoyi Zhou, Chunling Zhu, Runlong Zhou, Qiwen Cui, Abhishek Gupta, and Simon Shaolei Du. Free from bellman completeness: Trajectory stitching via model-based return-conditioned supervised learning. In *NeurIPS 2023 Foundation Models for Decision Making Workshop*.
- [69] Zhengbang Zhu, Hanye Zhao, Haoran He, Yichao Zhong, Shenyu Zhang, Haoquan Guo, Tingting Chen, and Weinan Zhang. Diffusion models for reinforcement learning: A survey. *arXiv preprint arXiv:2311.01223*, 2023.

## NeurIPS Paper Checklist

### 1. Claims

Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

Answer: [\[Yes\]](#)

Justification: The abstract and introduction accurately reflect the paper's contributions and scope.

Guidelines:

- The answer NA means that the abstract and introduction do not include the claims made in the paper.
- The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
- The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

### 2. Limitations

Question: Does the paper discuss the limitations of the work performed by the authors?

Answer: [\[Yes\]](#)

Justification: The limitations of our work are discussed in experiments in section 5.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

### 3. Theory assumptions and proofs

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [\[Yes\]](#)

Justification: The complete set of assumptions is presented alongside the theoretical analysis in the main manuscript, and the corresponding proofs are provided in Appendix A.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

#### 4. Experimental result reproducibility

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [\[Yes\]](#)

Justification: The algorithmic details are thoroughly presented in the main manuscript, while the training and inference procedures are described in Appendix B. Comprehensive experimental settings and configurations are provided in Appendix C.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general, releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  - (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  - (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  - (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  - (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

## 5. Open access to data and code

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [\[Yes\]](#)

Justification: The paper utilizes open-source datasets, and the complete code of the algorithm is provided in the supplementary material, along with detailed instructions to faithfully reproduce the main experimental results.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so No is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

## 6. Experimental setting/details

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [\[Yes\]](#)

Justification: Section 5, along with Appendix B and Appendix C, provides a comprehensive description of the experimental settings and details, including all training and test configurations, data splits, hyperparameters, and optimizer selections.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

## 7. Experiment statistical significance

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [\[Yes\]](#)

Justification: The experimental results are presented with error bars that represent the standard error of the mean (SEM), as reported in the corresponding tables (e.g., Table 1).

Guidelines:

- The answer NA means that the paper does not include experiments.

- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

## 8. Experiments compute resources

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: **[Yes]**

Justification: The computational resources used for our experiments, including hardware specifications, memory usage, and execution time, are detailed in Appendix D.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

## 9. Code of ethics

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics <https://neurips.cc/public/EthicsGuidelines>?

Answer: **[Yes]**

Justification: The research conducted in this paper fully complies with the NeurIPS Code of Ethics. All datasets used are publicly available and legally obtained, with no privacy concerns. The experimental evaluations are designed to ensure fairness and unbiased comparisons across different methods. Additionally, the computational resources are managed responsibly, and the study does not pose any negative environmental or societal impacts.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

## 10. Broader impacts

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [NA]

Justification: This work primarily focuses on foundational research in machine learning, specifically in the development and evaluation of offline reinforcement learning algorithms. It does not involve applications or deployment scenarios that would directly impact societal, ethical, or privacy-related domains. Therefore, the study does not present any identifiable positive or negative societal impacts.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

## 11. Safeguards

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: This work focuses on offline reinforcement learning algorithms evaluated on publicly available datasets (e.g., D4RL benchmarks) that do not contain sensitive or scraped information. The proposed models are purely for experimental evaluation and do not have direct deployment scenarios that would pose risks for misuse or dual-use concerns.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

## 12. Licenses for existing assets

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: All datasets, codebases, and pre-trained models used in this paper are properly cited, and their licenses and terms of use are explicitly followed as specified by the original authors.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, [paperswithcode.com/datasets](http://paperswithcode.com/datasets) has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

### 13. New assets

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: This work introduces an implementation of the RVDT algorithm, which is open-sourced and well-documented. The documentation provides sufficient instructions for reproduction and experimental setups, ensuring clarity and ease of use.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

### 14. Crowdsourcing and research with human subjects

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: This paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

### 15. Institutional review board (IRB) approvals or equivalent for research with human subjects

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: This paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

#### 16. Declaration of LLM usage

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (<https://neurips.cc/Conferences/2025/LLM>) for what should or should not be described.

## A Proofs

### A.1 Proof of Theorem 1

The original proof is provided in [6]. For completeness, we restate the proof within our context to better align with the assumptions and notations adopted in this paper.

*Proof.* Let  $g(s_1, a_{1:H})$  be the value of the return by rolling out the open loop sequence of actions  $a_{1:H}$  under the deterministic dynamics induced by  $T$  and  $r$ . Then we can write

$$\begin{aligned} \mathbb{E}_{s_1}[f(s_1)] - J(\pi_f) &= \mathbb{E}_{s_1}[\mathbb{E}_{\pi_f|s_1}[f(s_1) - g_1]] \\ &= \mathbb{E}_{s_1}[\mathbb{E}_{a_{1:H} \sim \pi_f|s_1}[f(s_1) - g(s_1, a_{1:H})]] \\ &\quad + \mathbb{E}_{s_1}[\mathbb{E}_{a_{1:H} \sim \pi_f|s_1}[g(s_1, a_{1:H}) - g_1]] \\ &\leq \mathbb{E}_{s_1}[\mathbb{E}_{a_{1:H} \sim \pi_f|s_1}[f(s_1) - g(s_1, a_{1:H})]] + \epsilon H^2 \end{aligned} \quad (8)$$

where the last step follows by bounding the magnitude of the difference between  $g_1$  and  $g(s_1, a_{1:H})$  by  $H$  and applying a union bound over the  $H$  steps in the trajectory (using the near determinism assumption), namely:

$$H \cdot \sup_{s_1} \bigcup_t P_{a_t \sim \pi_f|s_1}(r_t \neq r(s_t, a_t) \text{ or } s_{t+1} \neq T(s_t, a_t)) \leq \epsilon H^2$$

Now we consider the first term from (8). Again bounding the magnitude of the difference by  $H$  we get that

$$\mathbb{E}_{s_1}[\mathbb{E}_{a_{1:H} \sim \pi_f|s_1}[f(s_1) - g(s_1, a_{1:H})]] \leq \mathbb{E}_{s_1} \int_{a_{1:H}} P_{\pi_f}(a_{1:H} | s_1) \mathbb{I}[g(s_1, a_{1:H}) \neq f(s_1)] H \quad (9)$$

To simplify notation, let  $\bar{s}_t = T(s_1, a_{1:t-1})$  be the result of following the deterministic dynamics defined by  $T$  up until step  $t$ . Expanding it out, applying the near determinism, the consistency of  $f$ , the coverage assumption, canceling some terms, and then inducting we see that:

$$\begin{aligned} P_{\pi_f}(a_{1:H}|s_1) &= \pi_f(a_1|s_1) \int_{s_2} P(s_2|s_1, a_1) P_{\pi_f}(a_{2:H}|s_1, s_2) \\ &\leq \pi_f(a_1|s_1) P_{\pi_f}(a_{2:H}|s_1, \bar{s}_2) + \epsilon \\ &= \pi_\beta(a_1|s_1) \frac{P_{\pi_\beta}(g_1 = f(s_1)|s_1, a_1)}{P_{\pi_\beta}(g_1 = f(s_1)|s_1)} P_{\pi_f}(a_{2:H}|s_1, \bar{s}_2) + \epsilon \\ &\leq \pi_\beta(a_1|s_1) \frac{\epsilon + P_{\pi_\beta}(g_1 - r(s_1, a_1) = f(s_1) - r(s_1, a_1)|s_1, a_1, \bar{s}_2)}{P_{\pi_\beta}(g_1 = f(s_1)|s_1)} P_{\pi_f}(a_{2:H}|s_1, \bar{s}_2) + \epsilon \\ &= \pi_\beta(a_1|s_1) \frac{\epsilon + P_{\pi_\beta}(g_2 = f(\bar{s}_2)|\bar{s}_2)}{P_{\pi_\beta}(g_1 = f(s_1)|s_1)} P_{\pi_f}(a_{2:H}|s_1, \bar{s}_2) + \epsilon \\ &\leq \pi_\beta(a_1|s_1) \frac{P_{\pi_\beta}(g_2 = f(\bar{s}_2)|\bar{s}_2)}{P_{\pi_\beta}(g_1 = f(s_1)|s_1)} P_{\pi_f}(a_{2:H}|s_1, \bar{s}_2) + \epsilon \left( \frac{1}{\alpha_f} + 1 \right) \\ &\leq \pi_\beta(a_1|s_1) \pi_\beta(a_2|\bar{s}_2) \frac{P_{\pi_\beta}(g_2 = f(\bar{s}_2)|\bar{s}_2)}{P_{\pi_\beta}(g_1 = f(s_1)|s_1)} \cdot \frac{P_{\pi_\beta}(g_3 = f(\bar{s}_3)|\bar{s}_3, a_3)}{P_{\pi_\beta}(g_2 = f(\bar{s}_2)|\bar{s}_2)} P_{\pi_f}(a_{3:H}|s_1, \bar{s}_3) \\ &\quad + 2\epsilon \left( \frac{1}{\alpha_f} + 1 \right) \\ &\leq \prod_{t=1}^H \pi_\beta(a_t|\bar{s}_t) \frac{P_{\pi_\beta}(g_H = f(\bar{s}_H)|\bar{s}_H, a_H)}{P_{\pi_\beta}(g_1 = f(s_1)|s_1)} + H\epsilon \left( \frac{1}{\alpha_f} + 1 \right) \\ &= \prod_{t=1}^H \pi_\beta(a_t|\bar{s}_t) \frac{\mathbb{I}[g(s_1, a_{1:H}) = f(s_1)]}{P_{\pi_\beta}(g_1 = f(s_1)|s_1)} + H\epsilon \left( \frac{1}{\alpha_f} + 1 \right) \end{aligned}$$

where the last step follows from the determinism of the trajectory that determines  $\bar{s}_H$  and the consistency of  $f$ . Plugging this back into (9) and noticing that the two indicator functions can never both be 1, we get that:

$$\mathbb{E}_{s_1} [\mathbb{E}_{a_{1:H} \sim \pi_f | s_1} [f(s_1) - g(s_1, a_{1:H})]] \leq H^2 \epsilon \left( \frac{1}{\alpha_f} + 1 \right)$$

Plugging this back into (8) yields the result.  $\square$

## A.2 Proof of Theorem 2 (performance gap)

We first give the following Lemmas to help the proof of Theorem 2.

**Lemma 1** (Brandfonbrener et al. [6]). *Consider the assumptions of Theorem 1. There exists an optimal conditioning function  $f^*$ , defined such that  $f^*(s_1)$  corresponds to the return of the deterministic optimal policy  $\pi^*$  for the underlying MDP. Assume optimal return-coverage  $P_{\pi^*}(g = f^*(s_1) | s_1) \geq \alpha_f^*$  holds for all initial states  $s_1$ . Then we have that*

$$J(\pi^*) - J(\pi_{f^*}^{CSM}) \leq \left( \frac{1}{\alpha_{f^*}} + 3 \right) H^2 \epsilon.$$

*Proof.* Since  $f^*(s_1)$  corresponds to the return of the deterministic optimal policy  $\pi^*$  for the underlying MDP, so that  $\mathbb{E}[f^*(s_1)]$  is approximately  $J(\pi^*)$ .

Let  $T^{\pi^*}(s_1, t)$  represent the state reached by running  $\pi^*$  from  $s_1$  for  $t$  steps under the deterministic dynamics defined by  $T$ . Then:

$$f^*(s_1) = \sum_{t=1}^H r(T^{\pi^*}(s_1, t), \pi^*(T^{\pi^*}(s_1, t)))$$

Now we have as in the proof of Theorem 1 that the probability that  $g \neq f^*(s)$  is bounded by  $\epsilon H$ , so that

$$\mathbb{E}_{s_1} [f^*(s_1)] - J(\pi^*) = \mathbb{E}_{s_1} [\mathbb{E}_{g \sim \pi^* | s_1} [f^*(s_1) - g]] \leq \mathbb{E}_{s_1} [P_{\pi^*}(g \neq f^*(s_1) | s_1) \cdot H] \leq \epsilon H^2$$

Combining this with Theorem 1 yields the result.  $\square$

**Lemma 2** (Achiam et al. [1]). *Let  $d_\pi$  refer to the marginal distribution of  $P_\pi$  over states only. For any two policies  $\pi, \pi'$  we have:*

$$\|d_\pi - d_{\pi'}\|_1 \leq 2H \cdot \mathbb{E}_{s \sim d_\pi} [TV(\pi(\cdot | s) \| \hat{\pi}'(\cdot | s))]$$

*Proof.* First we will define a few useful objects. Let  $d_\pi^h(s) = P_\pi(s_h = s)$ . Let  $\Delta_h = \|d_\pi^h(s) - d_{\pi'}^h(s)\|_1$ . Let  $\delta_h = 2\mathbb{E}_{s \sim d_\pi^h} [TV(\pi(\cdot | s) \| \hat{\pi}'(\cdot | s))]$ .

Now we claim that  $\Delta_h \leq \delta_{h-1} + \Delta_{h-1}$  for  $h > 1$  and  $\Delta_1 = 0$ .

To see this, consider some fixed  $h$ . Note that  $d_\pi^h(s) = \int_{s'} d_\pi^{h-1}(s') \int_{a'} \pi(a' | s') P(s | s', a')$ . Then expanding the definitions and adding and subtracting we see that

$$\begin{aligned} \Delta_h &= \int_s |d_\pi^h(s) - d_{\pi'}^h(s)| \\ &\leq \int_s \left| \int_{s'} d_\pi^{h-1}(s') \int_{a'} (\pi(a' | s') - \pi'(a' | s')) P(s | s', a') \right| \\ &\quad + \int_s \left| \int_{s'} (d_\pi^{h-1}(s') - d_{\pi'}^{h-1}(s')) \int_{a'} \pi'(a' | s') P(s | s', a') \right| \\ &\leq 2\mathbb{E}_{s \sim d_\pi^{h-1}} [TV(\pi(\cdot | s) \| \hat{\pi}'(\cdot | s))] + \|d_\pi^{h-1} - d_{\pi'}^{h-1}\|_1 = \delta_{h-1} + \Delta_{h-1} \end{aligned}$$

Now applying the claim and the definition of  $d_\pi$  we get that

$$\|d_\pi - d_{\pi'}\|_1 \leq \frac{1}{H} \sum_{h=1}^H \Delta_h \leq \frac{1}{H} \sum_{h=1}^H \sum_{j=1}^{h-1} \delta_j \leq H \frac{1}{H} \sum_{h=1}^H \delta_h = 2H \cdot \mathbb{E}_{s \sim d_\pi} [TV(\pi(\cdot | s) \| \hat{\pi}'(\cdot | s))]$$

□

Now we are ready to **prove Theorem 2**.

*Proof.* Let  $f^*(s_1)$  denote the return of the optimal deterministic policy  $\pi^*$  under deterministic dynamics. According to Theorem 1 and Lemma 1, we have:

$$J(\pi^*) - J(\pi_{f^*}^{\text{CSM}}) \leq \epsilon \left( \frac{1}{\alpha_{f^*}} + 3 \right) H^2. \quad (10)$$

$$|J(\pi_f^{\text{CSM}}) - J(\pi_{f^*}^{\text{CSM}})| \leq |J(\pi_f^{\text{CSM}}) - J(\pi_\beta)| + |J(\pi_{f^*}^{\text{CSM}}) - J(\pi_\beta)|. \quad (11)$$

Denote  $\pi_{f'}^{\text{CSM}}$  with an arbitrary conditioning function  $f'$  by  $\pi_{f'}$ . Let  $P_{\pi_{f'}}$  and  $P_{\pi_\beta}$  denote the stationary distribution over states, actions, and returns induced by policy  $\pi_{f'}$  and  $\pi_\beta$  respectively.

$$\begin{aligned} |J(\pi_{f'}) - J(\pi_\beta)| &= H \left| \left( \mathbb{E}_{P_{\pi_{f'}}} [r(s, a)] - \mathbb{E}_{P_{\pi_\beta}} [r(s, a)] \right) \right| \\ &\leq H \|d_{\pi_{f'}} - d_{\pi_\beta}\|_1 \\ &\leq 2H^2 \cdot \mathbb{E}_{s \sim d_{\pi_\beta}} [TV(\pi_{f'}(\cdot | s) \| \pi_\beta(\cdot | s))] \\ &\leq H^2 \cdot \mathbb{E}_{s \sim d_{\pi_\beta}} \left[ \int_a |\pi_{f'}(a | s) - \pi_\beta(a | s)| \right] \\ &\leq H^2 \mathbb{E}_{s \sim d_{\pi_\beta}} \left[ \int_a \pi_\beta(a | s) \frac{P_{\pi_\beta}(f'(s) | s) - P_{\pi_\beta}(f'(s) | s, a)}{P_{\pi_\beta}(f'(s) | s)} \right] \\ &\leq \frac{H^2}{\alpha_{f'}} \mathbb{E}_{s \sim d_{\pi_\beta}} \left[ \int_a \pi_\beta(a | s) \right], \end{aligned} \quad (12)$$

where the third inequality above utilizes the result of Lemma 2.

Denote  $\mathbb{E}_{s \sim d_{\pi_\beta}} [\int_a \pi_\beta(a | s)]$  by  $\bar{\alpha}_\beta$ . By equation (12) and equation (11) we have

$$|J(\pi_f^{\text{CSM}}) - J(\pi_{f^*}^{\text{CSM}})| \leq \left( \frac{1}{\alpha_f} + \frac{1}{\alpha_f^*} \right) H^2 \bar{\alpha}_\beta \quad (13)$$

Combine (10) and (13) we conclude

$$J(\pi^*) - J(\pi_f^{\text{CSM}}) \leq \epsilon \left( \frac{1}{\alpha_f^*} + 3 \right) H^2 + \left( \frac{1}{\alpha_f} + \frac{1}{\alpha_f^*} \right) H^2 \bar{\alpha}_\beta \quad (14)$$

Where we have the condition

$$\alpha_f + \alpha_{f^*} \leq 1, \alpha_f \in (0, 1]; \quad (15)$$

$$\text{if } f \neq f^*, \alpha_f^* \in (0, 1]. \quad (16)$$

That is  $\beta$  should have both good coverage on sample  $f$  and the optimal  $f^*$ , when expert data increase,  $f^*$  will increase but  $f$  may decrease.

Should balance expert data and full-spectrum data!

(17)

□

### A.3 Proof of Theorem 3 (sample complexity)

We first introduce the following useful lemma, whose proof synthesizes the techniques from Achiam et al.[1] and Brandfonbrener et al.[6].

**Lemma 3** (Sample complexity of CSM). *Consider any conditioning function  $f : \mathcal{S} \rightarrow \mathbb{R}$  such that the following assumptions hold: (i) Bounded occupancy mismatch:  $P_{\pi_f^{CSM}}(s) \leq C_f P_{\pi_\beta}(s)$  for all  $s$ . (ii) Return-coverage:  $P_{\pi_\beta}(g = f(s) \mid s) \geq \alpha_f$  for all  $s$ . (iii) The policy class  $\Pi$  is finite. (iv)  $|\log \pi(a \mid s, g) - \log \pi(a' \mid s', g')| \leq c$  for any  $(a, s, g, a', s', g')$  and all  $\pi \in \Pi$ . (v) The approximation error of  $\Pi$  is bounded by  $\epsilon_{approx}$ , i.e.,  $\min_{\pi \in \Pi} L(\pi) \leq \epsilon_{approx}$ .*

Define the expected loss for a CSM algorithm as

$$L(\hat{\pi}) = \mathbb{E}_{s \sim P_{\pi_\beta}} \mathbb{E}_{g \sim P_{\pi_\beta}(\cdot \mid s)} [\text{KL}(P_{\pi_\beta}(\cdot \mid s, g) \parallel \hat{\pi}(\cdot \mid s, g))].$$

Then with probability at least  $1 - \delta$ ,

$$J(\pi_f^{CSM}) - J(\hat{\pi}_f) \leq O \left( \frac{C_f}{\alpha_f} H^2 \left( \sqrt{c} \left( \frac{\log |\Pi|/\delta}{N} \right)^{1/4} + \sqrt{\epsilon_{approx}} \right) \right). \quad (18)$$

*Proof.* Applying the definition of  $J$  and Lemma 2, we get

$$\begin{aligned} J(\pi_f^{CSM}) - J(\hat{\pi}_f) &= H \left( \mathbb{E}_{P_{\pi_f^{CSM}}} [r(s, a)] - \mathbb{E}_{P_{\hat{\pi}_f}} [r(s, a)] \right) \\ &\leq H \left\| d_{\pi_f^{CSM}} - d_{\hat{\pi}_f} \right\|_1 \\ &\leq 2 \cdot \mathbb{E}_{s \sim d_{\pi_f^{CSM}}} [TV(\pi_f^{CSM}(\cdot \mid s) \parallel \hat{\pi}_f(\cdot \mid s))] H^2 \end{aligned}$$

Expanding definitions, using the multiply and divide trick, and applying the assumptions:

$$\begin{aligned} &2 \cdot \mathbb{E}_{s \sim d_{\pi_f^{CSM}}} [TV(\pi_f^{CSM}(\cdot \mid s) \parallel \hat{\pi}_f(\cdot \mid s))] \\ &= \mathbb{E}_{s \sim d_{\pi_f^{CSM}}} \left[ \int_a |P_{\pi_\beta}(a \mid s, f(s)) - \hat{\pi}(a \mid s, f(s))| \right] \\ &= \mathbb{E}_{s \sim d_{\pi_f^{CSM}}} \left[ \frac{P_{\pi_\beta}(f(s) \mid s)}{P_{\pi_\beta}(f(s) \mid s)} \int_a |P_{\pi_\beta}(a \mid s, f(s)) - \hat{\pi}(a \mid s, f(s))| \right] \\ &\leq \frac{C_f}{\alpha_f} \mathbb{E}_{s \sim d_{\pi_\beta}} \left[ P_{\pi_\beta}(f(s) \mid s) \int_a |P_{\pi_\beta}(a \mid s, f(s)) - \hat{\pi}(a \mid s, f(s))| \right] \\ &\leq \frac{C_f}{\alpha_f} \mathbb{E}_{s \sim d_{\pi_\beta}} \left[ \int_g P_{\pi_\beta}(g \mid s) \int_a |P_{\pi_\beta}(a \mid s, g) - \hat{\pi}(a \mid s, g)| \right] \\ &= 2 \frac{C_f}{\alpha_f} \mathbb{E}_{s \sim d_{\pi_f^{CSM}}, g \sim P_{\pi_\beta} \mid s} [TV(P_{\pi_\beta}(\cdot \mid s, g) \parallel \hat{\pi}(\cdot \mid s, g))] \\ &\leq \frac{C_f}{\alpha_f} \sqrt{2L(\hat{\pi})}, \end{aligned}$$

where the last step comes from Pinsker's inequality.

Combining with the above bound on the difference in expected values, we can get:

$$J(\pi_f^{CSM}) - J(\hat{\pi}_f) \leq \frac{C_f}{\alpha_f} H^2 \sqrt{2L(\hat{\pi})}. \quad (19)$$

We now write  $L(\pi) = \bar{L}(\pi) - H_{\pi_\beta}$ , where  $H_{\pi_\beta} = -\mathbb{E}_{(s, a, g) \sim P_{\pi_\beta}} [\log P_{\pi_\beta}(a \mid s, g)]$  and  $\bar{L}(\pi) := -\mathbb{E}_{(s, a, g) \sim P_{\pi_\beta}} [\log \pi(a \mid s, g)]$  is the cross-entropy loss.

Denoting  $\pi^\dagger \in \arg \min_{\pi \in \Pi} L(\pi)$ , we have

$$L(\hat{\pi}) = L(\hat{\pi}) - L(\pi^\dagger) + L(\pi^\dagger) \leq \bar{L}(\hat{\pi}) - \bar{L}(\pi^\dagger) + \epsilon_{approx}.$$

Denoting  $\hat{L}$  the empirical cross-entropy loss that is minimized by  $\hat{\pi}$ , we may further decompose

$$\begin{aligned}\bar{L}(\hat{\pi}) - \bar{L}(\pi^\dagger) &= \bar{L}(\hat{\pi}) - \hat{L}(\hat{\pi}) + \hat{L}(\hat{\pi}) - \hat{L}(\pi^\dagger) + \hat{L}(\pi^\dagger) - \bar{L}(\pi^\dagger) \\ &\leq 2 \sup_{\pi \in \Pi} |\bar{L}(\pi) - \hat{L}(\pi)|\end{aligned}$$

Under the assumptions on bounded loss differences, we may bound this, e.g., using McDiarmid's inequality and a union bound on  $\Pi$  to obtain the final result in (18) [50, 6].  $\square$

Now we are ready to prove the theorem 3

*Proof.* By Lemma 3 we have the following result that

$$J(\pi_f^{CSM}) - J(\hat{\pi}_f) \leq O\left(\frac{C_f}{\alpha_f} H^2 \left(\sqrt{c} \left(\frac{\log |\Pi|/\delta}{N}\right)^{1/4} + \sqrt{\epsilon_{\text{approx}}}\right)\right). \quad (20)$$

By Theorem 2 we have the following result that

$$J(\pi^*) - J(\pi_f^{CSM}) \leq \left(\frac{1}{\alpha_f^*} + 3\right) H^2 \epsilon + \left(\frac{1}{\alpha_f} + \frac{1}{\alpha_f^*}\right) H^2 C. \quad (21)$$

Thus, by combining the results from (20) and (21), we conclude that

$$J(\pi^*) - J(\hat{\pi}_f) \leq O\left(\frac{C_f}{\alpha_f} H^2 \left(\sqrt{c} \left(\frac{\log |\Pi|/\delta}{N}\right)^{1/4} + \sqrt{\epsilon_{\text{approx}}}\right) + \left(\frac{\epsilon + C}{\alpha_f^*} + \frac{C}{\alpha_f}\right) H^2\right).$$

$\square$

#### A.4 Proof of Proposition 1

*Proof.* To prove Proposition 1, we show that the KL regularization term in (4) is proportional to the negative log-likelihood (NLL) loss, under the assumption that the policy  $\pi_\theta$  is a factorized Gaussian distribution with a fixed isotropic covariance. The equivalence then implies that KL regularization explicitly reweights the NLL loss, thereby implementing an explicit resampling mechanism.

Recall the KL divergence between two multivariate Gaussians  $\mathcal{N}(\mu_1, \Sigma_1)$  and  $\mathcal{N}(\mu_2, \Sigma_2)$  is given by:

$$D_{\text{KL}}(\mathcal{N}(\mu_1, \Sigma_1) \parallel \mathcal{N}(\mu_2, \Sigma_2)) = \frac{1}{2} \left[ \text{tr}(\Sigma_2^{-1} \Sigma_1) + (\mu_2 - \mu_1)^\top \Sigma_2^{-1} (\mu_2 - \mu_1) - d + \log \frac{|\Sigma_2|}{|\Sigma_1|} \right], \quad (22)$$

where  $d$  is the dimensionality of the action space.

In our setting, the policy is parameterized as  $\pi_\theta(a|s) = \mathcal{N}(\mu_\theta(s), \xi I)$  and the target is a Dirac approximation centered at the action  $a$ , i.e.,  $\mathcal{N}(a, \xi I)$ . Substituting into (22), we obtain:

$$D_{\text{KL}}(\mathcal{N}(\mu_\theta(s), \xi I) \parallel \mathcal{N}(a, \xi I)) = \frac{1}{2} \left[ \text{tr}(I) + (\mu_\theta(s) - a)^\top (\xi I)^{-1} (\mu_\theta(s) - a) - d + \log 1 \right],$$

where we use the fact that  $\Sigma_1 = \Sigma_2 = \xi I$ . Simplifying yields:

$$D_{\text{KL}}(\mathcal{N}(\mu_\theta(s), \xi I) \parallel \mathcal{N}(a, \xi I)) = \frac{1}{2\xi} \|\mu_\theta(s) - a\|^2.$$

Taking expectation over  $(s, a) \sim \mathcal{D}$  gives:

$$\mathbb{E}_{(s, a) \sim \mathcal{D}} [D_{\text{KL}}(\mathcal{N}(\mu_\theta(s), \xi I) \parallel \mathcal{N}(a, \xi I))] = \frac{1}{2\xi} \mathbb{E}_{(s, a) \sim \mathcal{D}} [\|\mu_\theta(s) - a\|^2],$$

which shows that the KL divergence is proportional to the mean squared error (MSE) loss between  $\mu_\theta(s)$  and the ground-truth action  $a$ .

The log-likelihood of the Gaussian policy  $\pi_\theta(a|s) = \mathcal{N}(a; \mu_\theta(s), \xi I)$  is:

$$\log \pi_\theta(a|s) = -\frac{d}{2} \log(2\pi\xi) - \frac{1}{2\xi} \|a - \mu_\theta(s)\|^2.$$

Taking the negative log-likelihood gives:

$$-\log \pi_\theta(a|s) = \frac{1}{2\xi} \|\mu_\theta(s) - a\|^2 + \frac{d}{2} \log(2\pi\xi).$$

The constant term  $\frac{d}{2} \log(2\pi\xi)$  is independent of the policy parameters  $\theta$  and does not affect optimization. Therefore, minimizing the KL divergence is equivalent (up to scaling and additive constants) to minimizing the NLL loss:

$$\mathbb{E}_{(s,a) \sim \mathcal{D}} [D_{\text{KL}}(\mathcal{N}(\mu_\theta(s), \xi I) \| \mathcal{N}(a, \xi I))] = \mathbb{E}_{(s,a) \sim \mathcal{D}} [-\log \pi_\theta(a|s)] + \text{constant.} \quad (23)$$

Recall that the RDT objective in (4) augments the standard NLL loss with a KL regularization term over expert trajectories:

$$\mathcal{L}_{\text{RDT}}(\theta) = \mathbb{E}_{\tau \sim \mathcal{D}} \sum_{i=1}^H \left[ -\log \pi_\theta(a_i|s_i, g(\tau_i), \bar{\tau}_{t-1}^K) \right] + \alpha \mathbb{E}_{\tau \sim \mathcal{D}_e} \sum_{i=1}^H \left[ D_{\text{KL}}(\pi_\theta(\cdot|s_i, g(\tau_i), \bar{\tau}_{t-1}^K) \| \pi_e(\cdot|s_i)) \right].$$

If  $\pi_e$  is a delta function centered at  $a_i$  (i.e., using the empirical action from the expert trajectory), and both policies are Gaussians with identical covariance  $\xi I$ , the KL term simplifies to an MSE loss, and hence to an NLL term by (23). Thus, the second expectation term becomes equivalent to an additional NLL loss over  $\mathcal{D}_e$  with weighting factor  $\alpha$ .

This yields the final objective:

$$\mathcal{L}_{\text{RDT}}(\theta) = \mathbb{E}_{\tau \sim \mathcal{D}} \left[ (1 + \alpha \cdot \mathbb{I}[\tau \in \mathcal{D}_e]) \cdot \sum_{i=1}^H -\log \pi_\theta(a_i|s_i, g(\tau_i), \bar{\tau}_{t-1}^K) \right] + \text{constant.}$$

Since the constant term does not affect the optimization process, the proof is thus completed.  $\square$

## A.5 Formal Statement and Proof of Proposition 2

*Proof.* To formally state Proposition 2, we first define the loss functions of the respective algorithms.

We begin by explicitly stating our key assumptions:

- Infinite dataset size, i.e.,  $|D| \rightarrow \infty$
- Both  $\pi$  and  $\pi_\beta$  are gaussian policy

Under these conditions, it has been shown that minimizing the negative log-likelihood is equivalent to minimizing the KL divergence between  $\pi$  and  $\pi_\beta$ :

$$\begin{aligned} \text{NLL} &= \mathbb{E}_{(s,a) \sim D} [-\log \pi(a|s)] \\ &= \mathbb{E}_{s \sim d_{\pi_\beta}(\cdot)} [D_{\text{KL}}(\pi(\cdot|s) \| \pi_\beta(\cdot|s))] \end{aligned} \quad (24)$$

As shown in Proposition 1, KL regularization corresponds to a reweighting of the data distribution. We define the reweighted dataset  $\tilde{D}$  and the corresponding mixture policy  $\pi_{\tilde{\beta}}$  as:

$$\begin{aligned} \tilde{D} &= \frac{1}{1+\alpha} D + \frac{\alpha}{1+\alpha} D_e \\ \pi_{\tilde{\beta}} &= \frac{1}{1+\alpha} \pi_\beta + \frac{\alpha}{1+\alpha} \pi_e \end{aligned} \quad (25)$$

With these definitions, we rewrite the learning objectives for DT, QT, RDT and RVDT:

$$\pi_{\text{dt}}^* := \arg \max_{\pi} L_{\text{DT}}(\pi) = \arg \max_{\pi} \mathbb{E}_{s \sim d_{\pi_\beta}(\cdot)} [-D_{\text{KL}}(\pi(\cdot|s) \| \pi_\beta(\cdot|s))] = \pi_\beta \quad (26)$$

$$\pi_{\text{qt}}^* := \arg \max_{\pi} L_{\text{QT}}(\pi) = \arg \max_{\pi} \mathbb{E}_{s \sim d_{\pi_{\beta}}(\cdot)} [-D_{KL}(\pi(\cdot|s) \|\pi_{\beta}(\cdot|s)) + \eta \mathbb{E}_{a \sim \pi(\cdot|s)} [Q_{\pi}(s, a)]] \quad (27)$$

$$\pi_{\omega}^* := \arg \max_{\pi} L_{\text{RDT}}(\pi) = \arg \max_{\pi} \mathbb{E}_{s \sim d_{\pi_{\beta}}(\cdot)} [-D_{KL}(\pi(\cdot|s) \|\pi_{\tilde{\beta}}(\cdot|s))] = \pi_{\tilde{\beta}} \quad (28)$$

$$\pi_e^* := \arg \max_{\pi} L_{\text{RVDT}}(\pi) = \arg \max_{\pi} \mathbb{E}_{s \sim d_{\pi_{\beta}}(\cdot)} [-D_{KL}(\pi(\cdot|s) \|\pi_{\tilde{\beta}}(\cdot|s)) + \eta \mathbb{E}_{a \sim \pi(\cdot|s)} [Q_{\pi}(s, a)]] \quad (29)$$

Based on these optimization objectives, we can formally restate Proposition 2.

**Proposition 3** (Formal statement of Proposition 2). *Let  $\pi_e^*$ ,  $\pi_{\omega}^*$ ,  $\pi_{\text{qt}}^*$ , and  $\pi_{\text{dt}}^*$  denote the optimal policies learned from the objectives of RVDT (29), RDT (28), QT (27) and DT (26) respectively. Given the behavior policy  $\pi_{\beta}$ , there exists an expert-level subset  $\mathcal{D}_e \subseteq \mathcal{D}$  and an appropriate coefficient  $\alpha \geq 0$  such that the following inequalities hold for all states  $s \sim d_{\pi_{\beta}}$ :*

$$\begin{aligned} \mathbb{E}_s V^{\pi_e^*}(s) &\geq \mathbb{E}_s V^{\pi_{\omega}^*}(s) \geq \mathbb{E}_s V^{\pi_{\text{dt}}^*}(s), \quad \text{and} \\ \mathbb{E}_s V^{\pi_e^*}(s) &\geq \mathbb{E}_s V^{\pi_{\text{qt}}^*}(s) \geq \mathbb{E}_s V^{\pi_{\tilde{\beta}}}(s). \end{aligned} \quad (30)$$

Since  $\pi_{\text{dt}}^* = \pi_{\beta}$ , we aim to prove the following policy improvement results:

$$\mathbb{E}_{s \sim d_{\pi_{\beta}}(\cdot)} V^{\pi_{\beta}}(s) \leq \mathbb{E}_{s \sim d_{\pi_{\beta}}(\cdot)} V^{\pi_{\omega}^*}(s) \quad (31)$$

$$\mathbb{E}_{s \sim d_{\pi_{\beta}}(\cdot)} V^{\pi_{\text{qt}}^*}(s) \leq \mathbb{E}_{s \sim d_{\pi_{\beta}}(\cdot)} V^{\pi_e^*}(s) \quad (32)$$

$$\mathbb{E}_{s \sim d_{\pi_{\beta}}(\cdot)} V^{\pi_{\beta}}(s) \leq \mathbb{E}_{s \sim d_{\pi_{\beta}}(\cdot)} V^{\pi_{\text{qt}}}(s) \quad (33)$$

$$\mathbb{E}_{s \sim d_{\pi_{\beta}}(\cdot)} V^{\pi_{\omega}^*}(s) \leq \mathbb{E}_{s \sim d_{\pi_{\beta}}(\cdot)} V^{\pi_e^*}(s) \quad (34)$$

Inequalities (31) and (32) follow directly since (1) and (6) are special cases of (4) and (37), respectively, when  $\alpha = 0$ .

We now prove (33).

$$\begin{aligned} &\eta \mathbb{E}_{s \sim d_{\pi_{\beta}}(\cdot)} [V^{\pi_{\text{qt}}}(s)] \\ &= \eta \mathbb{E}_{s \sim d_{\pi_{\beta}}(\cdot), a \sim \pi_{\text{qt}}^*(\cdot|s)} [Q^{\pi_{\text{qt}}^*}(s, a)] \\ &\geq \eta \mathbb{E}_{s \sim d_{\pi_{\beta}}(\cdot), a \sim \pi_{\text{qt}}^*(\cdot|s)} [Q^{\pi_{\text{qt}}^*}(s, a)] - \mathbb{E}_{s \sim d_{\pi_{\beta}}(\cdot)} [D_{KL}(\pi(\cdot|s) \|\pi_{\beta}(\cdot|s))] \\ &\geq \eta \mathbb{E}_{s \sim d_{\pi_{\beta}}(\cdot), a \sim \pi_{\beta}} [Q^{\pi_{\beta}}(s, a)] - \mathbb{E}_{s \sim d_{\pi_{\beta}}(\cdot)} [D_{KL}(\pi_{\beta}(\cdot|s) \|\pi_{\beta}(\cdot|s))] \\ &= \eta \mathbb{E}_{s \sim d_{\pi_{\beta}}(\cdot), a \sim \pi_{\beta}} [Q^{\pi_{\beta}}(s, a)] = \eta \mathbb{E}_{s \sim d_{\pi_{\beta}}(\cdot)} [V^{\pi_{\beta}}(s)] \end{aligned} \quad (35)$$

where the first inequality follows the non-negativity of KL divergence, the second inequality follows  $\pi_{\text{qt}}^*$  is the optimal policy of 27.

Similarly, for (34):

$$\begin{aligned} &\eta \mathbb{E}_{s \sim d_{\pi_{\beta}}(\cdot)} [V^{\pi_e^*}(s)] \\ &= \eta \mathbb{E}_{s \sim d_{\pi_{\beta}}(\cdot), a \sim \pi_e^*(\cdot|s)} [Q^{\pi_e^*}(s, a)] \\ &\geq \eta \mathbb{E}_{s \sim d_{\pi_{\beta}}(\cdot), a \sim \pi_e^*(\cdot|s)} [Q^{\pi_e^*}(s, a)] - \mathbb{E}_{s \sim d_{\pi_{\beta}}(\cdot)} [D_{KL}(\pi(\cdot|s) \|\pi_{\tilde{\beta}}(\cdot|s))] \\ &\geq \eta \mathbb{E}_{s \sim d_{\pi_{\beta}}(\cdot), a \sim \pi_{\tilde{\beta}}} [Q^{\pi_{\tilde{\beta}}}(s, a)] = \eta \mathbb{E}_{s \sim d_{\pi_{\beta}}(\cdot)} [V^{\pi_{\tilde{\beta}}}(s)] \end{aligned} \quad (36)$$

This completes the proof.  $\square$

## B Training and Inference

We instantiate the previously described formulation using a transformer-based architecture. The RVDT architecture extends the DT framework by explicitly accounting for stochastic policies, predicting the policy’s mean and log-variance through two separate fully connected layers. The comprehensive training procedure for RVDT is summarized in Algorithm 1. For notational simplicity, Algorithm 1 represents the policy input using the shorthand  $\pi_\theta(\cdot|s_{i-K:i}, a_{i-K:i-1}, g_{i-K:i})$ , which explicitly treats the current state–return pair  $(s_i, g_i)$  as part of the context window. When  $i - K < 0$ , the context window is truncated to the available prefix of the trajectory.

In scenarios without supplementary datasets or direct access to expert policies, we rely on cumulative returns to differentiate trajectory quality. Specifically, we select the top  $\rho\%$  trajectories ( $\rho$  is a hyperparameter), ranked by cumulative return, to form the expert dataset  $\mathcal{D}_e$ . The expert policy  $\pi^e$  is subsequently trained via behavior cloning on  $\mathcal{D}_e$ . Note that alternative imitation learning approaches such as GAIL [20] or IRL [3] could similarly be utilized to derive  $\pi^e$ .

RVDT employs a two-step sampling process to ensure uniform sampling of sub-trajectories of length  $K$  from the replay buffer  $\mathcal{T}_{\text{replay}}$ . Initially, an entire trajectory is sampled with probability proportional to its length, and subsequently, a sub-trajectory of length  $K$  is uniformly extracted. For environments characterized by non-negative dense rewards, this approach resembles an importance sampling scheme [67].

We leverage offline Dynamic Programming techniques to train the value network  $Q_\psi$ , optimizing via Temporal Difference (TD) learning [53]. In practical implementation, standard stabilization techniques such as double Q-learning [18] and the use of a target network [43] are adopted. It is noteworthy that in the offline RL setting, conservative Q-learning (CQL) [40] is typically employed to mitigate the out-of-distribution issue by maintaining policy similarity to the behavior policy. However, since the RVDT inherently incorporates behavior cloning, which naturally ensures similarity to the behavior policy, the additional adoption of CQL yields minimal performance gains.

During inference, return-conditioned supervised methods initially require specifying a target return as the first RTG token, subsequently subtracting sampled rewards to update the RTG token iteratively. According to Theorem 1, achieving the target return critically depends on the coverage of the specified return in the dataset. Hence, a practical choice is to utilize the maximum return observed in the training dataset as the initial target. Nonetheless, given RVDT’s integration with dynamic programming principles and its strong trajectory stitching capability, the actual optimal target return may surpass the dataset’s maximum observed return. Building upon the inference strategy of QT, we randomly sample candidate returns within a positive 10% margin above the maximum return observed in the dataset, thereby obtaining multiple candidate return-to-go tokens  $\hat{g}_0^0, \hat{g}_0^1, \dots, \hat{g}_0^m$ . Actions corresponding to each candidate return-to-go token are simultaneously generated. Subsequently, we employ the learned Q-value function to preferentially select actions associated with higher returns. This selection mechanism is formally expressed as:  $\hat{a}_t^i = \arg \max_{\hat{a}_t^i} Q_\psi(s_t, \hat{a}_t^i)$ , where  $\hat{a}_t^i \sim \pi(\cdot|s_t, g_t^i, \bar{\tau}_{t-1}^K)$ .

## C Experimental Details

### C.1 Benchmark Details

Our evaluation is performed across five diverse domains in the D4RL benchmarks [16]: **Gym**, **Adroit**, **Kitchen**, **Maze2D**, and **AntMaze**. These domains are designed to comprehensively evaluate the capability of offline RL algorithms in handling different types of dynamics, reward structures, and state-action distributions.

- **Gym-MuJoCo** locomotion tasks serve as standard benchmarks for continuous control. These tasks, such as HalfCheetah, Hopper, and Walker2d, are characterized by smooth reward functions and dense feedback signals. The datasets include trajectories generated by policies of varying quality: `random`, `medium`, `expert`, and `medium-expert`, representing different levels of optimality. While `expert` and `medium-expert` provide high-quality trajectories, `medium`, `random`, and `medium-replay` contain suboptimal and exploratory behavior, offering diverse state-action coverage for offline learning.

---

**Algorithm 1:** RVDT: Return-rebalanced Value-regularized Decision Transformer

---

**Input:** Sequence length  $K$ , dataset  $\mathcal{D}$ , hyperparameters  $\alpha, \eta$ , expert ratio  $\rho\%$   
**Initialize:** Policy  $\pi_\theta$ , critic  $Q_\psi$   
Construct expert dataset  $\mathcal{D}_e$  by selecting top  $\rho\%$  trajectories from  $\mathcal{D}$ ;  
Train expert policy  $\pi^e$  on  $\mathcal{D}_e$  via BC;  
**for**  $t = 1$  **to**  $T$  **do**  
    Sample a trajectory  $\tau \sim \mathcal{D}$  and a length- $K$  subsequence from  $\tau$ ;  
    Denote the sampled subsequence as  $\{(s_i, a_i, g_i)\}_{i=1}^K$  by re-indexing;  
    // Critic update  
    Update critic  $Q_\psi$  using offline TD learning on transitions sampled from  $\mathcal{D}$ ;  
    // Policy update  
    **for**  $i = 1$  **to**  $K$  **do**  
        Sample action  $\bar{a}_i \sim \pi_\theta(\cdot | s_{i-K:i}, a_{i-K:i-1}, g_{i-K:i})$ ;  
        Compute  $Q_\psi(s_i, \bar{a}_i)$ ;  
        Compute  $\text{KL}(\pi_\theta(\cdot | s_{i-K:i}, a_{i-K:i-1}, g_{i-K:i}) \parallel \pi^e(\cdot | s_i))$ ;  
    **Update** policy  $\pi_\theta$  by minimizing Eq. (7);  
**return**  $\pi_\theta$ ;

---

- **Adroit** tasks involve high-dimensional dexterous manipulation using a 24-DoF robotic hand. The datasets are primarily collected from *human demonstrations* and *behavior cloned policies*, resulting in narrow state-action coverage and strong multi-modal behavior. Due to the inherent complexity of the hand’s control space, policies must be effectively regularized to avoid out-of-distribution (OOD) actions. The sparse nature of successful demonstrations and the high variance in human trajectories further challenge offline RL algorithms in terms of generalization and stability.
- **Kitchen** environment introduces long-horizon, multi-task manipulation scenarios, where an agent is required to execute a fixed sequence of four subtasks to achieve a goal configuration, such as turning on the microwave, opening the cabinet, or switching on the light. This setting emphasizes the importance of *temporal abstraction* and *multi-task planning*. The datasets are collected from *demonstrator policies* that exhibit varying degrees of task completion, leading to fragmented sub-trajectories that need to be stitched together effectively during offline learning.
- **Maze2D** tasks are specifically designed to evaluate the ability of offline RL algorithms to *stitch together fragmented sub-trajectories* and recover globally optimal paths in a continuous navigation setting. Unlike Gym-MuJoCo, Maze2D environments present *sparse rewards*, requiring the agent to combine distant state transitions to solve navigation tasks effectively. Although Maze2D also provides a dense reward setting, we primarily focus on its sparse reward version throughout this work unless otherwise specified. The agent controls a 2D ball in complex maze-like structures, where the challenge lies in leveraging suboptimal paths to construct optimal global trajectories.
- **AntMaze** represents a more challenging extension of Maze2D by introducing an 8-DoF *Ant* robot instead of a simple 2D ball. The sparse reward structure remains, but the high-dimensional state space and unstable dynamics of the *Ant* make trajectory stitching significantly more difficult. Successful navigation requires effective trajectory stitching and long-horizon credit assignment to traverse disconnected regions of the maze. Datasets are collected using goal-conditioned policies with diverse start-goal configurations, introducing variability in trajectory quality and return distributions.

Dataset descriptions and policy types for representative tasks in the D4RL benchmarks is given in Table 7.

Table 7: Dataset descriptions and policy types for representative tasks in the D4RL benchmarks.

Task	Policy Type	Description
hopper-medium-expert-v2	Expert + Medium policy	A 50-50 mixture of trajectories from a near-optimal expert policy and a medium-performance policy.
hopper-medium-v2	Medium behavior policy	Collected by a policy trained to partial convergence, representing moderately optimal behavior.
hopper-medium-replay-v2	Medium + Experience replay	A replay buffer from the medium policy, including a wide range of suboptimal and early-stage transitions.
pen-human-v1	Human demonstrations	Trajectories collected via human teleoperation using a virtual reality interface.
pen-cloned-v1	Behavior cloning policy	Trajectories generated by a policy trained via behavioral cloning on human demonstration data.
kitchen-complete-v0	Expert policy	Expert demonstrations accomplishing a fixed sequence of four goal-directed tasks.
kitchen-partial-v0	Expert policy	Expert demonstrations completing a variable subset of kitchen tasks in different orders.
maze2d-umaze-v1	Trajectory tracking (suboptimal)	Random-walk trajectories collected in a small U-shaped maze.
maze2d-medium-v1	Trajectory tracking (suboptimal)	Random-walk trajectories in a medium-sized maze with increased spatial complexity.
maze2d-large-v1	Trajectory tracking (suboptimal)	Random-walk trajectories in the largest maze with long-horizon navigation.
antmaze-umaze-v0	Expert policy	Goal-conditioned expert demonstrations in a small U-shaped maze.
antmaze-umaze-diverse-v0	Expert policy (diverse goals)	Goal-reaching demonstrations in the small maze with diverse start-goal pairs.
antmaze-medium-diverse-v0	Expert policy (diverse goals)	Goal-directed expert trajectories in a medium maze with varying start-goal configurations.
antmaze-large-diverse-v0	Expert policy (sparse + diverse)	Sparse, long-horizon goal-reaching demonstrations in a large maze using an 8-DoF quadruped agent.

## D Implementation Details

**The policy network** of RVDT is implemented as a Decision Transformer, built upon the open-source minGPT codebase<sup>3</sup> and the official QT implementation<sup>4</sup>. Detailed model parameters are provided in Table 8.

**The expert policy**  $\pi^e$  is modeled using a fully connected neural network and trained via behavior cloning. The corresponding model parameters are listed in Table 9.

**The Q-networks** are represented by three-layer MLPs with Mish activations and 256 hidden units for each layer.

All networks are trained using the Adam optimizer [35].

The code of RVDT, including all hyperparameter configurations and experimental setups, is provided in the supplementary material for reproducibility.

The experiments were conducted on a server equipped with two AMD EPYC 7542 32-Core Processors and 8 NVIDIA GeForce RTX 4090 GPUs with 24 GB of memory. For each task, the average training time on a single GPU was approximately 20,000 seconds, with a memory consumption of around 3000MB. These computational settings ensure reproducibility and align with the reported performance metrics.

<sup>3</sup><https://github.com/karpathy/minGPT>

<sup>4</sup><https://github.com/charleshsc/QT>

Table 8: Hyperparameters of RVDT in our experiment.

Parameter	Value
Number of layers	4
Number of attention heads	4
Embedding dimension	256
Nonlinearity function	ReLU
Batch size	256
Context length $K$	20
Dropout	0.1
Learning rate	3.0e - 4
Weight decay	1.0e - 4

Table 9: Hyperparameters of  $\pi^e$  in our experiment.

Parameter	Value
Number of layers	4
Embedding dimension	256
Nonlinearity function	ReLU
Batch size	256
Dropout	0.1
Learning rate	1.0e - 4
Weight decay	1.0e - 4

## E Hyperparameters for RVDT

We recall the loss function for RVDT as follows:

$$\begin{aligned}
 \mathcal{L}_{\text{RVDT}}(\theta) = & \kappa \mathbb{E}_{\tau \sim \mathcal{D}} \left[ \sum_{i=1}^H -\log \pi_\theta(a_i | s_i, g(\tau_i), \bar{\tau}_{t-1}^K) \right] \\
 & + \alpha \mathbb{E}_{\tau \sim \mathcal{D}_e} \sum_{i=1}^H \text{KL} \left[ \pi_\theta(\cdot | s_i, g(\tau_i), \bar{\tau}_{t-1}^K) \parallel \pi^e(\cdot | s_i) \right] \\
 & - \frac{\bar{\eta}}{\mathbb{E}_{\tau \sim \mathcal{D}} \mathbb{E}_{(s_i, a_i) \sim \tau} |Q_\psi(s_i, a_i)|} \mathbb{E}_{\tau \sim \mathcal{D}} \mathbb{E}_{s_i \sim \tau, a_i \sim \pi_\theta} [Q_\psi(s_i, a_i)].
 \end{aligned} \tag{37}$$

(37) involves four key hyperparameters that may vary across different tasks:

- The weight of the NLL,  $\kappa$ ;
- The weight of the expert policy KL regularizer,  $\alpha$ ;
- The weight of the Q-value guidance term,  $\bar{\eta}$ ;
- The proportion of expert-level trajectories selected from the dataset  $\mathcal{D}$  to form the expert dataset  $\mathcal{D}_e$ , denoted as  $\rho\%$ .

We employ the *Tree-structured Parzen Estimator (TPE)* for hyperparameter optimization in each environment. The optimal values are determined based on performance on a held-out validation set, aiming to achieve the best trade-off between policy regularization and value maximization. The key hyperparameters of RVDT is given in Table 10. Other hyperparameter configurations used for reproducing the experiments are provided in the code available in the supplementary material.

Table 10: Key hyperparameter settings for all selected tasks to reproduce the main results reported in Table 1.

Tasks	$\alpha$	$\bar{\eta}$	$\kappa$	Expert dataset ratio $\rho\%$	Grad Norm
halfcheetah-medium-expert-v2	0.1	5.0	0.02	50%	15.0
hopper-medium-expert-v2	0.01	1.0	0.02	50%	9.0
walker2d-medium-expert-v2	0.01	2.0	5e-6	10%	5.0
halfcheetah-medium-v2	0.1	5.0	0.02	10%	15.0
hopper-medium-v2	1.0	1.0	0.02	50%	9.0
walker2d-medium-v2	0.1	2.0	0.02	10%	5.0
halfcheetah-medium-replay-v2	1.0	5.0	2e-4	50%	15.0
hopper-medium-replay-v2	1.0	3.0	0.02	50%	9.0
walker2d-medium-replay-v2	0.1	2.0	0.02	10%	5.0
maze2d-open-v0	1.0	0.01	0.02	10%	9.0
maze2d-umaze-v1	1.0	5.0	1.0	30%	20.0
maze2d-medium-v1	0.1	5.0	5e-6	30%	9.0
maze2d-large-v1	1.0	4.0	2e-6	30%	9.0
antmaze-umaze-v0	0.1	0.05	0.02	30%	9.0
antmaze-umaze-diverse-v0	0.1	0.02	0.01	70%	9.0
pen-human-v1	10.0	0.1	2e-3	50%	9.0
hammer-human-v1	1.0	1.0	0.02	50%	5.0
pen-cloned-v1	10.0	0.1	0.02	30%	9.0
hammer-cloned-v1	0.5	0.01	2e-4	50%	9.0
kitchen-complete-v0	0.1	0.001	1e-4	50%	9.0
kitchen-partial-v0	10.0	0.01	0.02	50%	9.0
maze2d-open-dense-v0	1.0	0.01	0.02	10%	9.0
maze2d-umaze-dense-v1	0.5	3.0	0.02	10%	5.0
maze2d-medium-dense-v1	100.0	5.0	2e-4	10%	9.0
maze2d-large-dense-v1	1.0	4.0	2e-3	10%	9.0
antmaze-medium-diverse-v0	0.1	0.01	0.02	30%	9.0
antmaze-large-diverse-v0	0.1	0.005	0.02	30%	9.0

## F Additional Experimental Results

### F.1 Performance in Low-data Regimes

We provide a detailed description of the four sub-datasets  $\mathcal{D}_1 - \mathcal{D}_4$  constructed for the low-data regime experiments. First, note that each of the original `maze2d-umaze`, `maze2d-medium`, and `maze2d-large` datasets consists of 2000 trajectories. For each task, we filter the original dataset to distinguish trajectories of different quality levels. We define three quality-based sub-datasets: expert, medium, and noise, denoted as  $\mathcal{D}_e$ ,  $\mathcal{D}_m$ , and  $\mathcal{D}_n$ , respectively. These subsets are obtained by randomly sampling 200 trajectories from the top 30%, middle 40%, and bottom 30% of returns in the original dataset.

The sub-datasets  $\mathcal{D}_1 - \mathcal{D}_4$  are then constructed by randomly sampling from  $\mathcal{D}_e$ ,  $\mathcal{D}_m$ , and  $\mathcal{D}_n$  according to the following proportions:

1.  $\mathcal{D}_1$ : 50%  $\mathcal{D}_e$  + 50%  $\mathcal{D}_m$  + 40%  $\mathcal{D}_n$ , comprising 280 trajectories, designed to achieve full return-coverage.
2.  $\mathcal{D}_2$ : 20%  $\mathcal{D}_e$  + 20%  $\mathcal{D}_m$  + 100%  $\mathcal{D}_n$ , comprising 280 trajectories, dominated by random-policy data with a smaller fraction of medium and expert trajectories.
3.  $\mathcal{D}_3$ : 25%  $\mathcal{D}_e$  + 25%  $\mathcal{D}_m$  + 20%  $\mathcal{D}_n$ , comprising 140 trajectories, mirroring the empirical return distribution of  $\mathcal{D}_1$  but with fewer samples.
4.  $\mathcal{D}_4$ : 10%  $\mathcal{D}_e$  + 10%  $\mathcal{D}_m$  + 50%  $\mathcal{D}_n$ , comprising 140 trajectories, mirroring the empirical return distribution of  $\mathcal{D}_2$  with a reduced sample size.

Table 3 demonstrates the significant performance improvements of RVDT over QT in low-data regimes. The percentage improvements of RVDT over QT, derived from Table 3, are summarized in Table 11. These results confirm that RVDT, by incorporating both explicit and implicit rebalancing mechanisms, substantially enhances its ability to recover optimal policies from limited data sam-

ples. This further highlights the non-trivial nature of RVDT’s pioneering integration of explicit and implicit rebalancing strategies.

Table 11: Percentage improvement of RVDT over QT across different datasets. For example, an  $71.2\% \uparrow$  indicates that RVDT outperforms QT by 71.2% on the `maze2d-umaze` environment when trained with the  $\mathcal{D}_1$  dataset.

Tasks (sparse R)	$\mathcal{D}_1$	$\mathcal{D}_2$	$\mathcal{D}_3$	$\mathcal{D}_4$
maze2d-umaze	$71.2\% \uparrow$	$24.4\% \uparrow$	$5.2\% \uparrow$	$63.7\% \uparrow$
maze2d-medium	$36.7\% \uparrow$	$8.4\% \uparrow$	$11.5\% \uparrow$	$78.3\% \uparrow$
maze2d-large	$28.2\% \uparrow$	$10.6\% \uparrow$	$25.8\% \uparrow$	$-5.1\% \downarrow$
<b>Average</b>	$45.4\% \uparrow$	$14.5\% \uparrow$	$14.2\% \uparrow$	$45.6\% \uparrow$

## F.2 Impact of Expert dataset ratio

We evaluate the performance of RVDT when trained with varying proportions of subsampled expert trajectories, i.e., by adjusting the expert dataset ratio  $\rho\%$ . Adjusting the proportion  $\rho\%$  directly affects the trade-off between  $\alpha_f^*$  and  $\alpha_f$ : as the proportion increases from 10% to 100%,  $\alpha_f^*$  first rises and then declines, while  $\alpha_f$  exhibits the opposite trend. This experiment aims to (1) examine how the interplay between  $\alpha_f^*$  and  $\alpha_f$  influences algorithmic performance, and (2) identify the optimal ratio of expert-level trajectories required for effective learning.

The results, summarized in Table 12, show that using approximately 30% of the top-performing trajectories consistently yields the best results across Maze2D tasks. Specifically, the 30% expert subset achieves higher scores than both smaller (10%) and larger (50%, 70%, and 100%) proportions, suggesting that  $\alpha_f^*$  and  $\alpha_f$  reach a near-optimal balance under this configuration.

This observation aligns closely with our theoretical analysis, which highlights the importance of maintaining a balance between expert-level and full-spectrum return-coverage. It reveals that effective rebalancing does not simply entail maximizing the proportion of high-quality expert trajectories. Intuitively, relying exclusively on expert data (i.e., maximizing  $\alpha_f^*$  while neglecting  $\alpha_f$ ) can cause significant performance degradation due to distributional shift, particularly when training data are limited. Therefore, careful calibration between expert-level and full-spectrum return-coverage is crucial for achieving optimal performance in offline RL.

Table 12: Performance comparison with different proportions of subsampled expert datasets. For example, 10% RVDT corresponds to the performance of RVDT when the expert dataset  $\mathcal{D}_e$  consists of the top 10% of trajectories ranked by return from the full offline dataset  $\mathcal{D}$ .

Task (sparse R)	10% RVDT	30% RVDT	50% RVDT	70% RVDT	100% RVDT
maze2d-umaze-v1	93.1	<b>143.0</b>	100.4	102.4	101.6
maze2d-medium-v1	189.9	<b>195.0</b>	192.5	191.6	188.6
maze2d-large-v1	246.5	<b>250.0</b>	248.5	248.1	243.6
<b>Average</b>	176.50	<b>196.0</b>	180.47	180.70	177.93

## F.3 Influence of Expert Data Quality

To empirically investigate how the quality of the expert policy affects RVDT performance, we evaluate RVDT under varying proportions of expert trajectory augmentation. As summarized in Table 13, each configuration (e.g., RVDT (15%)) is trained on a mixed dataset composed of the base dataset (merged `-m` and `-m-r`) and the top- $k\%$  expert trajectories sampled from the `-m-e` dataset, which can be regarded as expert-level trajectories.

The results show that RVDT performance consistently improves as the proportion of expert trajectories increases, with the most notable gain observed between RVDT (0%) and RVDT (15%). These findings indicate that higher-quality expert policies contribute to better performance. Combined with the results in Table 12, we can clearly observe the influence of adjusting and rebalancing return

coverage on algorithm performance, thereby further confirming the effectiveness of the proposed return-coverage rebalancing mechanism.

It is important to note that the expert policy used in RVDT for the main experiments (see Table 14) is not trained from any externally provided demonstrations. Instead, it is derived from a subset of the existing offline dataset, denoted as  $\mathcal{D}_e \subset \mathcal{D}$ , by selecting relatively high-return trajectories. Consequently, our method is fully applicable to standard offline RL settings without requiring any additional expert data. Moreover, the return coverage assumed in our theoretical analysis is not restricted to strictly optimal trajectories. In practice, the presence of near-expert or moderately high-return trajectories is often sufficient to ensure effective learning. This property underscores that RVDT does not rely on idealized expert-level datasets and remains robust as long as the dataset exhibits sufficient diversity to support relative return ranking. As demonstrated in Table 14, RVDT continues to achieve notable improvements even on the -medium and -medium-replay datasets, which contain no expert demonstrations, validating its practicality under realistic offline RL conditions.

Table 13: Performance comparison under varying proportions of expert trajectory augmentation. Each configuration (e.g., RVDT (15%)) is trained on a mixed dataset composed of the base dataset (merged -m and -m-r) and top- $k\%$  expert trajectories sampled from the -m-e dataset.

Task	RVDT (0%)	RVDT (15%)	RVDT (30%)	RVDT (50%)
halfcheetah	49.3	92.7	93.3	94.4
hopper	100.2	113.1	113.8	113.9
walker2d	92.7	110.3	112.4	113.8
<b>Average</b>	80.3	105.4	106.5	107.0

Table 14: Performance comparison across D4RL tasks. For each task, RVDT reports the mean and standard error of normalized scores [16] over 30 random rollouts and averaged over 5 random seeds.

Gym Tasks	CQL	IQL	BCQ	TD3+BC	MoRel	BC	DD	DT	StAR	GDT	CGDT	QT	RVDT
halfcheetah-m-e	91.6	86.7	69.6	90.7	53.3	55.2	90.6	86.8	93.7	93.2	93.6	$93.2 \pm 0.8$	<b>94.4 <math>\pm 0.1</math></b>
hopper-m-e	105.4	91.5	109.1	98.0	108.7	52.5	111.8	107.6	111.1	111.1	107.6	$113.0 \pm 0.2$	<b>113.1 <math>\pm 0.5</math></b>
walker2d-m-e	108.8	109.6	67.3	110.1	95.6	107.5	108.8	108.1	109.0	107.7	109.3	$112.0 \pm 0.3$	<b>112.7 <math>\pm 1.6</math></b>
halfcheetah-m	49.2	47.4	41.5	48.4	42.1	42.6	49.1	42.6	42.9	42.9	43.0	$51.0 \pm 0.2$	<b>51.9 <math>\pm 0.3</math></b>
hopper-m	69.4	66.3	65.1	59.3	95.4	52.9	79.3	67.6	59.5	77.1	96.9	$99.6 \pm 1.3$	<b>100.2 <math>\pm 0.1</math></b>
walker2d-m	83.0	78.3	52.0	83.7	77.8	75.3	82.5	74.0	73.8	76.5	79.1	$87.2 \pm 1.0$	<b>90.2 <math>\pm 0.1</math></b>
halfcheetah-m-r	45.5	44.2	34.8	44.6	40.2	36.6	39.3	36.6	36.8	40.5	40.4	$48.8 \pm 1.5$	<b>53.8 <math>\pm 2.0</math></b>
hopper-m-r	95.0	94.7	31.1	60.9	93.6	18.1	100.0	82.7	29.2	85.3	93.4	$102.1 \pm 0.6$	<b>103.2 <math>\pm 1.9</math></b>
walker2d-m-r	77.2	73.9	13.7	81.8	49.8	32.3	75.0	79.4	39.8	77.5	78.1	$97.8 \pm 0.9$	<b>99.3 <math>\pm 0.8</math></b>
Average	80.6	77.0	53.8	75.3	72.9	52.6	81.8	76.2	66.2	79.1	82.4	89.4	<b>91.2</b>
Adroit Tasks	CQL	IQL	BCQ	BEAR	O-RL	BC	DD	D-QL	DT	StAR	GDT	QT	RVDT
pen-human	37.5	71.5	66.9	-1.0	90.7	63.9	66.7	72.8	79.5	77.9	92.5	$111.9 \pm 11.8$	<b>127.2 <math>\pm 5.5</math></b>
hammer-human	4.4	1.4	0.9	0.3	0.2	1.2	1.9	0.2	3.7	3.7	5.5	$10.4 \pm 5.3$	<b>24.0 <math>\pm 1.5</math></b>
pen-cloned	39.2	37.3	50.9	26.5	60.0	37.0	42.8	57.3	75.8	33.1	86.2	$85.8 \pm 12.6$	<b>117.8 <math>\pm 8.6</math></b>
hammer-cloned	2.1	2.1	0.4	0.3	2.0	0.6	1.7	3.1	3.0	0.3	8.9	$11.8 \pm 5.8$	<b>21.3 <math>\pm 2.7</math></b>
Average	20.8	28.1	29.8	6.5	38.2	25.7	28.3	33.4	40.5	28.8	48.3	55.0	<b>72.6</b>
Kitchen Tasks	CQL	IQL	BCQ	BEAR	O-RL	BC	DD	D-QL	DT	StAR	GDT	QT	RVDT
kitchen-Comp.	43.8	62.5	8.1	0.0	2.0	65.0	65.0	84.0	50.8	40.8	43.8	$81.7 \pm 1.7$	<b>84.5 <math>\pm 2.3</math></b>
kitchen-partial	49.8	46.3	18.9	13.1	35.5	33.8	57.0	60.5	57.9	12.3	73.3	$72.5 \pm 2.5$	<b>75.0 <math>\pm 2.5</math></b>
Average	46.8	54.4	13.5	6.6	18.8	49.4	61.0	72.2	54.4	26.6	58.6	77.1	<b>79.8</b>
Maze2D Tasks	CQL	IQL	BCQ	BEAR	TD3+BC	BC	Diffuser	DD	DT	GDT	QDT	QT	RVDT
maze2d-u	94.7	42.1	49.1	65.7	14.8	88.9	113.9	116.2	31.0	50.4	57.3	$99.2 \pm 1.0$	<b>145.1 <math>\pm 3.8</math></b>
maze2d-m	41.8	34.9	17.1	25.0	62.1	38.3	121.5	122.3	8.2	7.8	13.3	$168.8 \pm 8.5$	<b>183.5 <math>\pm 4.5</math></b>
maze2d-l	49.6	61.7	30.8	81.0	88.6	1.5	123.0	125.9	2.3	0.7	31.0	$242.7 \pm 7.3$	<b>254.3 <math>\pm 4.6</math></b>
Average	62.0	46.2	32.3	57.2	55.2	42.9	119.5	121.5	13.8	19.6	33.9	170.2	<b>194.3</b>
AntMaze Tasks	CQL	IQL	BCQ	BEAR	TD3+BC	BC	DD	D-QL	DT	StAR	GDT	QT	RVDT
antmaze-u	74.0	87.5	78.9	73.0	78.6	54.6	73.1	93.4	59.2	51.3	76.0	$96.0 \pm 3.0$	<b>98.0 <math>\pm 4.0</math></b>
antmaze-u-d	84.0	62.2	55.0	61.0	71.4	45.6	49.2	66.2	53.0	45.6	69.0	$92.0 \pm 6.2$	<b>98.0 <math>\pm 4.0</math></b>
antmaze-m-d	53.7	70.0	0.0	8.0	3.0	0.0	24.6	<b>78.6</b>	0.0	0.0	6.0	$24.0 \pm 5.6$	$30.0 \pm 6.3$
antmaze-l-d	14.9	47.5	2.2	0.0	0.0	0.0	7.5	<b>56.6</b>	0.0	0.0	0.0	$10.0 \pm 0.0$	$10.0 \pm 0.0$
Average	56.6	66.8	34.0	35.5	38.2	25.0	38.6	<b>73.7</b>	28.0	24.2	37.8	57	59.0

#### F.4 Details for Ablation Studies

The ablation variants focus on three key aspects: explicit rebalancing mechanisms, implicit rebalancing mechanisms, and policy types. Explicit rebalancing mechanisms include expert-policy KL regularization and expert-level data duplication. For the latter, the duplicated expert data correspond to the top 50% of trajectories ranked by return, with a single additional copy included. The implicit rebalancing mechanism refers specifically to Q-value guidance. Regarding policy types, we consider two policy types, Gaussian stochastic policies and deterministic policies.

Table 15: Component-level breakdown across ablation variants.

Component	DT	DT-Dup	RDT	QT	VDT	RVDT-Dup	RVDT-Determ	RVDT
Explicit Rebal.	None	Dup.	KL	None	None	Dup.	KL	KL
Implicit Rebal.	None	None	None	Q-value	Q-value	Q-value	Q-value	Q-value
Policy Type	Determ.	Stoch.	Stoch.	Determ.	Stoch.	Stoch.	Determ.	Stoch.

Detailed descriptions of the ablation variants in Table 15 are as follows:

- **DT:** The original Decision Transformer method utilizing deterministic policies, implemented by disabling the Q-value guidance term of QT.
- **DT-Dup:** A stochastic variant of DT enhanced with expert data duplication, modeling the policy as a parameterized Gaussian distribution.
- **RDT:** An RVDT variant without the implicit rebalancing mechanism (Q-value guidance), isolating the effect of the expert-policy KL regularizer.
- **QT:** The original QT model implemented using the official codebase, serving as the deterministic-policy counterpart of RVDT without explicit rebalancing mechanisms.
- **VDT:** An RVDT variant excluding expert-policy KL regularization to specifically isolate the contribution of Q-value guidance.
- **RVDT-Dup:** An RVDT variant employing expert-level data duplication as the explicit rebalancing mechanism, enabling comparison between KL-based rebalancing and data duplication approaches.
- **RVDT-Determ:** A deterministic policy variant of RVDT, replacing the stochastic policy with deterministic outputs and substituting KL regularization with an MSE loss, assessing the role of policy expressiveness.
- **RVDT:** The complete RVDT method integrating expert-policy KL regularization and Q-value guidance with a stochastic policy parameterized by a Gaussian distribution.

#### F.5 Influence of Policy Expressiveness on Performance

To further investigate the influence of policy expressiveness [4] on algorithmic performance, we compare RVDT with several representative Q-learning-based baselines (CQL, IQL, and D-QL) on the D4RL benchmark. As shown in Table 16, CQL and IQL employ Gaussian stochastic policies, while D-QL adopts a more expressive diffusion policy. In contrast, RVDT is built upon conditional sequence modeling (CSM) techniques and incorporates a simple Gaussian stochastic policy.

From the results, we observe that D-QL, which leverages the diffusion policy, significantly outperforms the simpler Gaussian-policy baselines (CQL and IQL), both across most tasks and in terms of overall average performance, demonstrating the advantage of higher policy expressiveness. Nevertheless, RVDT achieves superior performance on most tasks, which highlights the effectiveness of leveraging CSM together with the return-coverage rebalancing mechanism.

However, in the AntMaze tasks with larger mazes ( $-m, -1$ ), D-QL performs better than RVDT. This may be attributed to the multi-legged morphology of the Ant agent, which entails a complex and highly multi-modal action distribution. In these scenarios, the expressiveness of the diffusion policy becomes particularly beneficial, whereas RVDT's Gaussian policy may be insufficient to capture such rich multimodal behaviors.

Table 16: Performance of Q-learning-based baselines (CQL, IQL, D-QL) and RVDT on D4RL tasks, where CQL and IQL adopt Gaussian policies, and D-QL employs a diffusion policy. Results are averaged over 5 random seeds.

Tasks	CQL	IQL	D-QL	<b>RVDT</b>
pen-human	37.5	71.5	72.8	<b>127.2</b>
hammer-human	4.4	1.4	0.2	<b>24.0</b>
pen-cloned	39.2	37.3	57.3	<b>117.8</b>
hammer-cloned	2.1	2.1	3.1	<b>21.3</b>
kitchen-Comp.	43.8	62.5	84.0	<b>84.5</b>
kitchen-partial	49.8	46.3	60.5	<b>75.0</b>
maze2d-m	41.8	34.9	91.0	<b>183.5</b>
maze2d-l	49.6	61.7	200.7	<b>254.3</b>
antmaze-u	74.0	87.5	93.4	<b>98.0</b>
antmaze-u-d	84.0	62.2	66.2	<b>98.0</b>
antmaze-m-d	53.7	70.0	<b>78.6</b>	30.0
antmaze-l-d	14.9	47.5	<b>56.6</b>	10.0
Average	41.2	48.7	72.0	<b>93.5</b>

Overall, these findings suggest that both policy expressiveness and CSM with return-coverage rebalancing are effective factors in improving offline RL performance. The two mechanisms are complementary: diffusion policies enhance local action diversity, while CSM with rebalancing strengthens long-horizon modeling capability. Exploring their integration represents a promising direction for future research.

## F.6 Training Time

To evaluate the computational efficiency of RVDT, we provide a detailed comparison of training times among DT, QT, and RVDT on the D4RL benchmark. The results, summarized in Table 17, report the total training time measured until convergence (120 epochs) on an NVIDIA GeForce RTX 4090 GPU. The reported time for RVDT additionally includes the training phase of the expert policy  $\pi_e$ . As shown in the table, the expert policy training phase typically accounts for approximately one-fifth of the total RVDT training time. When this phase is excluded, the additional overhead introduced by RVDT relative to QT is less than 10%. It can also be observed that QT itself incurs a substantial increase in training time compared to DT, which primarily stems from its additional Q-value learning process.

Table 17: Training time comparison measured until convergence. The reported training time for RVDT additionally includes the expert policy training phase.

Gym Tasks	DT	QT	RVDT (total)	Expert policy in RVDT
halfcheetah-m-e	7.3hr	12.2hr	15.5hr	3.3hr
hopper-m-e	3.6hr	10.3hr	14.6hr	3.0hr
walker2d-m-e	6.1hr	11.7hr	15.4hr	3.3hr
halfcheetah-m	6.3hr	11.8hr	16.1hr	3.2hr
hopper-m	3.5hr	9.2hr	14.0hr	3.2hr
walker2d-m	6.1hr	11.5hr	16.0hr	3.2hr
halfcheetah-m-r	7.2hr	12.2hr	17.4hr	3.3hr
hopper-m-r	3.5hr	8.5hr	12.3hr	3.2hr
walker2d-m-r	6.2hr	11.2hr	14.8hr	2.8hr
<b>Average</b>	<b>5.5hr</b>	<b>11.0hr</b>	<b>15.2hr</b>	<b>3.2hr</b>

## G Discussion

### G.1 Imbalance of Datasets

Learning from imbalanced datasets has long been recognized as a critical challenge in offline reinforcement learning. Such imbalance typically manifests in datasets predominantly composed of low-return trajectories, with relatively few high-return trajectories. This scenario is common in real-world applications since acquiring high-return trajectories is usually more costly compared to low-return ones.

Existing offline RL algorithms, including dynamic programming (DP) based methods, imitation learning (IL) based methods, and conditional sequence modeling (CSM) based methods, encounter significant difficulties when learning from imbalanced datasets. DP-based methods often incorporate inductive biases or regularization terms to enforce policy closeness to the behavioral distribution observed in the dataset, thereby mitigating out-of-distribution (OOD) errors [17, 9], which in turn leads the learned policy to resemble the action distribution of the non-optimal behavior policy. IL-based methods essentially imitate the underlying suboptimal behavior policy that generated the imbalanced dataset, which can result in overly conservative policies due to excessive imitation of actions from the abundant low-performing trajectories. CSM-based methods typically condition on optimal (or near-optimal) returns during inference for decision-making. However, due to the skewed distribution of near-optimal versus low-return trajectories in the training datasets, CSM-based methods frequently train on trajectories with low returns. This discrepancy in return conditioning between the training and inference phases leads CSM methods to experience severe distributional shift problems when conditioning on high returns for inference.

Recent studies describe dataset imbalance from multiple perspectives. For instance, Hong et al. [23] define dataset imbalance through the positive-sided variance of returns (RPSV, defined in Definition 1). Other research interprets dataset imbalance from the perspective of data coverage, where data coverage broadly encompasses both the coverage of state and action spaces [68] and the trajectory return-coverage within the return space [6].

**Definition 1.** *RPSV of a dataset,  $\mathbb{V}_+[g(\tau_i)]$ , corresponds to the second-order moment of the positive component of the difference between trajectory return:  $g(\tau_i) := \sum_{t=0}^{T_i-1} \gamma^t r(s_t^i, a_t^i)$  and its expectation, where  $\tau_i$  denotes a trajectory in the dataset:*

$$\mathbb{V}_+[g(\tau_i)] \doteq \mathbb{E}_{\tau_i \sim \mathcal{D}} \left[ (g(\tau_i) - \mathbb{E}_{\tau_i \sim \mathcal{D}}[g(\tau_i)])_+^2 \right], \text{ with } x_+ = \max\{x, 0\}.$$

### G.2 Expert-Level and Full-Spectrum Return-Coverage

In this paper, we uncover the relationship between the return distribution of the training dataset and the inference-time performance of CSM policies. Since the return distribution of an offline RL dataset corresponds to a discrete distribution over the return space, their probability density functions (PDFs) or cumulative distribution functions (CDFs) are challenging to represent. Therefore, this paper interprets return distributions from the perspective of return-coverage, offering a mathematically tractable description suitable for our theoretical analysis.

It is commonly understood that for a well-trained CSM algorithm, its inference-time performance largely aligns with the initial input return-to-go (RTG) token, which is a manually specified mapping from the initial state  $s_1$  to a real number, i.e.,  $f(s_1) : \mathcal{S} \rightarrow \mathbb{R}$ . The initial RTG token represents the human-expected final return for the test algorithm; thus, it is generally chosen to correspond to the optimal return associated with the optimal policy  $\pi^*$  of the underlying MDP. We denote this optimal policy-derived conditioning function as the *optimal conditioning function*  $f^*$ . The inference-time returns associated with  $f^*$  are referred to as the *optimal conditional returns*. For CSM algorithms, apart from the initial RTG token derived directly from  $f^*$ , subsequent RTG tokens are computed by subtracting the sampled reward  $r$  from the previous RTG token, i.e.,  $f(s_{t+1}) = f(s_t) - r$ . Due to the inherent randomness of  $r$  and the stochastic nature of environmental transitions, subsequent RTG tokens cannot consistently remain optimal [5, 13]. Consequently, returns encountered during policy execution follow a more general conditioning function  $f$ , which with high probability differs from  $f^*$ . We term this broader class of conditioning functions as *runtime conditioning functions*, and the corresponding inference-time returns as *runtime conditional returns*.

Based on the preceding descriptions of optimal conditional returns and runtime conditional returns, we define their corresponding coverage as *expert-level return-coverage* and *full-spectrum return-coverage*, respectively (consistent with definitions presented in Section 4.1). Let  $P_{\pi_\beta}$  denote the distribution over states, actions, and returns induced by the behavior policy  $\pi_\beta$ . We define the *expert-level return-coverage* in an offline dataset generated by behavior policy  $\pi_\beta$  as  $P_{\pi_\beta}(g = f^*(s_1)|s_1)$ , where  $f^*$  corresponds to the optimal conditioning function. Similarly, the *full-spectrum return-coverage* is defined as  $P_{\pi_\beta}(g = f(s_1)|s_1)$ , where  $f$  represents the runtime conditioning function.

Based on these definitions, it becomes evident that return-coverage captures the representation of inference-time conditional returns within the dataset. The central theoretical analysis of this paper (detailed in Section 4) aims to explain how expert-level return-coverage and full-spectrum return-coverage collectively influence the performance of CSM policies. Motivated by this theoretical insight, we propose a simple yet effective return-coverage rebalancing mechanism, designed to substantially enhance the performance of state-of-the-art CSM algorithms.

### G.3 Limitations and Future Directions

Several aspects merit further exploration. First, RVDT involves training an additional imitation policy for the return-coverage rebalancing process, which introduces a modest computational overhead. This cost, however, remains manageable and is well justified by the resulting improvements in policy quality. Future work may explore more efficient architectures or parameter-sharing strategies to further reduce this overhead. Second, the current design of RVDT focuses on the purely offline RL setting. Extending RVDT to online or hybrid settings, where limited online interactions are available, could enable the model to dynamically expand return support beyond the static dataset. In addition, integrating RVDT with data augmentation or generative modeling techniques, such as diffusion-based trajectory stitching or sub-trajectory generation [42, 34], may further enhance return support and improve algorithmic robustness. Third, the current rebalancing mechanism relies on trajectory returns as a measure of trajectory quality. In certain real-world domains, such as healthcare or autonomous driving, reliable return estimation can be challenging. In such cases, alternative evaluation criteria or task-specific metrics could be incorporated to approximate trajectory quality, thereby broadening the applicability of the proposed framework.