How to Leverage Unlabeled Data in Offline Reinforcement Learning

Tianhe Yu [*1 2] **Aviral Kumar** [*3 2] **Yevgen Chebotar** [2] **Karol Hausman** [2] **Chelsea Finn** [1 2] **Sergey Levine** [3 2]

Abstract

Offline reinforcement learning (RL) can learn control policies from static datasets but, like standard RL methods, it requires reward annotations for every transition. In many cases, labeling large datasets with rewards may be costly, especially if those rewards must be provided by human labelers, while collecting diverse unlabeled data might be comparatively inexpensive. How can we best leverage such unlabeled data in offline RL? One natural solution is to learn a reward function from the labeled data and use it to label the unlabeled data. In this paper, we find that, perhaps surprisingly, a much simpler method that simply applies zero rewards to unlabeled data leads to effective data sharing both in theory and in practice, without learning any reward model at all. While this approach might seem strange (and incorrect) at first, we provide extensive theoretical and empirical analysis that illustrates how it trades off reward bias, sample complexity and distributional shift, often leading to good results. We characterize conditions under which this simple strategy is effective, and further show that extending it with a simple reweighting approach can further alleviate the bias introduced by using incorrect reward labels. Our empirical evaluation confirms these findings in simulated robotic locomotion, navigation, and manipulation settings.

1 Introduction

Offline reinforcement learning (RL) provides the promise of a fully data-driven framework for learning performant policies. To avoid costly active data collection and exploration, offline RL methods utilize a previously collected dataset to extract the best possible behavior, making it feasible to use RL to solve real-world problems where active exploration is expensive, dangerous, or otherwise infeasible (Zhan et al., 2021; de Lima & Krohling, 2021; Wang et al., 2018; Kalashnikov et al., 2018). However, this concept is only viable when a significant amount of data for the target task is available in advance. A more realistic scenario might allow for a much smaller amount of task-specific data, combined with a large amount of task-agnostic data, that is not labeled with task rewards and some of which may not be relevant. For example, if our goal is to train a robot to perform a new manipulation task (e.g., cutting an onion), we might have some data of the robot (suboptimally) attempting that task, perhaps collected under human teleoperation and manually labeled with rewards, combined with background data, some of which might be structurally related (e.g., picking up an onion, or cutting a carrot). This scenario presents several questions: How do we decide which prior data should be included when learning the new task? And how do we determine reward labels to use for this prior data?

Prior methods have attempted to answer these two questions, typically in isolation. For the first question, it has been noted that a naïve strategy of sharing all of the prior data can be highly suboptimal (Kalashnikov et al., 2021), even when it is annotated with reward labels, and some works have proposed both manual (Kalashnikov et al., 2021) and automated (Yu et al., 2021a; Eysenbach et al., 2020) data sharing strategies that prioritize the most structurally similar prior data. However, these methods assume that shared data can automatically be relabeled with the reward function for the task, but the assumption that we have access to the functional form of this reward is a strong one: for example, in many real-world settings, computing the reward might require human labeling (Cabi et al., 2019; Finn et al., 2016b). Prior works use learned classifiers for reward labeling (Fu et al., 2018b; Xie et al., 2018; Singh et al., 2019), or other automated mechanisms (Konyushkova et al., 2020). But these mechanisms themselves add complexity and potential brittleness to the pipeline. Thus, we aim to study the possibility of tackling this problem with a simple method that determines which rewards to use for shared data, with minimal supervision and no additional modeling and learning.

In this paper, we make the potentially surprising observation that prior data such as data from other tasks can be utilized with naïve constant reward labels in offline RL, which corresponds to zero reward or whatever is the minimum reward

[*]Equal contribution [1]Stanford University [2]Google Research [3]UC Berkeley. Correspondence to: Tianhe Yu <tianheyu@cs.stanford.edu>, Aviral Kumar <aviralk@berkeley.edu>.

Proceedings of the 39^{th} International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022.

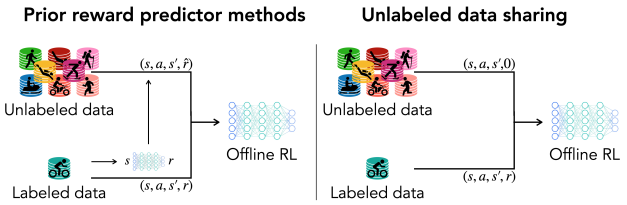

Figure 1. Overview of various methods dealing with unlabeled offline data. UDS is a simple method compared to the complex reward learning approaches yet achieves effective results.

for the task (which we can set to zero via rescaling, without loss of generality). It may at first seem that such an approach would lead to incorrect solutions due to using the wrong reward values. However, we show that this simple reward relabeling strategy, which we refer to as unlabeled data sharing (UDS), can outperform more sophisticated methods that separately learn a reward model and lead to good results in a range of settings. We visualize UDS along with more complex methods in Figure 1. Naïve UDS does not work in all cases, but we further show that carefully reweighting the unlabeled transitions (to modify their distribution) can significantly increase the applicability of this approach and reduce the effects of reward bias, while still preserving many of the benefits. We analyze this theoretically and show that, in practice, a method based on conservative data sharing (Yu et al., 2021a), which reweights unlabeled data to minimize distributional shift (originally designed for use with ground truth reward labels) can be particularly effective in this role.

Our main contribution is the finding that simply labeling unlabeled data with a reward of zero for use in offline RL is surprisingly effective in many cases. We perform extensive theoretical and empirical analysis to study conditions under which this simple approach, UDS, would either excel or fail, and we analyze how reweighting the relabeled data (while still using zero reward as the label) can increase the range of settings when UDS is successful. Our empirical evaluation, conducted over various single-task and multi-task offline RL scenarios such as locomotion, robotic manipulation from visual inputs, and ant-maze navigation shows that this simple zero reward strategy leads to improved performance, even compared to methods that use more sophisticated learning and label propagation strategies to infer rewards. This further supports our hypothesis that relabeling unlabeled data with zero reward is an effective approach for utilizing unlabeled prior data in offline RL.

2 Related Work

Offline RL. Offline RL (Ernst et al., 2005; Riedmiller, 2005; Lange et al., 2012; Levine et al., 2020) considers the problem of learning a policy from a static dataset without interacting with the environment, which has shown promises in many practical applications such as robotic control (Kalashnikov et al., 2018; Rafailov et al., 2021), NLP (Jaques et al., 2019), and healthcare (Shortreed et al., 2011; Wang et al., 2018). The main challenge of offline RL is the distributional shift between the learned policy and the behavior policy (Fujimoto et al., 2018), which can cause erroneous value backups. To address this issue, prior methods have constrained the learned policy to be close to the behavior policy via policy regularization (Liu et al., 2020; Wu et al., 2019; Kumar et al., 2019; Zhou et al., 2020; Ghasemipour et al., 2021; Fujimoto & Gu, 2021), conservative value functions (Kumar et al., 2020), and model-based training with conservative penalties (Yu et al., 2020c; Kidambi et al., 2020; Swazinna et al., 2020; Lee et al., 2021; Yu et al., 2021b). Unlike these prior works, we study how unlabeled data can be incorporated into the offline RL framework.

RL with unlabeled data. Prior works tackle the problem of learning from data without reward labels via either directly imitating expert trajectories (Pomerleau, 1988; Ross & Bagnell, 2012; Ho & Ermon, 2016), learning reward functions from expert data using inverse RL (Abbeel & Ng, 2004; Ng & Russell, 2000; Ziebart et al., 2008; Finn et al., 2016a; Fu et al., 2018a; Konyushkova et al., 2020), or learning a reward / value classifier that discriminates successes and failures (Xie et al., 2018; Fu et al., 2018b; Singh et al., 2019; Eysenbach et al., 2021) in standard online RL settings. Unlike these prior works, the goal of our paper is not to propose a sophisticated new algorithm for reward learning, but rather to study when the simple solution of using zero as the reward label can work well from offline data. While Singh et al. (2020) also considers relabeling prior data with zero reward in the standard, single-task offline RL setting, the key distinction is that the unlabeled data in Singh et al. (2020) cannot solve the target task and *actually* gets zero rewards and is thus labeled with the true reward. Unlike Singh et al. (2020), we show that UDS can surprisingly also be effective even when the unlabeled data is incorrectly labeled with zero reward, and discuss how it can be improved (Section 4.2). Additionally, we consider both single-task and multi-task offline RL settings.

Data sharing. Sharing data across different tasks has been found to be effective in multi-task (Eysenbach et al., 2020; Kalashnikov et al., 2021; Yu et al., 2021a) and meta-RL (Dorfman & Tamar, 2020; Mitchell et al., 2021) and it improves performance significantly in multi-task offline RL. Prior works share data based on learned Q-values (Eysenbach et al., 2020; Li et al., 2020; Yu et al., 2021a), domain knowledge (Kalashnikov et al., 2021) and distance to goals in goal-conditioned settings (Andrychowicz et al., 2017; Liu et al., 2019; Sun et al., 2019; Lin et al., 2019; Chebotar et al., 2021), and the learned distance with robust inference in the offline meta-RL setting (Li et al., 2019). However, all of these either require access to the functional form of the reward for relabling or are limited to goal-conditioned settings. In contrast, we attempt to tackle a more general problem where reward labels are not provided for a subset

of the data, either in a multi-task or a single-task setting and find that simple approaches for relabeling data from other tasks with the constant zero can work well.

3 Preliminaries

Offline RL. Standard RL considers a Markov decision process (MDP), $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, \gamma, R)$, where $\mathcal{S}$ and $\mathcal{A}$ denote the state and action spaces respectively, $P(\mathbf{s}'|\mathbf{s}, \mathbf{a})$ denotes the dynamics, $\gamma \in [0, 1)$ is the discount factor, and R correspond to the reward function. Offline RL tackles the problem of learning a policy $\pi(\mathbf{a}|\mathbf{s})$ from a static dataset with $\mathcal{D}$, generated by a behavior policy $\pi_\beta(\mathbf{a}|\mathbf{s})$.

Data sharing in offline RL. Data sharing has been considered in the multi-task offline RL setting where there is a static multi-task dataset with $\mathcal{D} = \cup_{i=1}^{N} \mathcal{D}_i$ where N is the number of tasks. Prior works (Kalashnikov et al., 2021; Eysenbach et al., 2020; Yu et al., 2021a) show that sharing data from different tasks to task i to be conducive. To do so, these prior methods assume access to the functional form of the reward r_i. This is a strong assumption in practice, as it necessitates access to a functional (programmatic) form for the reward function. In offline RL, it might be desirable to simply label the reward function by hand, but then the algorithm does not have access to the functional form of the reward, and all unlabeled data also needs to be labeled by hand for use with such methods. Our aim in this paper is to utilize unlabeled data without any reward labels at all. If however functional access to the reward *is* available, a simple strategy is to naïvely share data across all tasks, which we refer to as Sharing All. Formally, Sharing All defines the dataset of transitions relabeled from task j to task i as $\mathcal{D}_{j \to i}$ and the method can be then defined as $\mathcal{D}_i^{\text{eff}} := \mathcal{D}_i \cup (\cup_{j \neq i} \mathcal{D}_{j \to i})$, where $\mathcal{D}_i^{\text{eff}}$ denotes the effective dataset for task i. Therefore, the policy optimization objective in Sharing All can be written as follows:

$$\forall i \in [N], \;\; \pi^*(\mathbf{a}|\mathbf{s}, i) := \arg\max_{\pi} \;\; J_{\mathcal{D}_i^{\text{eff}}}(\pi) - \alpha D(\pi, \pi_\beta^{\text{eff}}),$$

where $\pi_\beta^{\text{eff}}(\mathbf{a}|\mathbf{s}, i)$ is the effective behavior policy for task i denoted as $\pi_\beta^{\text{eff}}(\mathbf{a}|\mathbf{s}, i) := |\mathcal{D}_i^{\text{eff}}(\mathbf{s}, \mathbf{a})|/|\mathcal{D}_i^{\text{eff}}(\mathbf{s})|$, $J_{\mathcal{D}_i^{\text{eff}}}(\pi)$ denotes the average return of policy π in the empirical MDP induced by the effective dataset, and $D(\pi, \pi_\beta^{\text{eff}})$ denotes a divergence measure (e.g., KL-divergence (Jaques et al., 2019; Wu et al., 2019), fisher divergence (Kostrikov et al., 2021), MMD distance (Kumar et al., 2019) or D_{CQL} from conservative Q-values (Kumar et al., 2020)) between the learned policy π and the effective behavior policy π_β^{eff}. Note that conservative Q-values refer to the Q-value for a given policy corresponding to a modified reward function $r(\mathbf{s}, \mathbf{a}) - \alpha\pi(\mathbf{a}|\mathbf{s}) \cdot (\pi(\mathbf{a}|\mathbf{s})/\pi_\beta(\mathbf{a}|\mathbf{s}) - 1)$, computed on the empirical MDP. We also note that Sharing All can be easily adapted to the single-task setting where there is only one target task with labeled data $\mathcal{D}_{\text{L}}$ and unlabeled prior data $\mathcal{D}_{\text{U}}$. While data sharing tends to show promising results, it requires the assumption of the access to the functional form of the reward function. We instead focus on the data sharing problem where we do not make such an assumption and instead, only have the reward labels for originally commanded task, which we will discuss in the following section.

4 How To Use Unlabeled Data in Offline RL

Arguably the easiest approach to utilize unlabeled data in offline RL is to simply assign the lowest possible reward to all the transitions in the unlabeled data, which we will assume to be 0 without loss of generality, and use this data for training the underlying offline RL method. We will refer to this approach as UDS. We will show that, although this strategy might seem simplistic, in fact, it can work well both in theory and practice, when the dataset composition and the task satisfies certain criteria. Not all tasks and datasets satisfy these criteria, but we will argue that many realistic offline RL problems do satisfy them. For example, UDS can work well in a problem where the labeled data consists of high-quality, near-expert demonstrations, while the unlabeled data consists of mediocre or low-reward interactions in the environment, as is often the case in robotics (Xie et al., 2019). We will analyze the performance of this approach theoretically in Section 4.1 and empirically in Section 5.2.

Although the simple UDS approach can perform well in some scenarios, relabeling with zero reward of course also biases the learning process. We therefore further show that reward bias can be mitigated if the unlabeled transitions are additionally reweighted, so as to change the unlabeled data distribution (while still using a label of zero for all unlabeled data). While such reweighting reduces the sample size, it can provide an overall benefit by reducing the reward bias and distributional shift. We will derive the optimal scheme for reweighting the transitions in the unlabeled dataset that minimizes the impact of reward bias in Section 4.2. Surprisingly, our analysis reveals that a near-optimal solution to reducing reward bias is to combine UDS with an already existing reweighting scheme for multi-task offline RL with full reward information. For example, one can choose to reweight unlabeled data with the CDS scheme of Yu et al. (2021a), which preferentially upweights transitions based on their conservative Q-values.

4.1 Theoretical Analysis of UDS in Offline RL

In this section, we will derive a performance bounds for UDS that allow us to understand several cases when this simple baseline approach still works well. We will show that the increase in the effective dataset size can often outweigh the bias incurred from using the wrong reward in several practical situations. We will also discuss conditions on the data composition and the relative distributions of labeled and unlabeled data that enable UDS to be successful. Formally, UDS trains on an effective dataset given by:

$$\mathcal{D}^{\text{eff}} = \mathcal{D}_{\text{L}} \cup \{(\mathbf{s}_j, \mathbf{a}_j, \mathbf{s}'_j, 0) \in \mathcal{D}_{\text{U}}\}. \tag{1}$$

We now present a policy improvement guarantee for UDS below, and then analyze the bound under special conditions. Our theoretical result builds on techniques for showing safe policy improvement bounds (Laroche et al., 2019; Kumar et al., 2020; Yu et al., 2021a).

Theorem 4.1 (**Policy improvement guarantee for UDS**). *Let π^*_{UDS} denote the policy learned by UDS, and let $\pi^{\text{eff}}_\beta(\mathbf{a}|\mathbf{s})$ denote the behavior policy for the combined dataset $\mathcal{D}^{\text{eff}}$. Then with high probability $\geq 1-\delta$, π^*_{UDS} is a safe policy improvement over π^{eff}_β, i.e.,*

$$J(\pi^*_{UDS}) \geq J(\pi^{\text{eff}}_\beta) - \zeta_{err} + \underbrace{\frac{\alpha}{1-\gamma} D(\pi^*_{UDS}, \pi^{\text{eff}}_\beta)}_{\text{(c): policy improvement}},$$

$$\zeta_{err} = \underbrace{\text{RewardBias}(\pi^*_{UDS}, \pi^{\text{eff}}_\beta)}_{\text{(a)}}$$

$$\underbrace{+ \mathcal{O}\left(\frac{\gamma}{(1-\gamma)^2}\right) \mathbb{E}_{\mathbf{s},\mathbf{a}\sim\widehat{d}^\pi}\left[\sqrt{\frac{D_{CQL}(\pi^*_{UDS}, \pi^{\text{eff}}_\beta)(\mathbf{s})}{|\mathcal{D}^{\text{eff}}(\mathbf{s})|}}\right]}_{\text{(b): sampling error}},$$

$$(a) := \frac{\sum_{\mathbf{s},\mathbf{a}} \Delta\left(\widehat{d}^{\pi_\beta{}^{\text{eff}}}, \widehat{d}^{\pi^*_{UDS}}\right) \cdot (1 - f(\mathbf{s},\mathbf{a})) \cdot r(\mathbf{s},\mathbf{a})}{1-\gamma},$$

where we use the notation $f(\mathbf{s},\mathbf{a}) := \frac{|\mathcal{D}_{\text{L}}(\mathbf{s},\mathbf{a})|}{|\mathcal{D}^{\text{eff}}(\mathbf{s},\mathbf{a})|}$ and $\Delta(\widehat{d}^{\pi_\beta{}^{\text{eff}}}, \widehat{d}^{\pi^*_{UDS}}) = \widehat{d}^{\pi_\beta{}^{\text{eff}}}(\mathbf{s},\mathbf{a}) - \widehat{d}^{\pi^*_{UDS}}(\mathbf{s},\mathbf{a})$.

A proof for Theorem 4.1 is provided in Appendix A.2. Intuitively, term (a) corresponds to the potential suboptimality incurred as a result of using the wrong reward, term (b) corresponds to the suboptimality induced due to sampling error. Finally, term (c) corresponds to the policy improvement in the empirical MDP induced by the transitions in $\mathcal{D}^{\text{eff}}$ that occurs as a result of offline RL. Intuitively, we expect that including more unlabeled data will reduce the sampling error (b) and potentially increase how much we can improve the policy (c), while potentially increasing the reward bias term (a). The key question is under which conditions we would expect the increase in bias (a) to be lower than the decrease in term (b) obtained from using the unlabeled data. We examine this question below, presenting a few common cases where we expect this tradeoff to be favorable.

(I) Unlabeled data is distributed identically as labeled data. The first special case we consider is when the distribution of state-action pairs in $\mathcal{D}_{\text{L}}$ is identical to the distribution of state-action pairs in the unlabeled dataset $\mathcal{D}_U$. This can arise in many application domains, such as in robotics (Xie et al., 2019; Dasari et al., 2020), where a large amount of offline data is available, but only a limited uniformly-selected fraction of it can be annotated with rewards. In this case, the fraction $f(\mathbf{s},\mathbf{a})$ takes on identical values for all state-action pairs, and term (a) simply reverts to be a difference between the performance of the effective behavior policy and the learned policy, in the empirical MDP. Since offline RL algorithms would improve over the effective behavior policy in the empirical MDP, this term is negative and hence, no additional suboptimality is incurred in the reward bias term. Moreover, the sampling error term (b) reduces when more data is utilized. Thus, in this case, UDS improves performance due to an increase in the dataset size $|\mathcal{D}^{\text{eff}}(\mathbf{s})|$, without incurring a cost due to the wrong reward.

(II) Quality of the unlabeled dataset. In practice, we are often provided with a small amount of high-quality reward annotated demonstration data, and a lot of unlabeled prior data of low or mediocre quality. In this case, assigning a low reward to the transitions in the unlabeled dataset does not negatively affect policy performance significantly because the bias due to wrong reward is low (due to low-quality of labeled data) and reduces sampling error (term (b)). On the other hand, when the unlabeled data is of high quality and large in size compared to the labeled dataset, even then the performance of the policy $J(\pi^*_{\text{UDS}})$ can be improved by using this unlabeled set since UDS improves $J(\pi^{\text{eff}}_\beta)$.

(III) Large unlabeled datasets for long-horizon tasks. Another scenario when UDS relabeling will be beneficial to do compared to not using the unlabeled data at all is in long-horizon tasks (large value of $1/(1-\gamma)$), where a lot of unlabeled data is available, while the labeled dataset, $\mathcal{D}_{\text{U}}$ is relatively very small. Define $H = \frac{1}{1-\gamma}$; then in the case that $|\mathcal{D}^{\text{eff}}(\mathbf{s})| = \Omega(H^2)|\mathcal{D}_{\text{L}}(\mathbf{s})|$, the sampling error term (term (b)) will consist of one less factor of $1/(1-\gamma)$ when utilizing unlabeled data, compared to when it is not. Since the sampling error grows quadratically in the horizon, whereas the reward bias only grow linearly, a reduction in this term by increasing $|\mathcal{D}^{\text{eff}}(\mathbf{s})|$ (i.e., denominator) can improve compared to only using the labeled data, $\mathcal{D}_{\text{L}}$. Thus, even though the rewards may be biased, the addition of large amounts of unlabeled, diverse data in long-horizon tasks can help despite the reward bias incurred.

We empirically verify that UDS indeed helps in the special cases discussed above in Section 5.2. However, there are also cases where UDS does not help because of large suboptimality induced due to term (a). In Table 1, we present a summary of the scenarios under which we expect UDS will help or hurt performance. As an example of the latter, when the amount of unlabeled data, $\mathcal{D}_{\text{U}}$ is not very large compared to the labeled dataset, such that a decrease in sampling error does not outweigh the suboptimality induced by reward bias, we should not expect UDS to attain very good performance, which we empirically show in Appendix B.2. To handle such cases, our key idea is to re-weight the unlabeled dataset before using it for offline RL training reduce the sampling error and reward bias.

Table 1. Summary of scenarios where UDS is expected to work and where it is not expected to work. **L** denotes the characteristics of labeled data, **U** denotes characteristics of unlabeled data. Limited/Abundant refers to the relative amount of data available High-quality/medium-quality/low-quality refers to the actual performance of the behavior policy generating the datasets. Narrow/broad refers to the relative state coverage of the datasets that we study where high coverage can lead to low distributional shift during offline RL. We provide intuitions on why UDS works or does not work under each scenario and refer to the cases shown in our analysis in Section 4.1.

Scenarios	UDS	Intuition
L: limited + high-quality + narrow, **U**: abundant + low/medium-quality + broad	✓	increase coverage (case **II** and **III**)
L: limited + medium-quality + narrow, **U**: abundant + low/medium-quality + broad	×	reward bias outweighs high coverage
L: limited + high-quality + narrow, **U**: abundant + high-quality + narrow	✓	identical distribution (case **I**)
L: limited + low-quality + narrow, **U**: abundant + high-quality + narrow	✓	increase data quality (case **II**)

4.2 Reducing Reward Bias by Reweighting Data

As discussed above, one way to reduce the suboptimality induced due to reward bias is by preferentially reweighting transitions in the unlabeled data. We would hope that such a scheme can provide a favorable bias-sampling error tradeoff, even though it reduces sample size. In this section, we will derive an optimized reweighting scheme that attains a favorable tradeoff. Intuitively, this optimized scheme suggests that reweighting must reduce distributional shift against the state-action marginal of the policy obtained by running offline RL on only the transitions in the labeled data. This scheme intuitively matches the conservative data sharing method Yu et al. (2021a) previously proposed for (fully-labeled) multi-task offline RL. We will show that this conservative sharing approach reduces reward bias, controls distributional shift that appears in the numerator of sampling error, and increases the sample size.

Formally, we will derive this reweighting scheme from the perspective of optimizing the effective behavior policy, π_β^{eff} induced by the dataset $\mathcal{D}$ after preferentially sharing transitions from the unlabeled data, so as to minimize reward bias and sampling error, while improving the dataset quality. For understanding purposes, we begin by deriving the choice of π_β^{eff} that reduces only the reward bias component:

Theorem 4.2 (**Optimized reward bias reduction**). *The optimal effective behavior policy that minimizes the bias (a) in Theorem 4.1,* $\text{RewardBias}(\pi, \pi_\beta^{\text{eff}})$*, satisfies*

$$\widehat{d}^{\pi_\beta^{\text{eff}}}(\mathbf{s}, \mathbf{a}) \propto \sqrt{d_{\text{L}}(\mathbf{s}, \mathbf{a}) d^\pi(\mathbf{s}, \mathbf{a})},$$

where d^π denotes the state-action marginal of a policy π, and $d_{\text{L}}(\mathbf{s}, \mathbf{a})$ denotes the density of state-action pair $(\mathbf{s}, \mathbf{a})$ under the labeled dataset.

A proof of Theorem 4.2 is provided in Appendix A.3. The expression implies that the state-action marginal of the effective behavior policy (i.e., the rebalanced dataset distribution) must place mass on state-action tuples that are both likely to under the learned policy d^π and appear frequently in the distribution induced by the labeled dataset $\widehat{d}_{\text{L}}$. However, note that Theorem 4.2 only accounts for the reward bias and not the other terms pertaining to sampling error and the performance of the effective behavior policy that appear in Theorem 4.1. To address both of these issues, in our next theoretical result, Theorem 4.3, we derive the reweighting distribution that controls all the terms.

Theorem 4.3 (**Optimized reweighting unlabeled data**; Informal)**.** *The optimal effective behavior policy that maximizes a lower bound on $J(\pi_\beta^{\text{eff}}) - [(a) + (b)]$ in Theorem 4.1 satisfies $d^{\widehat{\pi}_\beta^{\text{eff}}}(\mathbf{s}, \mathbf{a}) = p^*(\mathbf{s}, \mathbf{a})$, where:*

$$p^* = \arg\min_{p \in \Delta^{|\mathcal{S}||\mathcal{A}|}} \sum_{\mathbf{s},\mathbf{a}} C_1 \frac{\widehat{d}^\pi(\mathbf{s}, \mathbf{a})}{\sqrt{p(\mathbf{s}, \mathbf{a})}} + C_2 |\widehat{d}_{\text{L}}(\mathbf{s}, \mathbf{a})| \frac{\widehat{d}^\pi(\mathbf{s}, \mathbf{a})}{p(\mathbf{s}, \mathbf{a})},$$

where C_1 and C_2 are universal positive constants that depend on the sizes of the labeled and unlabeled datasets.

A proof of Theorem 4.3 along with a more formal statement listing the constants C_1 and C_2 is provided in in Appendix A.4. The first term in the optimization objective of p^* appearing above arises from the sampling error term, while the second term corresponds to the reward bias term in Theorem 4.1. The optimal solution for p^* must place high density on $(\mathbf{s}, \mathbf{a})$ pairs where $\widehat{d}^\pi(\mathbf{s}, \mathbf{a})$ is high, because this would reduce the objective in the optimization problem above. This corroborates the analysis of Yu et al. (2021a), which shows that utilizing a reweighting scheme that reduces distributional shift (i.e., makes $\pi(\mathbf{a}|\mathbf{s})/\widehat{\pi}_\beta^{\text{eff}}(\mathbf{a}|\mathbf{s})$ or equivalently, as we find, making $\widehat{d}^\pi(\mathbf{s}, \mathbf{a})/\widehat{d}^{\pi_\beta^{\text{eff}}}(\mathbf{s}, \mathbf{a})$ smaller) will control sampling error, and give rise to performance benefits. In addition, as also shown in Theorem 4.2, the reward bias term is also controlled when low distributional shift appears. This means that rebalancing the dataset to control distributional shift between the learned policy $\widehat{d}^\pi(\mathbf{s}, \mathbf{a})$ and $\widehat{d}^{\pi_\beta^{\text{eff}}}(\mathbf{s}, \mathbf{a})$ is effective in unlabeled settings as well.

While the scheme derived above can, in principle, be implemented exactly in practice, it would require computing state-marginals. Since, computing state marginals accurately in an offline setting has been an outstanding challenge, we instead can utilize a reweighting scheme that corrects for distributional shift approximately without needing state-marginals. One of such methods is conservative data sharing (CDS) (Yu et al., 2021a) that can be implemented without requiring additional components beyond the machinery of

Table 2. Results on single-task environments hopper and AntMaze from the D4RL (Fu et al., 2020) benchmark. The numbers are averaged over three random seeds. We only bold the best-performing method that does not have access to the true reward for relabeling. UDS outperforms No Sharing in three out of the four settings while achieving competitive performances compared to Sharing All in all the settings. CDS+UDS further improves over UDS in three out of four settings.

Environment	Labeled data	Unlabeled data	CDS+UDS	UDS	No Sharing	Reward Pred.	VICE	RCE	Oracle Methods CDS	Oracle Methods Sharing All
D4RL hopper	expert	random	**81.5**	78.6	77.1	67.6	n/a	n/a	83.3	86.1
	expert	medium	**78.3**	64.4	77.1	51.7	n/a	n/a	82.5	64.6
D4RL AntMaze	expert	medium-play	82.6	**82.7**	17.2	0.0	0.0	0.0	83.5	83.1
	expert	large-play	**47.1**	33.1	0.7	0.0	0.0	0.0	46.1	50.2

the offline RL method. Formally, CDS is given by:

$$\mathcal{D}_i^{\text{eff}} = \mathcal{D}_i \cup (\cup_{j \neq i} \{(\mathbf{s}_j, \mathbf{a}_j, \mathbf{s}'_j, r_i) \in \mathcal{D}_{j \to i} : \Delta^{\pi}(\mathbf{s}, \mathbf{a}) \geq 0\}),$$

where $\mathbf{s}_j, \mathbf{a}_j, \mathbf{s}'_j$ denote the transition from $\mathcal{D}_j$, r_i denotes the reward of $\mathbf{s}_j, \mathbf{a}_j, \mathbf{s}'_j$ relabeled for task i, π denotes the task-conditioned policy $\pi(\cdot|\cdot, i)$, $\Delta^{\pi}(\mathbf{s}_j, \mathbf{a}_j)$ is the condition that shares data only if the expected conservative Q-value of the relabeled transition exceeds the top k-percentile of the conservative Q-values of the original data. In our experiments and analysis in Section 5, we find that utilizing CDS improves performance over simply training with UDS in a number of domains supporting the theoretical analysis.

4.3 Why Can We Expect UDS to Outperform Reward Prediction Methods?

While we have discussed various conditions where we would expect UDS (with or without various reweighting methods) to improve final performance over an approach that does not utilize the unlabeled data, it is also instructive to consider how it comes to a method that instead trains an approximate reward function, $\widehat{r}_\phi(\mathbf{s}, \mathbf{a})$ using the labeled data, and then uses this approximate reward to annotate the unlabeled data. In our experiments, we will show, perhaps surprisingly that UDS often outperforms prior reward learning methods. In this section, we provide some intuition for why. First, we note the following generic expression for reward bias (term (a) in Theorem 4.1) for any approach:

$$\text{RewardBias}(\pi, \pi_\beta^{\text{eff}}) = \frac{\sum_{\mathbf{s},\mathbf{a}} \Delta\left(\widehat{d}^{\pi_\beta{}^{\text{eff}}}, \widehat{d}^{\pi}\right) \cdot \Delta r(\mathbf{s}, \mathbf{a})}{1 - \gamma},$$

where $\Delta r(\mathbf{s}, \mathbf{a})$ is the error in the reward applied to the unlabeled data, such that $\Delta r(\mathbf{s}, \mathbf{a}) = (1 - f(\mathbf{s}, \mathbf{a}))r(\mathbf{s}, \mathbf{a})$ for UDS, and $\Delta r(\mathbf{s}, \mathbf{a}) = r(\mathbf{s}, \mathbf{a}) - \widehat{r}_\phi(\mathbf{s}, \mathbf{a})$ for a reward prediction method. Note that while for UDS, $\forall\, \mathbf{s}, \mathbf{a},\ \Delta r(\mathbf{s}, \mathbf{a}) \geq 0$, this is not necessarily the case for a generic reward prediction method.

UDS. Since $\Delta r(\mathbf{s}, \mathbf{a}) \geq 0$ for all state-action tuples, state-action pairs that appear more frequently under the learned policy compared to the effective behavior policy, i.e., when $\Delta\left(\widehat{d}^{\pi_\beta{}^{\text{eff}}}, \widehat{d}^{\pi}\right) < 0$, contribute to reducing the suboptimality induced due to reward bias.

Reward prediction. When $\Delta r(\mathbf{s}, \mathbf{a}) = r(\mathbf{s}, \mathbf{a}) - \widehat{r}_\phi(\mathbf{s}, \mathbf{a})$, $\Delta r(\mathbf{s}, \mathbf{a})$ may not be positive on all state-action tuples, and thus reward prediction methods fail to reduce the contribution of such state-action pairs in term (a). Since policy optimization will seek out those policies that maximize $\widehat{d}^{\pi}(\mathbf{s}, \mathbf{a})$ on state-action pairs with high rewards, state-action pairs where $\Delta(\widehat{d}^{\pi_\beta^{\text{eff}}}, \widehat{d}^{\pi}) < 0$ (i.e., state-action pairs that appear more under the learned policy) are likely to also have $\Delta r(\mathbf{s}, \mathbf{a}) < 0$. This "exploitation" inhibits the correction of reward bias and provides an explanation for why reward prediction approaches may still not perform well due to overestimation errors in the reward function.

5 Experiments

In our experiments, we aim to evaluate whether the theoretical potential for simple minimum-reward relabeling to attain good results is reflected in benchmark tasks and more complex offline RL settings. In particular, we will study: **(1)** can UDS match or exceed the performance of sophisticated reward inference methods and methods with oracle reward functions in simulated locomotion and navigation tasks? **(2)** can an optimized reweighting strategy (e.g., CDS+UDS) further improve the results achieved by UDS?, **(3)** how does UDS behave with and without combining with an optimized reweighting strategy in multi-task offline RL settings?, **(4)** under which conditions does UDS work and not work and does optimizing for reweighting help?

Comparisons. We evaluate multiple approaches alongside with **UDS** and **CDS+UDS**: **No Sharing**, which uses the labeled data only without using any of the unlabeled data, **Reward Predictor**, which is trained via supervised classification or regression to directly predict the reward in sparse and dense reward settings respectively, **VICE** (Fu et al., 2018c) and **RCE** (Eysenbach et al., 2021), inverse RL methods only applicable to sparse reward settings, that learn either a reward or Q-function classifier from both the labeled data and unlabeled data (treated as negatives) and then annotate the unlabeled data with the learned classifier, In the multi-task setting, we modify **VICE** and **RCE** accordingly by extracting transitions with reward labels equal to 1 and treating these datapoints as positives to learn the classifier for each task. We also train **No Sharing**, **Reward Predictor**,

Table 3. Results for multi-task robotic manipulation (Meta-World) and navigation environments (AntMaze) with low-dimensional state inputs. Numbers are averaged across 6 seeds, ± the 95%-confidence interval. We take the results of No Sharing, Sharing All and CDS, directly from (Yu et al., 2021a). We bold the best-performing method that does not have access to the true rewards for relabeling. We include per-task performance for Meta-World domains and the overall performance averaged across tasks (highlighted in gray) for all three domains. Both CDS+UDS and UDS outperforms prior vanilla multi-task offline RL approach (No Sharing) and reward learning methods (Reward Predictor, VICE and RCE) while performing competitively compared to oracle reward relabeling methods.

Environment	Tasks	CDS+UDS	UDS	VICE	RCE	No Sharing	Reward Pred.	Oracle Methods CDS	Oracle Methods Sharing All
Meta-World	door open	**61.3%**±7.9%	51.9%±25.3%	0.0%±0.0%	0.0%±0.0%	14.5%±12.7%	0.0%±0.0%	58.4%±9.3%	34.3%±17.9%
	door close	54.0% ±42.5%	12.3%±27.6%	66.7%%±47.1%	0.0%±0.0%	4.0%±6.1%	**99.3%**±0.9%	65.3%±27.7%	48.3%±27.3%
	drawer open	**73.5%**±9.6%	61.8%±16.3%	0.0%±0.0%	0.0%±0.0%	16.0%±17.5%	13.3%±18.9%	57.9%±16.2%	55.1%±9.4%
	drawer close	99.3%±0.7%	**99.6%**±0.7%	19.3%±27.3%	2.7%±1.7%	99.0%±0.7%	50.3%±35.8%	99.0%±0.7%	98.8%±0.7%
	average	**71.2%** ± 11.3%	56.4%±12.8%	21.5%±0.7%	0.7%±0.4%	33.4%±8.3%	41.0%±11.9%	70.1%±8.1%	59.4%±5.7%
AntMaze	medium (3 tasks)	**31.5%**±3.0%	26.5%±9.1%	2.9%±1.0%	0.0%±0.0%	21.6%±7.1%	3.8%±3.8%	36.7%±6.2%	22.9%±3.6%
	large (7 tasks)	**18.4%**±6.1%	14.2%±3.9%	2.5%±1.1%	0.0%±0.0%	13.3% ± 8.6%	5.9%±4.1%	22.8% ± 4.5%	16.7% ± 7.0%

Table 4. Results for multi-task imaged-based robotic manipulation domains in (Yu et al., 2021a). Numbers are averaged across 3 seeds, ± the 95% confidence interval. UDS outperforms No Sharing in 7 out of 10 tasks as well as the average task performance, while performing comparably to Sharing All. CDS+UDS further improves the performance of UDS and outperforms No Sharing in all of the 10 tasks.

Task Name	CDS+UDS	UDS	No Sharing	CDS (oracle)	Sharing All (oracle)
`lift-banana`	**55.9%**±11.7%	48.6%±5.1%	20.0%±6.0%	**53.1%**±3.2%	41.8%±4.2%
`lift-bottle`	**72.9%**±12.8%	58.1%±3.6%	49.7%±8.7%	**74.0%**±6.3%	60.1%±10.2%
`lift-sausage`	**74.3%**±8.3%	66.8% ± 2.7%	60.9%±6.6%	**71.8%**±3.9%	70.0%±7.0%
`lift-milk`	73.5%±6.7%	**74.5%**±2.5%	68.4%±6.1%	**83.4%**±5.2%	72.5%±5.3%
`lift-food`	**66.3%**±8.3%	53.8%±8.8%	39.1%±7.0%	**61.4%**±9.5%	58.5%±7.0%
`lift-can`	**64.9%**±7.1%	61.0%±6.8%	49.1%±9.8%	**65.5%**±6.9%	57.7%±7.2%
`lift-carrot`	**84.1%**±3.6%	73.4%±5.8%	69.4%±7.6%	**83.8%**±3.5%	75.2%±7.6%
`place-bowl`	**83.4%**±3.6%	77.6%±1.6%	80.3%±8.6%	**81.0%**±8.1%	70.8%±7.8%
`place-plate`	**86.2%**±1.8%	78.7%±2.2%	86.1%±7.7%	**85.8%**±6.6%	78.7%±7.6%
`place-divider-plate`	**89.0%**±2.2%	80.2%±2.2%	85.0%±5.9%	**87.8%**±7.6%	79.2%±6.3%
average	**75.0%**±3.3%	67.3%±0.8%	60.8%±7.5%	**74.8%** ±6.4%	66.4%±7.2%

VICE and **RCE**, but adapt them to the offline setting using CQL, i.e. the same underlying offline RL method as in **UDS** and **CDS+UDS**. For more details on experimental set-up and hyperparameter settings, please see Appendix G. We also include evaluations of our methods under different quality of the relabeled data in Appendix C and comparisons to model-based offline RL approaches in Appendix E and prior methods (Yang & Nachum, 2021; Yang et al., 2021) that leverage unlabeled data for representation learning instead of sharing directly in Appendix F.

Datasets and tasks. To answer questions (1) and (2), we perform empirical evaluations on two state-based single-task locomotion datasets. To answer question (3), we further evaluate all methods on three state-based multi-task datasets that consist of robotic manipulation, navigation and locomotion domains respectively and one image-based multi-task manipulation dataset from prior work (Yu et al., 2021a).

Single-task domains & datasets. We adopt two environments: hopper and the AntMaze from the D4RL benchmark (Fu et al., 2020) for evaluation in the single-task setting. For the hopper environment, we consider two scenarios where we have 10k labeled data from the hopper-expert and 1M unlabeled transitions from the hopper-random datasets and hopper-medium datasets respectively. This setup is resembles practical problems where unlabeled data is of low-quality and even, irrelevant to the target task. For AntMaze, following the setup in (Yang et al., 2021), we use 10k expert transitions as the labeled data and the entire datasets of large-play and medium-play as the unlabeled data.

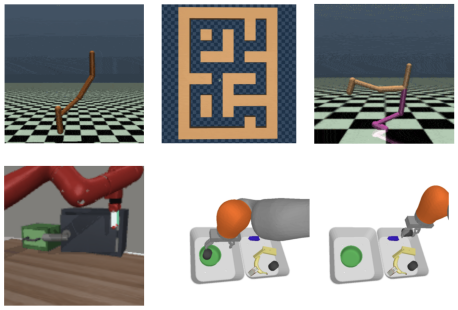

Figure 2. Environments (from left to right): single-task hopper, single-task and multi-task AntMaze, multi-task walker, multi-task Meta-World, and multi-task vision-based pick-place tasks.

Multi-task domains & datasets. We consider several multi-task domains. The first set of domains are from prior work (Yu et al., 2021a): **(i)** the Meta-World (Yu et al., 2020a) domain, which consists of four tasks of opening and closing doors and drawers; and **(ii)** the Antmaze (Fu et al., 2020) domain, which consists of mazes of two sizes (medium and large) with 3 and 7 tasks corresponding to different goal positions respectively. We also evaluate on the multi-task

Table 5. We perform an empirical analysis on the single-task environment `hopper` from D4RL (Fu et al., 2020) benchmark to test the sensitivity of UDS under the data quality and data coverage for both the labeled task data and unlabeled data. The numbers are averaged over ten random seeds. We bold the best method without true reward relabeling. UDS outperforms No Sharing in 5 out of 6 settings while achieving competitive performances compared to Sharing All in 5 out of 6 settings. CDS+UDS is able to further improve over UDS significantly in such a setting and also outperforms UDS in 5 out of 6 settings in general.

Environment	Labeled data type / size	Unlabeled data type / size	CDS+UDS	UDS	No Sharing	Oracle Methods	
						CDS	Sharing All
D4RL hopper	**(a)** expert / 10k transitions	random / 1M transitions	**82.1**	78.8	77.1	83.3	86.1
	(b) expert / 10k transitions	medium / 1M transitions	**78.1**	64.8	77.1	82.5	64.6
	(c) expert / 10k transitions	expert / 990k transitions	106.1	**108.4**	77.1	106.6	112.3
	(d) medium / 10k transitions	random / 1M transitions	**33.2**	9.9	28.7	41.2	38.9
	(e) medium / 10k transitions	expert / 1M transitions	**108.9**	106.7	28.7	111.1	107.5
	(f) random / 10k transitions	medium / 1M transitions	**63.5**	47.1	9.6	92.3	69.8
	(g) random / 10k transitions	expert / 1M transitions	**101.2**	95.9	9.6	110.9	102.8

walker environment with dense rewards, which we will discuss in detail in Appendix D. In addition, we test multi-task offline RL methods with UDS in the multi-task visual manipulation domain, which provides a realistic scenario, of the sorts we will encounter in robotic settings in practice. In this domain, there are 10 tasks with different combinations of object-oriented grasping, with 7 objects (banana, bottle, sausage, milk box, food box, can and carrot), as well as placing the picked objects onto one of three fixtures (bowl, plate and divider plate). For domains except the walker environment, we use binary rewards, where 1 denotes the successful completion of the task and 0 corresponds to failure. We also use the datasets used in (Yu et al., 2021a) for all domains. For details, see Appendix G.2.

5.1 Results of Empirical Evaluations

Results of Question (1) and (2). We evaluate each method on the single-task hopper and AntMaze domains. As shown in Table 2, we find that UDS outperforms No Sharing in 3 out of the 4 settings and reward learning methods in all the settings. We hypothesize that reward learning methods underperform because reward predictors are unable to achieve reasonable generalization ability in the limited labeled data setting. Note that UDS even achieves competitive performance as the oracle Sharing All method. Furthermore, an optimized reweighting scheme, i.e., CDS+UDS, is able to improve over UDS in each case, including cases where UDS fails to improve over No Sharing, indicating a large reward bias. These results testify to the effectiveness of optimizing for reweighting when dealing with unlabeled data. Note that on the AntMaze domain, CDS+UDS performs comparably to the recent approach (Yang et al., 2021) that performs representation learning on the offline dataset first and then run offline training leveraging the learned representation. CDS+UDS also outperforms all the prior methods in offline training with learned representations discussed in (Yang et al., 2021).

Results of Question (3). Observe in Table 3 that UDS outperforms naïve multi-task offline RL without data sharing and reward learning methods, suggesting that leveraging unlabeled data with our simple method UDS can boost offline RL performance in both multi-task manipulation and navigation domains. Since reward learning approaches obtain similar or worse results compared to No Sharing, which we suspect could be due to erroneous predictions on unseen transitions in the multi-task data, we only compare our methods to No Sharing and the oracle methods in the image-based experiments. As shown in Table 4, UDS outperforms No Sharing in 7 out of 10 tasks as well as the average task performance by a significant margin. Therefore, UDS is able to effectively leverage unlabeled data from other tasks and achieves potentially surprisingly strong results compared to more sophisticated methods that handle unlabeled offline data. We also find that CDS+UDS further improves upon the performance of UDS, which indicates that optimizing for reweighting via CDS+UDS can actually work well.

Finally, note that, CDS+UDS and UDS attain performance competitive with their oracle counterparts, CDS and Sharing All, that assume access to full reward information, both in Tables 3 & 4. This is potentially very surprising, and one could hypothesize that this might be because most transitions in the unlabeled dataset were actually failures, and hence, were labeled correctly by UDS. However, to the contrary, our diagnostic analysis in Table 9, Appendix C reveals that the unlabeled data *does not* consist entirely of failures; in fact, around 60% of the unlabeled data succeeds at the task of interest, indicating that the rewards are wrong for more than half the unlabeled data. In spite of this, UDS and CDS+UDS perform well. This indicates that the simple UDS approach can be effective in removing the dependence of functional form of reward functions, which is often costly, without much loss in the performance and CDS+UDS can boost performance by reducing the bias.

5.2 Empirical Analysis of UDS and CDS+UDS

To answer question (4), we analyze UDS and CDS+UDS on several offline RL problems designed to test robustness and sensitivity to the data coverage and the data quality on the single-task hopper domain. To create these instances, we

consider 7 different combinations of data quality (hopper-expert, hopper-medium or hopper-random) and amount of labeled and unlabeled data labeled as **(a)**-**(g)** in Table 5. In Appendix B.1, we present results for a similar analysis in the multi-task Meta-World setting. We evaluate UDS, CDS+UDS and No Sharing in each case, and also present results for CDS and Sharing All approaches which assume access to the actual reward, for understanding. Additionally, we conduct an ablation that compares UDS and CDS+UDS to reward learning methods in settings where the labeled data size and quality are varied, which is included in Appendix B.3.

When the labeled data is of expert-level, adding unlabeled random or medium data (cases **a** and **b** in Table 5) should only increase coverage, since the labeled data only consists of expert transitions. Moreover, the reward bias induced due to incorrect labeling of rewards on the medium unlabeled data should not hurt, since the 10k expert transitions retain their correct labels, and the medium/random data should only serve as negatives, provided the annotated rewards on this data are worse than the rewards in the expert data. Therefore, we expect the benefits of coverage to outweigh any reward bias; as seen in Table 5, we find that UDS indeed helps compared to No Sharing. In particular, in those two cases, UDS approaches the oracle Sharing All method.

Since reward bias is exacerbated when the unlabeled data is of higher quality and present in large amounts (cases **e, f, g**), it is reasonable to surmise that UDS will perform worse in such scenarios. However, on the contrary, in settings: random labeled data with medium or expert unlabeled data and medium labeled data with expert unlabeled data, we find that even though the rewards on the unlabeled transitions are incorrect, the addition of unlabeled data into training improves performance. This is because a higher quality unlabeled data improves the performance of the effective behavior policy, thereby improving performance for a conservative offline RL method. This result validates Theorem 4.1. We also conduct an ablation that varies the amount of unlabeled data in Appendix B.2. This ablation shows that the benefit of UDS reduces as the we reduce the amount of unlabeled data, which confirms our theory in Section 4.1.

In the case where labeled data is medium and unlabeled data is random (case **d**), we find that UDS hurts compared to No Sharing. This is because the labeled data as well as unlabeled data are both low-medium quality and medium data already provides decent coverage (not as high as random data, but not as low as expert data). Therefore, in this case, we believe that the addition of unlabeled data neither provides trajectories of good quality that can help improve performance, nor does it significantly improve coverage, and only hurts by incurring reward bias. We therefore believe that UDS may not help in such cases where the coverage does not improve, and added data is not so high quality, which also agrees with our theoretical analysis. Furthermore, in the case where the labeled and unlabeled data are both expert (case **c**), UDS performs close to oracle methods CDS and Sharing All, which confirms the insights derived from Section 4.1 that UDS does not introduce bias when the labeled and unlabeled data have the same distribution and hence should perform well.

Finally, note that CDS+UDS is able to improve over UDS in 5 out of 6 settings in hopper, suggesting that CDS+UDS gains benefit from reducing the reward bias as well as the sampling error shown in Section 4.

6 Discussion

In this paper, we study the problem of leveraging unlabeled data in offline RL where we find that a simple method UDS that relabels unlabeled data with zero rewards is surprisingly effective in various single-task and multi-task offline RL domains. We provide both theoretical and empirical analysis of UDS to study conditions where it works and does not work. Furthermore, we show that by utilizing the optimized reweighting strategy, reward bias introduced in UDS can be reduced and the policy performance bound is improved in our theoretical analysis, which is also verified empirically. We believe that our analysis justifies the effectiveness of such simple methods for using unlabeled data in offline RL and shed light on directions for future work that can better control the reward bias and enjoy better policy performance.

Acknowledgements

We thank Kanishka Rao, Julian Ibarz, other members of IRIS at Stanford, RAIL at UC Berkeley and Robotics at Google and Google Research and anonymous reviewers for valuable and constructive feedback on an early version of this manuscript. This research was funded in part by Google, ONR grants N00014-21-1-2685 and N00014-21-1-2838, Intel Corporation and the DARPA Assured Autonomy Program. CF is a CIFAR Fellow in the Learning in Machines and Brains program.

References

Abbeel, P. and Ng, A. Y. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the twenty-first international conference on Machine learning*, pp. 1, 2004.

Achiam, J., Held, D., Tamar, A., and Abbeel, P. Constrained policy optimization. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 22–31. JMLR. org, 2017.

Andrychowicz, M., Wolski, F., Ray, A., Schneider, J., Fong, R., Welinder, P., McGrew, B., Tobin, J., Abbeel, P., and Zaremba, W. Hindsight experience replay. *arXiv preprint arXiv:1707.01495*, 2017.

Cabi, S., Colmenarejo, S. G., Novikov, A., Konyushkova, K., Reed, S., Jeong, R., Zolna, K., Aytar, Y., Budden, D., Vecerik, M., et al. Scaling data-driven robotics with reward sketching and batch reinforcement learning. *arXiv preprint arXiv:1909.12200*, 2019.

Chebotar, Y., Hausman, K., Lu, Y., Xiao, T., Kalashnikov, D., Varley, J., Irpan, A., Eysenbach, B., Julian, R., Finn, C., and Levine, S. Actionable models: Unsupervised offline reinforcement learning of robotic skills. *arXiv preprint arXiv:2104.07749*, 2021.

Dasari, S., Ebert, F., Tian, S., Nair, S., Bucher, B., Schmeckpeper, K., Singh, S., Levine, S., and Finn, C. Robonet: Large-scale multi-robot learning, 2020.

de Lima, L. M. and Krohling, R. A. Discovering an aid policy to minimize student evasion using offline reinforcement learning. *arXiv preprint arXiv:2104.10258*, 2021.

Dorfman, R. and Tamar, A. Offline meta reinforcement learning. *arXiv preprint arXiv:2008.02598*, 2020.

Duan, Y., Jia, Z., and Wang, M. Minimax-optimal off-policy evaluation with linear function approximation. In *International Conference on Machine Learning*, pp. 2701–2709. PMLR, 2020.

Ernst, D., Geurts, P., and Wehenkel, L. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research*, 6:503–556, 2005.

Eysenbach, B., Geng, X., Levine, S., and Salakhutdinov, R. Rewriting history with inverse rl: Hindsight inference for policy improvement. *arXiv preprint arXiv:2002.11089*, 2020.

Eysenbach, B., Levine, S., and Salakhutdinov, R. Replacing rewards with examples: Example-based policy search via recursive classification. *arXiv preprint arXiv:2103.12656*, 2021.

Finn, C., Levine, S., and Abbeel, P. Guided cost learning: Deep inverse optimal control via policy optimization. In *International conference on machine learning*, pp. 49–58. PMLR, 2016a.

Finn, C., Tan, X. Y., Duan, Y., Darrell, T., Levine, S., and Abbeel, P. Deep spatial autoencoders for visuomotor learning. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 512–519. IEEE, 2016b.

Fu, J., Luo, K., and Levine, S. Learning robust rewards with adversarial inverse reinforcement learning. *International Conference on Learning Representations*, 2018a.

Fu, J., Singh, A., Ghosh, D., Yang, L., and Levine, S. Variational inverse control with events: A general framework for data-driven reward definition. *Conference on Neural Information Processing Systems*, 2018b.

Fu, J., Singh, A., Ghosh, D., Yang, L., and Levine, S. Variational inverse control with events: A general framework for data-driven reward definition. *arXiv preprint arXiv:1805.11686*, 2018c.

Fu, J., Kumar, A., Nachum, O., Tucker, G., and Levine, S. D4rl: Datasets for deep data-driven reinforcement learning, 2020.

Fujimoto, S. and Gu, S. S. A minimalist approach to offline reinforcement learning. *arXiv preprint arXiv:2106.06860*, 2021.

Fujimoto, S., Meger, D., and Precup, D. Off-policy deep reinforcement learning without exploration. *arXiv preprint arXiv:1812.02900*, 2018.

Ghasemipour, S. K. S., Schuurmans, D., and Gu, S. S. Emaq: Expected-max q-learning operator for simple yet effective offline and online rl. In *International Conference on Machine Learning*, pp. 3682–3691. PMLR, 2021.

Ho, J. and Ermon, S. Generative adversarial imitation learning. *Conference on Neural Information Processing Systems*, 2016.

Jaques, N., Ghandeharioun, A., Shen, J. H., Ferguson, C., Lapedriza, A., Jones, N., Gu, S., and Picard, R. Way off-policy batch deep reinforcement learning of implicit human preferences in dialog. *arXiv preprint arXiv:1907.00456*, 2019.

Kalashnikov, D., Irpan, A., Pastor, P., Ibarz, J., Herzog, A., Jang, E., Quillen, D., Holly, E., Kalakrishnan, M., Vanhoucke, V., et al. Scalable deep reinforcement learning for vision-based robotic manipulation. In *Conference on Robot Learning*, pp. 651–673. PMLR, 2018.

Kalashnikov, D., Varley, J., Chebotar, Y., Swanson, B., Jonschkowski, R., Finn, C., Levine, S., and Hausman, K. Mt-opt: Continuous multi-task robotic reinforcement learning at scale. *Conference on Robot Learning (CoRL)*, 2021.

Kidambi, R., Rajeswaran, A., Netrapalli, P., and Joachims, T. Morel: Model-based offline reinforcement learning. *arXiv preprint arXiv:2005.05951*, 2020.

Konyushkova, K., Zolna, K., Aytar, Y., Novikov, A., Reed, S., Cabi, S., and de Freitas, N. Semi-supervised reward learning for offline reinforcement learning. *arXiv preprint arXiv:2012.06899*, 2020.

Kostrikov, I., Fergus, R., Tompson, J., and Nachum, O. Offline reinforcement learning with fisher divergence critic regularization. In *International Conference on Machine Learning*, pp. 5774–5783. PMLR, 2021.

Kumar, A., Fu, J., Soh, M., Tucker, G., and Levine, S. Stabilizing off-policy q-learning via bootstrapping error reduction. In *Advances in Neural Information Processing Systems*, pp. 11761–11771, 2019.

Kumar, A., Zhou, A., Tucker, G., and Levine, S. Conservative q-learning for offline reinforcement learning. *arXiv preprint arXiv:2006.04779*, 2020.

Lange, S., Gabel, T., and Riedmiller, M. A. Batch reinforcement learning. In *Reinforcement Learning*, volume 12. Springer, 2012.

Laroche, R., Trichelair, P., and Des Combes, R. T. Safe policy improvement with baseline bootstrapping. In *International Conference on Machine Learning*, pp. 3652–3661. PMLR, 2019.

Lee, B.-J., Lee, J., and Kim, K.-E. Representation balancing offline model-based reinforcement learning. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=QpNz8r_Ri2Y.

Levine, S., Kumar, A., Tucker, G., and Fu, J. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

Li, A. C., Pinto, L., and Abbeel, P. Generalized hindsight for reinforcement learning. *arXiv preprint arXiv:2002.11708*, 2020.

Li, J., Vuong, Q., Liu, S., Liu, M., Ciosek, K., Ross, K., Christensen, H. I., and Su, H. Multi-task batch reinforcement learning with metric learning. *arXiv preprint arXiv:1909.11373*, 2019.

Lin, X., Baweja, H. S., and Held, D. Reinforcement learning without ground-truth state. *arXiv preprint arXiv:1905.07866*, 2019.

Liu, H., Trott, A., Socher, R., and Xiong, C. Competitive experience replay. *arXiv preprint arXiv:1902.00528*, 2019.

Liu, Y., Swaminathan, A., Agarwal, A., and Brunskill, E. Provably good batch reinforcement learning without great exploration. *arXiv preprint arXiv:2007.08202*, 2020.

Mitchell, E., Rafailov, R., Peng, X. B., Levine, S., and Finn, C. Offline meta-reinforcement learning with advantage weighting. In *International Conference on Machine Learning*, pp. 7780–7791. PMLR, 2021.

Ng, A. Y. and Russell, S. J. Algorithms for inverse reinforcement learning. In *Proceedings of the Seventeenth International Conference on Machine Learning*, ICML '00, 2000.

Pomerleau, D. A. Alvinn: an autonomous land vehicle in a neural network. In *Proceedings of the 1st International Conference on Neural Information Processing Systems*, pp. 305–313, 1988.

Rafailov, R., Yu, T., Rajeswaran, A., and Finn, C. Offline reinforcement learning from images with latent space models. *Learning for Decision Making and Control (L4DC)*, 2021.

Riedmiller, M. Neural fitted q iteration–first experiences with a data efficient neural reinforcement learning method. In *European Conference on Machine Learning*, pp. 317–328. Springer, 2005.

Ross, S. and Bagnell, D. Agnostic system identification for model-based reinforcement learning. In *ICML*, 2012.

Shortreed, S. M., Laber, E., Lizotte, D. J., Stroup, T. S., Pineau, J., and Murphy, S. A. Informing sequential clinical decision-making through reinforcement learning: an empirical study. *Machine learning*, 84(1-2):109–136, 2011.

Singh, A., Yang, L., Hartikainen, K., Finn, C., and Levine, S. End-to-end robotic reinforcement learning without reward engineering. *arXiv preprint arXiv:1904.07854*, 2019.

Singh, A., Yu, A., Yang, J., Zhang, J., Kumar, A., and Levine, S. Cog: Connecting new skills to past experience with offline reinforcement learning. *arXiv preprint arXiv:2010.14500*, 2020.

Sun, H., Li, Z., Liu, X., Lin, D., and Zhou, B. Policy continuation with hindsight inverse dynamics. *arXiv preprint arXiv:1910.14055*, 2019.

Swazinna, P., Udluft, S., and Runkler, T. Overcoming model bias for robust offline deep reinforcement learning. *arXiv preprint arXiv:2008.05533*, 2020.

Wang, L., Zhang, W., He, X., and Zha, H. Supervised reinforcement learning with recurrent neural network for dynamic treatment recommendation. *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018.

Wu, Y., Tucker, G., and Nachum, O. Behavior regularized offline reinforcement learning. *arXiv preprint arXiv:1911.11361*, 2019.

Xie, A., Singh, A., Levine, S., and Finn, C. Few-shot goal inference for visuomotor learning and planning. In *Conference on Robot Learning*, pp. 40–52. PMLR, 2018.

Xie, A., Ebert, F., Levine, S., and Finn, C. Improvisation through physical understanding: Using novel objects as tools with visual foresight. *Robotics: Science and Systems (RSS)*, 2019.

Yang, M. and Nachum, O. Representation matters: Offline pretraining for sequential decision making. *arXiv preprint arXiv:2102.05815*, 2021.

Yang, M., Levine, S., and Nachum, O. Trail: Near-optimal imitation learning with suboptimal data. *arXiv preprint arXiv:2110.14770*, 2021.

Yu, T., Quillen, D., He, Z., Julian, R., Hausman, K., Finn, C., and Levine, S. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on Robot Learning*, pp. 1094–1100. PMLR, 2020a.

Yu, T., Quillen, D., He, Z., Julian, R., Hausman, K., Finn, C., and Levine, S. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on Robot Learning*, pp. 1094–1100. PMLR, 2020b.

Yu, T., Thomas, G., Yu, L., Ermon, S., Zou, J., Levine, S., Finn, C., and Ma, T. Mopo: Model-based offline policy optimization. *arXiv preprint arXiv:2005.13239*, 2020c.

Yu, T., Kumar, A., Chebotar, Y., Hausman, K., Levine, S., and Finn, C. Conservative data sharing for multi-task offline reinforcement learning. *arXiv preprint arXiv:2109.08128*, 2021a.

Yu, T., Kumar, A., Rafailov, R., Rajeswaran, A., Levine, S., and Finn, C. Combo: Conservative offline model-based policy optimization. *arXiv preprint arXiv:2102.08363*, 2021b.

Zhan, X., Xu, H., Zhang, Y., Huo, Y., Zhu, X., Yin, H., and Zheng, Y. Deepthermal: Combustion optimization for thermal power generating units using offline reinforcement learning. *arXiv preprint arXiv:2102.11492*, 2021.

Zhou, W., Bajracharya, S., and Held, D. Plas: Latent action space for offline reinforcement learning. *arXiv preprint arXiv:2011.07213*, 2020.

Ziebart, B. D., Maas, A. L., Bagnell, J. A., and Dey, A. K. Maximum entropy inverse reinforcement learning. In *Aaai*, volume 8, pp. 1433–1438. Chicago, IL, USA, 2008.

A Proofs for Theoretical Analysis of UDS and Optimized Reweighting Schemes

In this section, we will theoretically analyze UDS and other reweighting schemes to better understand when these approaches can perform well. We will first discuss our notation, then present our theoretical results, then discuss the intuitive explanations of these results, and finally, provide proofs of the theoretical results.

A.1 Notation and Assumptions

Let $\pi_\beta(\mathbf{a}|\mathbf{s})$ denote the behavior policy of the dataset. The dataset, $\mathcal{D}$ is generated from the marginal state-action distribution of π_β, i.e., $\mathcal{D} \sim d^{\pi_\beta}(\mathbf{s})\pi_\beta(\mathbf{a}|\mathbf{s})$. We define $d^\pi_\mathcal{D}$ as the state marginal distribution introduced by the dataset $\mathcal{D}$ under π. For our analysis, we will abstract conservative offline RL algorithms into a generic constrained policy optimization problem (Kumar et al., 2020):

$$\pi^*(\mathbf{a}|\mathbf{s}) := \arg\max_\pi \;\; \widehat{J}_\mathcal{D}(\pi) - \frac{\alpha}{1-\gamma} D(\pi, \pi_\beta). \tag{2}$$

$J_\mathcal{D}(\pi)$ denotes the average return of policy π in the empirical MDP induced by the transitions in the dataset, and $D(\pi, \pi_\beta)$ denotes a divergence measure (e.g., KL-divergence (Jaques et al., 2019; Wu et al., 2019), MMD distance (Kumar et al., 2019) or D_CQL (Kumar et al., 2020)) between the learned policy π and the behavior policy π_β computed in expectation over the marginal state-action distribution induced by the policy in the empirical MDP induced by the dataset:

$$D(\pi, \pi_\beta) = \mathbb{E}_{\mathbf{s}\sim\widehat{d}^\pi_\mathcal{D}} \left[D(\pi(\cdot|\mathbf{s}), \pi_\beta(\cdot|\mathbf{s})] \right.\!.$$

Let $D_\text{CQL}(p, q)$ denote the following distance between two distributions $p(\mathbf{x})$ and $q(\mathbf{x})$ with equal support $\mathcal{X}$:

$$D_\text{CQL}(p, q) := \sum_{\mathbf{x}\in\mathcal{X}} p(\mathbf{x}) \left(\frac{p(\mathbf{x})}{q(\mathbf{x})} - 1 \right).$$

Unless otherwise mentioned, we will drop the subscript "CQL" from D_CQL and use D and D_CQL interchangeably. Prior works (Kumar et al., 2020; Yu et al., 2021a) have shown that the optimal policy π^* that optimizes Equation 2 attains a high probability safe-policy improvement guarantee, i.e., $J(\pi^*) \geq J(\pi_\beta) - \zeta$, where ζ is:

$$\zeta = \mathcal{O}\left(\frac{1}{(1-\gamma)^2} \right) \mathbb{E}_{\mathbf{s}\sim d^{\pi^*}_\mathcal{D}} \left[\sqrt{\frac{D_\text{CQL}(\pi^*, \pi_\beta)(\mathbf{s}) + 1}{|\mathcal{D}(\mathbf{s})|}} \right] - \frac{\alpha}{1-\gamma} D(\pi^*, \pi_\beta). \tag{3}$$

The first term in Equation 3 corresponds to the decrease in performance due to sampling error and this term is high when the single-task optimal policy π^*_i visits rarely observed states in the dataset $\mathcal{D}_i$ and/or when the divergence from the behavior policy π_β is higher under the states visited by the single-task policy $\mathbf{s} \sim d^{\pi^*_i}_{\mathcal{D}_i}$. We will show that UDS and CDS+UDS enjoy safe policy improvement. In our analysis, we assume $r(\mathbf{s}, \mathbf{a}) \in [0, 1]$. Finally, as discussed in Section 3, let $\mathcal{D}^\text{eff}_i$ denote the relabeled dataset for task i, which includes both $\mathcal{D}_i$ and the transitions from other tasks relabeled with a 0 reward.

Assumptions. To prove our theoretical results, following prior work (Kumar et al., 2020; Yu et al., 2021a) we assume that the empirical rewards and dynamics concentrate towards their mean.

Assumption A.1. $\forall\, \mathbf{s}, \mathbf{a}$, the following relationships hold with high probability, $\geq 1 - \delta$

$$|\widehat{r}(\mathbf{s}, \mathbf{a}) - r(\mathbf{s}, \mathbf{a})| \leq \frac{C_{r,\delta}}{\sqrt{|\mathcal{D}(\mathbf{s}, \mathbf{a})|}}, \quad ||\widehat{P}(\mathbf{s}'|\mathbf{s}, \mathbf{a}) - P(\mathbf{s}'|\mathbf{s}, \mathbf{a})||_1 \leq \frac{C_{P,\delta}}{\sqrt{|\mathcal{D}(\mathbf{s}, \mathbf{a})|}}.$$

Similar to prior work (Kumar et al., 2020; Yu et al., 2021a), we also make a coverage assumption, i.e., we assume that each state-action pair is observed in the dataset $\mathcal{D}$, but the rewards and transition dynamics are stochastic, so, the occurrence of each state-action pair does not trivially imply good performance. To relax this assumption, we can extend our analysis to function approximation (e.g., linear function approximation (Duan et al., 2020)), where such a coverage assumption is only required on all directions of the feature space, and not all state-action pairs. This would not significantly change the analysis, and hence we opt for the simple but, at the same time, an illustrative analysis in a tabular setting here.

A.2 Performance Guarantee for UDS: Proof for Theorem 4.1

We first restate Proposition 4.1 below for convenience, and we then provide a proof of the result.

Theorem A.2 (**Policy improvement guarantee for UDS**; restated.)**.** *Let π^*_{UDS} denote the policy learned by UDS, and let $\pi_\beta^{\text{eff}}(\mathbf{a}|\mathbf{s})$ denote the behavior policy for the combined dataset $\mathcal{D}^{\text{eff}}$. Then with high probability $\geq 1-\delta$, π^*_{UDS} is a safe policy improvement over π_β^{eff}, i.e.,*

$$J(\pi^*_{UDS}) \geq J(\pi_\beta^{\text{eff}}) - \zeta_{err} + \underbrace{\frac{\alpha}{1-\gamma} D(\pi^*_{UDS}, \pi_\beta^{\text{eff}})}_{\text{(c): policy improvement}},$$

$$\text{where: } \zeta_{err} = \underbrace{\frac{\sum_{\mathbf{s},\mathbf{a}} \left(\widehat{d}^{\pi_\beta{}^{\text{eff}}}(\mathbf{s},\mathbf{a}) - \widehat{d}^{\pi^*_{UDS}}(\mathbf{s},\mathbf{a})\right) \cdot (1 - f(\mathbf{s},\mathbf{a})) \cdot r(\mathbf{s},\mathbf{a})}{1-\gamma}}_{\text{(a): reward bias}} + \underbrace{\mathcal{O}\left(\frac{\gamma}{(1-\gamma)^2}\right) \left[\sqrt{\frac{D_{CQL}(\pi^*_{UDS}, \pi_\beta^{\text{eff}})(\mathbf{s})}{|\mathcal{D}^{\text{eff}}(\mathbf{s})|}}\right]}_{\text{(b): sampling error}},$$

where we use the notation $f(\mathbf{s},\mathbf{a}) := \frac{|\mathcal{D}_{\text{L}}(\mathbf{s},\mathbf{a})|}{|\mathcal{D}^{\text{eff}}(\mathbf{s},\mathbf{a})|}$.

Proof. We start with the loss decomposition of the improvement of the learned policy relative to the behavior policy with the affine transformation g:

$$J(\pi) - J(\pi_\beta) := \underbrace{J(\pi) - \widehat{J}(\pi)}_{(i)} + \underbrace{\widehat{J}(\pi) - \widehat{J}(\pi_\beta)}_{(ii)} + \underbrace{\widehat{J}(\pi_\beta) - J(\pi_\beta)}_{(iii)}.$$

Now we will discuss how to bound each of the terms: terms (i) and (ii) correspond to the divergence between the empirical policy return and the actual return. While usually, this difference depends on the sampling error and distributional shift, in our case, it additionally depends on the reward bias induced on the unlabeled data and the transformation g. We first discuss the terms that contribute to this reward bias.

Bounding the reward bias. Denote the effective reward of a particular transition $(\mathbf{s},\mathbf{a},r,\mathbf{s}') \in \mathcal{D}^{\text{eff}}$, as $\widehat{r}^{\text{eff}}$, which considers contributions from both the reward $\widehat{r}(\mathbf{s},\mathbf{a})$ observed in dataset $\mathcal{D}_{\text{L}}$, and the contribution of 0 reward from the relabeled dataset:

$$\widehat{r}^{\text{eff}}(\mathbf{s},\mathbf{a}) = \frac{|\mathcal{D}(\mathbf{s},\mathbf{a})| \cdot \widehat{r}(\mathbf{s},\mathbf{a}) + |\mathcal{D}^{\text{eff}}(\mathbf{s},\mathbf{a}) \setminus \mathcal{D}(\mathbf{s},\mathbf{a})| \cdot 0}{|\mathcal{D}^{\text{eff}}(\mathbf{s},\mathbf{a})|} \tag{4}$$

Define $f(\mathbf{s},\mathbf{a}) := \frac{|\mathcal{D}(\mathbf{s},\mathbf{a})|}{|\mathcal{D}^{\text{eff}}(\mathbf{s},\mathbf{a})|}$ for notation compactness. Equation 4 can then be used to derive the following difference against the true rewards:

$$\widehat{r}^{\text{eff}}(\mathbf{s},\mathbf{a}) - r(\mathbf{s},\mathbf{a}) = f(\mathbf{s},\mathbf{a})\left(\widehat{r}(\mathbf{s},\mathbf{a}) - r(\mathbf{s},\mathbf{a})\right) + (1 - f(\mathbf{s},\mathbf{a})) \cdot (0 - r(\mathbf{s},\mathbf{a})) \tag{5}$$

$$\leq f(\mathbf{s},\mathbf{a}) \cdot \frac{C_{r,\delta}}{\sqrt{|\mathcal{D}(\mathbf{s},\mathbf{a})|}} - (1 - f(\mathbf{s},\mathbf{a})) \cdot r(\mathbf{s},\mathbf{a}), \tag{6}$$

where the last step follows from the fact that the ground-truth reward $r(\mathbf{s},\mathbf{a}) \in [0,1]$. Now, we lower bound the reward bias as follows:

$$\widehat{r}^{\text{eff}}(\mathbf{s},\mathbf{a}) + - r(\mathbf{s},\mathbf{a}) = f(\mathbf{s},\mathbf{a}) \cdot (\widehat{r}(\mathbf{s},\mathbf{a}) - r(\mathbf{s},\mathbf{a})) + (1 - f(\mathbf{s},\mathbf{a})) \cdot (-r(\mathbf{s},\mathbf{a})) \tag{7}$$

$$\geq -f(\mathbf{s},\mathbf{a}) \cdot \frac{C_{r,\delta}}{\sqrt{|\mathcal{D}(\mathbf{s},\mathbf{a})|}} - (1 - f(\mathbf{s},\mathbf{a})) \cdot r(\mathbf{s},\mathbf{a}),$$

where the last step follows from the fact that $r(\mathbf{s},\mathbf{a}) \leq 1$.

Upper bounding $\widehat{J}(\pi) - J(\pi)$. Next, using the upper and lower bounds on the reward bias, we now derive an upper bound on the difference between the value of a policy computed under the empirical MDP and the actual MDP. To compute this

difference, we follow the following steps

$$\widehat{J}(\pi) - J(\pi) = \frac{1}{1-\gamma} \sum_{\mathbf{s},\mathbf{a}} \left(\widehat{d}^{\pi}_{\mathcal{D}^{\text{eff}}}(\mathbf{s}) \pi(\mathbf{a}|\mathbf{s}) \widehat{r}^{\text{eff}}(\mathbf{s},\mathbf{a}) - d^{\pi}(\mathbf{s})\pi(\mathbf{a}|\mathbf{s}) r(\mathbf{s},\mathbf{a}) \right) \tag{8}$$

$$= \underbrace{\frac{1}{1-\gamma} \sum_{\mathbf{s},\mathbf{a}} \widehat{d}^{\pi}_{\mathcal{D}^{\text{eff}}}(\mathbf{s}) \pi(\mathbf{a}|\mathbf{s}) \left(\widehat{r}^{\text{eff}}(\mathbf{s},\mathbf{a}) - r(\mathbf{s},\mathbf{a}) \right)}_{:=\Delta_1} + \underbrace{\frac{1}{1-\gamma} \sum_{\mathbf{s},\mathbf{a}} \left(\widehat{d}^{\pi}_{\mathcal{D}^{\text{eff}}}(\mathbf{s}) - d^{\pi}(\mathbf{s}) \right) \pi(\mathbf{a}|\mathbf{s}) r(\mathbf{s},\mathbf{a})}_{:=\Delta_2}$$

Following Kumar et al. (2020) (Theorem 3.6), we can bound the second term Δ_2 using:

$$|\Delta_2| \leq \frac{\gamma C_{P,\delta}}{1-\gamma} \mathbb{E}_{\mathbf{s} \sim \widehat{d}^{\pi}_{\mathcal{D}^{\text{eff}}}(\mathbf{s})} \left[\frac{\sqrt{|\mathcal{A}|}}{\sqrt{|\mathcal{D}^{\text{eff}}(\mathbf{s})|}} \sqrt{D(\pi, \widehat{\pi}^{\text{eff}}_{\beta})(\mathbf{s}) + 1} \right]. \tag{9}$$

To upper bound Δ_1, we utilize the reward upper bound from Equation 5:

$$\Delta_1 = \sum_{\mathbf{s}} \widehat{d}^{\pi}_{\mathcal{D}^{\text{eff}}}(\mathbf{s}) \left(\sum_{\mathbf{a}} \pi(\mathbf{a}|\mathbf{s}) \left(\widehat{r}^{\text{eff}}(\mathbf{s},\mathbf{a}) - r(\mathbf{s},\mathbf{a}) \right) \right) \tag{10}$$

$$\leq \underbrace{\sum_{\mathbf{s}} \widehat{d}^{\pi}_{\mathcal{D}^{\text{eff}}}(\mathbf{s}) \sum_{\mathbf{a}} f(\mathbf{s},\mathbf{a}) \frac{C_{r,\delta}}{\sqrt{|\mathcal{D}(\mathbf{s})|}} \frac{\pi(\mathbf{a}|\mathbf{s})}{\sqrt{\hat{\pi}_{\beta}(\mathbf{a}|\mathbf{s})}}}_{=\Delta_1'} - \underbrace{\sum_{\mathbf{s},\mathbf{a}} \widehat{d}^{\pi}_{\mathcal{D}^{\text{eff}}}(\mathbf{s}) \pi(\mathbf{a}|\mathbf{s}) \left[(1 - f(\mathbf{s},\mathbf{a})) \cdot r(\mathbf{s},\mathbf{a}) \right]}_{:=\Delta_4}. \tag{11}$$

Combining the results so far, we obtain, for any policy π:

$$J(\pi) \geq \widehat{J}(\pi) - \frac{|\Delta_2|}{1-\gamma} - \frac{|\Delta_1'|}{1-\gamma} + \frac{\Delta_4}{1-\gamma}. \tag{12}$$

Lower bounding $\widehat{J}(\pi) - J(\pi)$. To lower bound this quantity, we follow the step shown in Equation 8, and lower bound the term Δ_2 by using the negative of the RHS of Equation 9, and lower bound Δ_1 by upper bounding its absolute value as shown below:

$$\Delta_1 = \sum_{\mathbf{s}} \widehat{d}^{\pi}_{\mathcal{D}^{\text{eff}}}(\mathbf{s}) \left(\sum_{\mathbf{a}} \pi(\mathbf{a}|\mathbf{s}) \left(\widehat{r}^{\text{eff}}(\mathbf{s},\mathbf{a}) - r(\mathbf{s},\mathbf{a}) \right) \right) \tag{13}$$

$$\geq \underbrace{\sum_{\mathbf{s}} \widehat{d}^{\pi}_{\mathcal{D}^{\text{eff}}}(\mathbf{s}) \sum_{\mathbf{a}} f(\mathbf{s},\mathbf{a}) \frac{C_{r,\delta}}{\sqrt{|\mathcal{D}(\mathbf{s})|}} \frac{\pi(\mathbf{a}|\mathbf{s})}{\sqrt{\hat{\pi}_{\beta}(\mathbf{a}|\mathbf{s})}}}_{=\Delta_1'} + \sum_{\mathbf{s}} \widehat{d}^{\pi}_{\mathcal{D}^{\text{eff}}}(\mathbf{s}) \sum_{\mathbf{a}} \pi(\mathbf{a}|\mathbf{s}) \cdot (1 - f(\mathbf{s},\mathbf{a})) r(\mathbf{s},\mathbf{a}). \tag{14}$$

This gives rise to the complete lower bound:

$$\widehat{J}(\pi) \geq J(\pi) - \frac{|\Delta_2|}{1-\gamma} - \frac{1}{1-\gamma} \sum_{\mathbf{s},\mathbf{a}} \widehat{d}^{\pi}_{\mathcal{D}_i^{\text{eff}}}(\mathbf{s}) \pi(\mathbf{a}|\mathbf{s}) (1 - f(\mathbf{s},\mathbf{a})) r(\mathbf{s},\mathbf{a}) - \frac{\Delta_1'}{1-\gamma}. \tag{15}$$

Policy improvement term (ii). Finally, the missing piece that needs to be bounded is the policy improvement term (ii) in the decomposition of $J(\pi) - J(\pi_\beta)$. Utilizing the abstract form of offline RL (Equation 2, we note that term (ii) is lower bounded as:

$$\text{term (ii)} \geq \frac{\alpha}{1-\gamma} D(\pi, \pi_\beta). \tag{16}$$

Putting it all together. To obtain the final expression of Proposition 4.1, we put all the parts together, and include some simplifications to obtain the final expression. The bound we show is relative to the effective behavior policy π^{eff}_{β}. Applying

Equation 15 for term (i) on policy π, Equation 16 for term (ii), and Equation 12 for the behavior policy π_β^{eff}, we obtain the following:

$$\begin{aligned}
J(\pi) - J(\pi_\beta^{\text{eff}}) &= J(\pi) - \widehat{J}(\pi) + \widehat{J}(\pi) - \widehat{J}(\pi_\beta^{\text{eff}}) + \widehat{J}(\pi_\beta^{\text{eff}}) - J(\pi_\beta^{\text{eff}}) \\
&\geq -\frac{2\gamma C_{P,\delta}}{(1-\gamma)^2}\mathbb{E}_{\mathbf{s}\sim\widehat{d}^{\pi}_{\mathcal{D}^{\text{eff}}}(\mathbf{s})}\left[\frac{\sqrt{|\mathcal{A}|}}{\sqrt{|\mathcal{D}^{\text{eff}}(\mathbf{s})|}}\sqrt{D(\pi,\widehat{\pi}_\beta^{\text{eff}})(\mathbf{s})+1}\right] - \frac{2C_{r,\delta}}{1-\gamma}\mathbb{E}_{\mathbf{s},\mathbf{a}\sim\widehat{d}^{\pi}_{\mathcal{D}^{\text{eff}}}}\left[\frac{f(\mathbf{s},\mathbf{a})}{\sqrt{|\mathcal{D}(\mathbf{s},\mathbf{a})|}}\right] \\
&\quad - \frac{1}{1-\gamma}\left(\mathbb{E}_{\mathbf{s},\mathbf{a}\sim d^{\pi_\beta^{\text{eff}}}_{\mathcal{D}^{\text{eff}}}}\left[(1-f(\mathbf{s},\mathbf{a}))\, r(\mathbf{s},\mathbf{a})\right]\right) + \frac{1}{1-\gamma}\mathbb{E}_{\mathbf{s},\mathbf{a}\sim\widehat{d}^{\pi}_{\mathcal{D}^{\text{eff}}}}\left[(1-f(\mathbf{s},\mathbf{a}))\cdot r(\mathbf{s},\mathbf{a})\right] \\
&\quad + \frac{\alpha}{1-\gamma}D(\pi,\pi_\beta^{\text{eff}}).
\end{aligned}$$

Note that in the second step above, we upper bound the quantities Δ_1' and Δ_2 corresponding to π_β^{eff} with twice the expression for policy π. This is because the effective behavior policy π_β^{eff} consists of a mixture of the original behavior policy $\hat{\pi}_\beta$ with the additional data, and thus the new effective dataset consists of the original dataset $\mathcal{D}_i$ as its part. Upper bounding it with twice the corresponding term for π is a valid bound, though a bit looser, but this bound suffices for our interpretations.

For our analysis purposes, we will define the suboptimality induced in the bound due to reward bias for a given u and v as:

$$\text{RewardBias}(\pi,\pi_\beta^{\text{eff}}) = -\frac{1}{1-\gamma}\left[\mathbb{E}_{\mathbf{s},\mathbf{a}\sim\widehat{d}^{\pi}_{\mathcal{D}^{\text{eff}}}}\left[(1-f(\mathbf{s},\mathbf{a}))\cdot r(\mathbf{s},\mathbf{a})\right] - \left(\mathbb{E}_{\mathbf{s},\mathbf{a}\sim d^{\pi_\beta^{\text{eff}}}_{\mathcal{D}^{\text{eff}}}}\left[(1-f(\mathbf{s},\mathbf{a}))\, r(\mathbf{s},\mathbf{a})\right]\right)\right] \tag{17}$$

Thus, we obtain the desired bound in Proposition 4.1. □

A.3 When is Reward Bias Small? Proof of Theorem 4.2

Next, we wish to understand when the reward bias in Equation 17 is small. Concretely, we wish to search for effective behavior policies such that the dataset induced by them attains a small reward bias. Therefore we provide a proof for Theorem 4.2 in this section.

Proof. We can express the reward bias as:

$$\text{RewardBias}(\pi,\pi_\beta^{\text{eff}}) := -\frac{1}{1-\gamma}\sum_{\mathbf{s},\mathbf{a}}\left(\widehat{d}^{\pi}_{\mathcal{D}^{\text{eff}}}(\mathbf{s},\mathbf{a}) - \widehat{d}^{\pi_{\beta^{\text{eff}}}}_{\mathcal{D}^{\text{eff}}}(\mathbf{s},\mathbf{a})\right)\cdot(1-f(\mathbf{s},\mathbf{a}))\cdot r(\mathbf{s},\mathbf{a}),$$

Now, since our goal is to minimize the reward bias with respect to the effective behavior policy, we minimize the expression for suboptimality induced due to reward bias, shown above with respect to π_β^{eff}. Before performing the differentiation step, we note the following simplification (we drop the $\mathcal{D}^{\text{eff}}$ from the notation in $d^{\pi_{\beta^{\text{eff}}}}_{\mathcal{D}^{\text{eff}}}$ to make the notation less cluttered below):

$$\begin{aligned}
\min_{\pi_\beta^{\text{eff}}}\ \text{RewardBias}(\pi,\pi_\beta^{\text{eff}}) &:= \min_{\pi_\beta^{\text{eff}}}\ -\left(\widehat{J}(\pi) - \widehat{J}(\pi_\beta^{\text{eff}})\right) + \frac{1}{(1-\gamma)}\sum_{\mathbf{s},\mathbf{a}} r(\mathbf{s},\mathbf{a})f(\mathbf{s},\mathbf{a})\left(\widehat{d}^{\pi}_{\mathcal{D}^{\text{eff}}}(\mathbf{s},\mathbf{a}) - d^{\pi_{\beta^{\text{eff}}}}_{\mathcal{D}^{\text{eff}}}(\mathbf{s},\mathbf{a})\right) \\
&= \min_{\pi_\beta^{\text{eff}}}\ -\widehat{J}(\pi) + \widehat{J}(\pi_\beta^{\text{eff}}) + \frac{1}{(1-\gamma)|\mathcal{D}^{\text{eff}}|}\sum_{\mathbf{s},\mathbf{a}}|\mathcal{D}(\mathbf{s},\mathbf{a})|r(\mathbf{s},\mathbf{a})\left(\frac{\widehat{d}^{\pi}(\mathbf{s},\mathbf{a})}{\widehat{d}^{\pi_\beta^{\text{eff}}}(\mathbf{s},\mathbf{a})} - 1\right) \\
&= \min_{\pi_\beta^{\text{eff}}}\ \widehat{J}(\pi_\beta^{\text{eff}}) + \frac{1}{(1-\gamma)|\mathcal{D}^{\text{eff}}|}\sum_{\mathbf{s},\mathbf{a}}|\mathcal{D}(\mathbf{s},\mathbf{a})|r(\mathbf{s},\mathbf{a})\left(\frac{\widehat{d}^{\pi}(\mathbf{s},\mathbf{a})}{\widehat{d}^{\pi_\beta^{\text{eff}}}(\mathbf{s},\mathbf{a})} - 1\right).
\end{aligned} \tag{18}$$

Now, since we can express the entire objective in Equation 18 as a function of $\widehat{d}^{\pi_{\beta^{\text{eff}}}}$ since $\widehat{J}(\pi_\beta^{\text{eff}}) = \sum_{\mathbf{s},\mathbf{a}}\widehat{d}^{\pi_{\beta^{\text{eff}}}}(\mathbf{s},\mathbf{a})r(\mathbf{s},\mathbf{a})$, we can compute and set the derivative of Equation 18 with respect to $\widehat{d}^{\pi_{\beta^{\text{eff}}}}(\mathbf{s},\mathbf{a})$ as 0 while adding down the constraints that pertain to the validity of π. This gives us:

$$\widehat{d}^{\pi}(\mathbf{s},\mathbf{a}) \propto \sqrt{\frac{|\mathcal{D}(\mathbf{s},\mathbf{a})|\cdot\widehat{d}^{\pi}(\mathbf{s},\mathbf{a})}{|\mathcal{D}^{\text{eff}}|}}.$$

Substituting $|\mathcal{D}(\mathbf{s},\mathbf{a})| = \widehat{d}_{\text{L}}(\mathbf{s},\mathbf{a})\cdot|\mathcal{D}_{\text{L}}|$ we obtain the desired result. □

A.4 When is The Bound in Theorem 4.1 Tightest?: Proof of Theorem 4.3

In this section, we will formally state and provide a proof for Theorem 4.3. To prove this result, we first compute an upper bound on the error terms (a) and (b) in Theorem 4.3, and show that the optimized distribution shown in Theorem 4.3 emerges as a direct consequence of optimizing this bound. To begin, we compute a different upper bound on sampling error than the one used in Theorem 4.3. Defining the sampling error for a policy π as the difference in return in the original and the empirical MDPs $\Delta_{\text{sampling}} = \widehat{J}(\pi) - J(\pi)$, we obtain the following Lemma:

Lemma A.3 (Upper bound on sampling error in terms of $\widehat{d}^{\pi}(\mathbf{s}, \mathbf{a})$)**.** *We can upper bound sampling error term as follows:*

$$\Delta_{\textit{sampling}} \leq \frac{\gamma C_{P,\delta}}{(1-\gamma)^2\sqrt{|\mathcal{D}|}} \sum_{\mathbf{s},\mathbf{a}} \frac{\widehat{d}^{\pi}(\mathbf{s}, \mathbf{a})}{\sqrt{\widehat{d}^{\pi^{\text{eff}}_{\beta}}(\mathbf{s}, \mathbf{a})}}. \tag{19}$$

Proof. For this proof, we will derive a bound on the sampling error, starting from scratch, but this time only in terms of the state-action marginals:

$$\Delta_{\text{sampling}} = \frac{1}{1-\gamma} \sum_{\mathbf{s},\mathbf{a}} \left(\widehat{d}^{\pi}(\mathbf{s}, \mathbf{a}) - d^{\pi}(\mathbf{s}, \mathbf{a}) \right) \cdot r(\mathbf{s}, \mathbf{a})$$

Note that we can bound Δ_{sampling} by upper bounding the total variation between marginal state-action distributions in the empirical and actual MDPs, i.e., $||\widehat{d}^{\pi} - d^{\pi}||_1$, since $|r_M(\mathbf{s}, \mathbf{a})| \leq R_{\max}$. and hence we bound the second term effectively. Our analysis is similar to Achiam et al. (2017). Define, $G = (I - \gamma P^{\pi})^{-1}$ and $\bar{G} = (I - \gamma \widehat{P}^{\pi})^{-1}$. Then,

$$\widehat{d}^{\pi} - d^{\pi} = (1-\gamma)(\bar{G} - G)\rho,$$

where $\rho(\mathbf{s})$ is the initial state distribution. Then we can use the derivation in proof of Theorem 3.6 in Kumar et al. (2020) or Equation 21 from Achiam et al. (2017), to bound this difference as

$$\begin{aligned} ||d^{\pi} - \widehat{d}^{\pi}||_1 &\leq \frac{\gamma}{1-\gamma} \sum_{\mathbf{s}} \widehat{d}^{\pi}(\mathbf{s}) \frac{C_{P,\delta}}{\sqrt{|\mathcal{D}(\mathbf{s})|}} \sum_{\mathbf{a}} \frac{\pi(\mathbf{a}|\mathbf{s})}{\sqrt{\pi_{\beta}(\mathbf{a}|\mathbf{s})}} \\ &\leq \frac{\gamma C_{P,\delta}}{(1-\gamma)\sqrt{|\mathcal{D}|}} \sum_{\mathbf{s},\mathbf{a}} \frac{\widehat{d}^{\pi}(\mathbf{s}, \mathbf{a})}{\sqrt{\widehat{d}^{\pi^{\text{eff}}_{\beta}}(\mathbf{s}, \mathbf{a})}}. \end{aligned}$$

Thus the sampling error can be bounded by:

$$\Delta_{\text{sampling}} \leq \frac{\gamma C_{P,\delta}}{(1-\gamma)^2\sqrt{|\mathcal{D}|}} \sum_{\mathbf{s},\mathbf{a}} \frac{\widehat{d}^{\pi}(\mathbf{s}, \mathbf{a})}{\sqrt{\widehat{d}^{\pi^{\text{eff}}_{\beta}}(\mathbf{s}, \mathbf{a})}},$$

which proves the lemma. □

Theorem A.4 (Optimized reweighting unlabeled data). *The optimal effective behavior policy that maximizes a lower bound on $J(\pi^{\text{eff}}_{\beta}) - [(a) + (b)]$ in Theorem 4.1 satisfies: $d^{\widehat{\pi}^{\text{eff}}_{\beta}}(\mathbf{s}, \mathbf{a}) = p^*(\mathbf{s}, \mathbf{a})$, where,*

$$p^* = \arg \min_{p \in \Delta^{|\mathcal{S}||\mathcal{A}|}} \sum_{\mathbf{s},\mathbf{a}} C_1 \frac{\widehat{d}^{\pi}(\mathbf{s}, \mathbf{a})}{\sqrt{p(\mathbf{s}, \mathbf{a})}} + C_2 |d_{\text{L}}(\mathbf{s}, \mathbf{a})| \frac{\widehat{d}^{\pi}(\mathbf{s}, \mathbf{a})}{p(\mathbf{s}, \mathbf{a})},$$

where C_1 and C_2 are universal positive constants that depend on the sizes of the labeled and unlabeled datasets are shown in Equation 20.

Proof. To prove this result, we will first simplify the expression containing all terms except the policy imnprovement term in Theorem 4.3, so that we can then maximize the bound to obtain the statement of our theoretical statement.

$$\begin{aligned} (\bullet) := J(\pi^{\text{eff}}_{\beta}) - [(a) + (b)] \,. &\geq J(\pi^{\text{eff}}_{\beta}) + \frac{1}{1-\gamma} \sum_{\mathbf{s},\mathbf{a}} \left(\widehat{d}^{\pi}(\mathbf{s}, \mathbf{a}) - \widehat{d}^{\pi_{\beta}{}^{\text{eff}}}(\mathbf{s}, \mathbf{a}) \right) \cdot (1 - f(\mathbf{s}, \mathbf{a})) \cdot r(\mathbf{s}, \mathbf{a}) - \Delta_{\text{sampling}} \\ &\geq J(\pi^{\text{eff}}_{\beta}) + \frac{1}{1-\gamma} \sum_{\mathbf{s},\mathbf{a}} \left(\widehat{d}^{\pi}(\mathbf{s}, \mathbf{a}) - \widehat{d}^{\pi_{\beta}{}^{\text{eff}}}(\mathbf{s}, \mathbf{a}) \right) \cdot (1 - f(\mathbf{s}, \mathbf{a})) \cdot r(\mathbf{s}, \mathbf{a}) - \frac{\gamma C_{P,\delta}}{(1-\gamma)^2\sqrt{|\mathcal{D}_{\text{eff}}|}} \sum_{\mathbf{s},\mathbf{a}} \frac{\widehat{d}^{\pi}(\mathbf{s}, \mathbf{a})}{\sqrt{\widehat{d}^{\pi^{\text{eff}}_{\beta}}(\mathbf{s}, \mathbf{a})}} \end{aligned}$$

First of all we can lower bound, $J(\pi_\beta^{\text{eff}})$ in terms of the return of $\pi_\beta^{\text{eff}})$ in the empirical MDP induced by the dataset $\mathcal{D}_{\text{eff}}$ and an irreducible sampling error term, that grows as $\mathcal{O}\left(\sqrt{1/|\mathcal{D}^{\text{eff}}|}\right)$ and does not depend on the distribution of state-action pairs in the effective dataset, $\widehat{d}^{\pi_\beta^{\text{eff}}}(\mathbf{s}, \mathbf{a})$, but only depends on its size. Using this result, and by performing algebraic manipulation, we can further simplify this as:

$$
\begin{aligned}
(\bullet) &\geq \cancel{\widehat{J}(\pi_\beta^{\text{eff}})} + \widehat{J}(\pi) - \cancel{\widehat{J}(\pi_\beta^{\text{eff}})} - \frac{\sum_{\mathbf{s},\mathbf{a}} |\mathcal{D}(\mathbf{s},\mathbf{a})| r(\mathbf{s},\mathbf{a}) \left(\frac{\widehat{d}^{\pi}(\mathbf{s},\mathbf{a})}{\widehat{d}^{\pi_\beta^{\text{eff}}}(\mathbf{s},\mathbf{a})} - 1 \right)}{(1-\gamma)|\mathcal{D}^{\text{eff}}|} - \frac{\gamma C_{P,\delta}}{(1-\gamma)^2 \sqrt{|\mathcal{D}_{\text{eff}}|}} \sum_{\mathbf{s},\mathbf{a}} \frac{\widehat{d}^{\pi}(\mathbf{s},\mathbf{a})}{\sqrt{\widehat{d}^{\pi_\beta^{\text{eff}}}(\mathbf{s},\mathbf{a})}} \\
&\quad + \mathcal{O}\left(\sqrt{\frac{1}{|\mathcal{D}^{\text{eff}}|}} \right) \\
&\geq \widehat{J}(\pi) - \frac{1}{(1-\gamma)|\mathcal{D}^{\text{eff}}|} \sum_{\mathbf{s},\mathbf{a}} |\mathcal{D}(\mathbf{s},\mathbf{a})| r(\mathbf{s},\mathbf{a}) \left(\frac{\widehat{d}^{\pi}(\mathbf{s},\mathbf{a})}{\widehat{d}^{\pi_\beta^{\text{eff}}}(\mathbf{s},\mathbf{a})} - 1 \right) - \frac{\gamma C_{P,\delta}}{(1-\gamma)^2 \sqrt{|\mathcal{D}_{\text{eff}}|}} \sum_{\mathbf{s},\mathbf{a}} \frac{\widehat{d}^{\pi}(\mathbf{s},\mathbf{a})}{\sqrt{\widehat{d}^{\pi_\beta^{\text{eff}}}(\mathbf{s},\mathbf{a})}} \\
&\quad + \mathcal{O}\left(\sqrt{\frac{1}{|\mathcal{D}^{\text{eff}}|}} \right) \\
&\geq \widehat{J}(\pi) - \frac{1}{(1-\gamma)|\mathcal{D}^{\text{eff}}|} \sum_{\mathbf{s},\mathbf{a}} |\mathcal{D}(\mathbf{s},\mathbf{a})| \left(\frac{\widehat{d}^{\pi}(\mathbf{s},\mathbf{a})}{\widehat{d}^{\pi_\beta^{\text{eff}}}(\mathbf{s},\mathbf{a})} - 1 \right) - \frac{\gamma C_{P,\delta}}{(1-\gamma)^2 \sqrt{|\mathcal{D}_{\text{eff}}|}} \sum_{\mathbf{s},\mathbf{a}} \frac{\widehat{d}^{\pi}(\mathbf{s},\mathbf{a})}{\sqrt{\widehat{d}^{\pi_\beta^{\text{eff}}}(\mathbf{s},\mathbf{a})}} + \mathcal{O}\left(\sqrt{\frac{1}{|\mathcal{D}^{\text{eff}}|}} \right),
\end{aligned}
$$

where the last inequality follows from the fact that $|r(\mathbf{s},\mathbf{a})| \leq 1$. Since $|\mathcal{D}^{\text{eff}}|$ and $\widehat{d}^{\pi_\beta^{\text{eff}}}$ are decoupled (one is the distribution; other is the size of the dataset), we can optimize over each of them independently, and hence, we find that the distribution that optimizes this bound is given by:

$$
p^* = \arg \min_{p \in \Delta^{|\mathcal{S}||\mathcal{A}|}} \sum_{\mathbf{s},\mathbf{a}} C_1 \frac{\widehat{d}^{\pi}(\mathbf{s},\mathbf{a})}{\sqrt{p(\mathbf{s},\mathbf{a})}} + C_2 |d_{\text{L}}(\mathbf{s},\mathbf{a})| \frac{\widehat{d}^{\pi}(\mathbf{s},\mathbf{a})}{p(\mathbf{s},\mathbf{a})},
$$

where:

$$
C_2 := \frac{|\mathcal{D}_{\text{L}}|}{(1-\gamma)|\mathcal{D}_{\text{eff}}|}, \qquad C_1 := \frac{\gamma C_{P,\delta}}{(1-\gamma)^2 \sqrt{|\mathcal{D}_{\text{eff}}|}}. \tag{20}
$$

This proves Theorem 4.3. □

B Additional empirical analysis of the reason that CDS+UDS and UDS work

In this section, we perform an empirical study on the Meta-World domain to better understand the reason that UDS and CDS+UDS work well. Our theoretical analysis suggests that UDS will help the most on domains with limited data or narrow coverage or low data quality. To test these conditions in practice, we perform empirical analysis on two domains as follows.

B.1 Meta-World Domains

We first choose the `door open` task with three different combinations of dataset size and data quality of the task-specific data with reward labels:

- 2k transitions with the expert-level performance (i.e. **high-quality data with limited sample size and narrow coverage**)
- 2k transitions with medium-level performance (i.e. **medium-quality data with limited sample size and narrow coverage**)
- a medium-replay dataset with 152k transitions (i.e. **medium-quality data with sufficient sample size and broad coverage**).

Environment	Dataset type / size	CDS+UDS	UDS	No Sharing
Meta-World door open	expert / 2k transitions	**67.6%**	58.8%	31.3%
	medium / 2k transitions	67.3%	**74.2%**	27.6%
	medium-replay / 152k transitions	**30.0%**	0.0%	14.8%

Table 6. We perform an empirical analysis on the Meta-World `door open` task where we use varying data quality and dataset size target task `door open`. We share the same dataset from the other three tasks in the multi-task Meta-World environment, `door close`, `drawer open` and `drawer close` to the target task. The numbers are averaged over three random seeds. CDS+UDS and UDS are able to outperform No Sharing in most of the settings except that UDS fails to achieve non-zero success rate in the medium-replay dataset with a large number of transitions. Such results suggest that CDS+UDS and UDS are robust to the data quality of the target task and work the best in settings where the target task has limited data.

Environment	Labeled dataset type / size	Unlabeled dataset type / size	CDS+UDS	UDS	Sharing All (oracle)
D4RL hopper	random / 10k transitions	expert / 10k transitions	**10.1**	10.0	71.9
	random / 10k transitions	expert / 100k transitions	**105.8**	81.8	96.3
	random / 10k transitions	expert / 1M transitions	**102.3**	97.0	102.8

Table 7. Ablation study on the unlabeled dataset size ranging from 10k to 1M transitions in the single-task hopper domain. We bold the best method without true reward relabeling.

We share the same data from the other three tasks, `door close`, `drawer open` and `drawer close` as in Table 3, which are . As shown in Table 6, both UDS and CDS+UDS are able to outperform No Sharing in the three settings, suggesting that increasing the coverage of the offline data as suggested by our theory does lead to performance boost in wherever we have limited good-quality data (expert), limited medium-quality data (medium) and abundant medium-quality data (medium-replay). It's worth noting that UDS and CDS+UDS significantly outperform No Sharing in the limited expert and medium data setting whereas in the medium-replay setting with broader coverage, CDS+UDS outperforms No sharing but UDS fails to achieve non-zero success rate. Such results suggest that UDS and CDS+UDS can yield greater benefit when the target task doesn't have sufficient data and the number of relabeled data is large. The fact that UDS is unable to learn on medium-replay datasets also suggests that data sharing without rewards is less useful in settings where the coverage of the labeled offline data is already quite broad.

B.2 D4RL Hopper Diagnostic Study on Varying Unlabeled Dataset Size

Following the discussion in Section 5.2, we further study the setting where relabeled data has higher quality than the labeled data in the single-task hopper task by varying the amount of unlabeled data within the range of (10k, 100k, 1M) transitions. We pick the case where labeled data is random and unlabeled data is expert for such ablation study. As shown in Table 7, as the unlabeled (expert) dataset size decreases, the result of UDS drops significantly whereas Sharing All retains a reasonable level of performance. This ablation suggests that as the effective dataset size decreases, the benefit of reducing sampling error is reduced and no longer able to outweigh the reward bias as indicated in our theoretical analysis. Moreover, Table 7 also suggests that, in the setting with medium unlabeled dataset size (100k transitions), CDS+UDS is able to prevent the performance drop as seen in UDS and even outperforms Sharing All. However, CDS+UDS cannot successfully tackle the case where there are only 10k unlabeled transitions, suggesting that the reward bias optimized by CDS+UDS is still detrimental in the limited unlabeled data regime.

B.3 D4RL Hopper Ablation Study on Reward Learning Methods with Varying Labeled Dataset Size and Quality

We performed an ablation that varies the labeled data size (10k & 20k transitions) and quality (expert / random) for reward learning methods on D4RL hopper, with unlabeled data being 1M `hopper-medium` transitions. Observe on the right, as expected, reward learning performs better with more labeled data. Furthermore, while reward learning works well when labeled data is high quality, it fails to perform well when labeled data is of low quality, potentially due to the bias in reward prediction. UDS and CDS+UDS are less sensitive to labeled data quality/size.

Env	Labeled dataset type / size	Unlabeled dataset type / size	CDS+UDS	UDS	Reward Pred.
hopper	expert / 10k transitions	medium / 1M transitions	**78.3**± 5.4	64.4±11.7	51.7 ±8.4
	expert / 20k transitions	medium / 1M transitions	95.5±5.4	**99.2**±3.1	96.7±3.6
	random / 10k transitions	medium / 1M transitions	**65.8**±11.3	51.9±2.4	33.4±5.4
	random / 20k transitions	medium / 1M transitions	**69.1**±4.8	59.9±5.6	47.5±5.9

Table 8. Ablation study on comparisons between CDS+UDS/UDS and Reward Predictor with varying labeled dataset size and quality in the single-task hopper domain. We bold the best method.

Environment	Tasks	Oracle Success Rate of the Shared data
Meta-World	drawer open	47.4%
	door close	99.2%
	drawer open	0.1%
	drawer close	91.6%%
	average	59.5%
AntMaze	medium maze (3 tasks) average	4.3%
	large maze (7 tasks) average	1.6%

Table 9. Success rate of the data shared from other tasks to the target task determined by the ground-truth multi-task reward function.

B.4 Takeaways from the empirical analysis

Given our empirical analysis in Table 5, Table 6, Table 7 and Table 8, we summarize the applicability of UDS / CDS+UDS under different scenarios for practitioners in Figure 3 below.

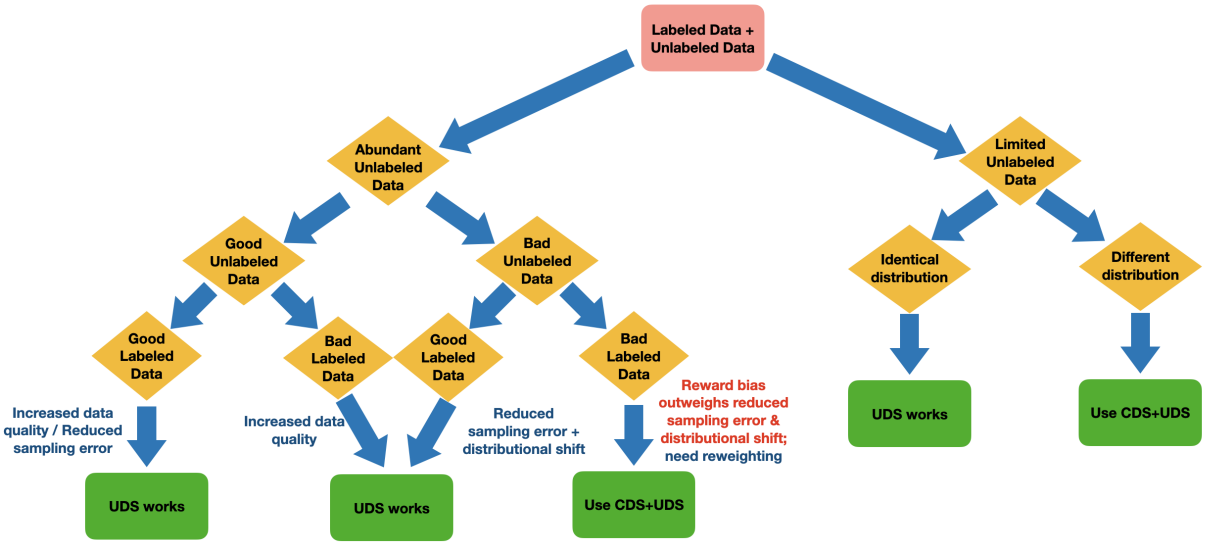

Figure 3. A tree plot that illustrates under which conditions a practitioner should run UDS over apply the optimized reweighting scheme on top of UDS.

C Additional details on the quality of data shared from other tasks in the multi-task offline RL setting

We present the success rate of the data shared from other tasks to the target task computed by the oracle multi-task reward function in both the multi-task Meta-World and AntMaze domains in Table 9. Note that the success rate of `drawer close` and `door close` are particularly high since for other tasks, the drawer / door is initialized to be closed and therefore the success rate of other task data for these two tasks are almost 100% as defined by the success condition in the public Meta-World repo. Apart from these two particularly high success rates, the success rates of the shared data are consistently above 0% across all tasks in both domains. This fact suggests that UDS and CDS+UDS are *not* relabeling with the ground truth reward where the relabeled data are actually all failures but rather performs the conservative bellman backups on relabeled data that is shown to be effective empirically.

Environment	Tasks	UDS	UDS-5% relabel success	UDS-50% relabel success	UDS-90% relabel success
Meta-World	drawer open	51.9%±25.3	0.0%±0.0%	57.3%±18.9%	73.3%±8.6%
	door close	12.3%±27.6%	0.0%±0.0%	0.0%±0.0%	0.0%±0.0%
	drawer open	61.8%±16.3%	19.4%±27.3%	61.0%±12.7%	56.3%±20.3%
	drawer close	99.6%±0.7%	66.0%±46.7%	99.7%±0.5%	100.0%±0.0%
	average	56.4%±12.8%	21.4% ±16.1%	54.3% ±2.0%	57.4%±3.3%

Table 10. Performance of UDS under different actual success rates of the relabeled data.

Environment	Tasks / Dataset type	CDS+UDS	UDS	No Sharing	Reward Predictor	CDS (oracle)	Sharing All (oracle)
Multi-Task Walker	run forward / medium-replay	**880.1**±108.8	665.0±84.9	590.1±48.6	520.7±373.6	1057.9±121.6	701.4±47.0
	run backward / medium	**717.8**±78.3	689.3±16.3	614.7±87.3	417.3±235.3	564.8±47.7	756.7±76.7
	jump / expert	1487.7±177.6	1036.0±247.1	**1575.2**±70.9	583.0±432.0	1418.2±138.4	885.1±152.9
	average	**1028.6**±76.8	796.7±106.3	926.6±37.7	506.7 ± 343.6	1013.6±71.5	781.0±100.8

Table 11. Results for multi-task walker experiment with dense rewards. We only bold the best-performing method that does not have access to the true reward during relabeling. CDS+UDS and UDS are able to outperform No Sharing and Reward Predictor while attaining competitive results compared to CDS and Sharing All with oracle rewards.

To better understand the performance of UDS under different relabeled data quality, we evaluate the UDS under different success rates of the data relabeled from other tasks in the multi-task Meta-World domain. Specifically, we filter out data shared from other tasks to ensure that the success rates of the relabeled data are 5%, 50% and 90% respectively. We compare the results of UDS on such data compositions to the performance of UDS in Table 3 where the success rate of relabeled data is 59.6% as shown in Table 9. The full results are in Table 10. UDS on relabeled data with 50% and 90% success rates achieves similar results compared to original UDS whereas UDS on relabel data with 5% success rate is significantly worse. Hence, UDS can obtain good results in settings where the relabeled data is of high quality despite incurring high reward bias, but is not helpful in settings where the shared data is of low quality and does not offer much information about solving the target task.

D Empirical Results of UDS and CDS+UDS in multi-task locomotion domain with dense rewards

In this section, we evaluate UDS and CDS+UDS in the multi-task locomotion setting with dense rewards. We pick the multi-task walker environment as used in prior work (Yu et al., 2021a), which consists of three tasks, `run forward`, `run backward` and `jump`. The reward functions of the three tasks are $r(s,a) = v_x - 0.001 * \|a\|_2^2$, $r(s,a) = -v_x - 0.001 * \|a\|_2^2$ and $r(s,a) = -\|v_x\| - 0.001 * \|a\|_2^2 + 10 * (z - \text{init z})$ respectively where v_x denotes the velocity along the x-axis and z denotes the z-position of the half-cheetah and init z denotes the initial z-position. In UDS and CDS+UDS, we relabel the rewards routed from other tasks with the minimum reward value in the offline dataset of the target task. As shown in Table 11, CDS+UDS and UDS outperform No Sharing and Reward Predictor by a large margin while also performing comparably to CDS and Sharing All. Therefore, CDS+UDS and UDS are able to solve multi-task locomotion tasks with dense rewards.

E Comparisons of CDS+UDS and UDS to Multi-Task Model-Based Offline RL Approaches

In this section, we compare CDS+UDS and UDS to a recent, state-of-the-art model-based offline RL method COMBO (Yu et al., 2021b) in the Meta-World domain. We adapt COMBO to the multi-task offline setting by learning the dynamics model on data of all tasks combined and and performing vanilla multi-task offline training without data sharing using the model learned with all of the data. As shown in Table 12, CDS+UDS and UDS indeed outperform COMBO in the average task success rate. The intuition behind this is that COMBO is unable to learn an accurate dynamics model for tasks with limited data as in our Meta-World setting.

F Comparisons to Pre-Trained Representation Learning on Unlabeled Data

A popular strategy to utilize large amounts of unlabeled data along with some limited amount of labeled data in supervised learning is to utilize the unlabeled data for learning representations, and then running supervised training on the labeled data. A similar strategy has been utilized for RL and imitation learning in some prior work (Yang & Nachum, 2021; Yang et al., 2021). On the other hand, the UDS and CDS+UDS strategies take a different route of handling unlabeled data, and it is

Environment	Tasks	CDS+UDS	UDS	COMBO (Yu et al., 2021b)
Meta-World	door open	**61.3%**±7.9%	51.9%±25.3%	0.0%±0.0%
	door close	**54.0%** ±42.5%	12.3%±27.6%	1.1%±1.6%
	drawer open	**73.5%**±9.6%	61.8%±16.3%	15.7%±15.2%
	drawer close	99.3%±0.7%	**99.6%**±0.7%	85.7%±13.3%
	average	**71.2%** ± 11.3%	56.4%±12.8%	25.6%±6.2%

Table 12. On the multi-task Meta-World domain, we compare CDS+UDS and UDS to the model-based offline RL method COMBO (Yu et al., 2021b) that trains a dynamics model on all of the data and performs model-based offline training using the learned model. CDS+UDS and UDS are able to outperform COMBO by a large margin.

Environment	Tasks	CDS + UDS (ours)	UDS (ours)	ACL
Meta-World	door open	**61.3%**±7.9%	51.9%±25.3%	2.8%±2.0%
	door close	54.0% ±42.5%	12.3%±27.6%	0.0%±0.0%
	drawer open	73.5%±9.6%	61.8%±16.3%	**83.2%**±14.2%
	drawer close	99.3%±0.7%	99.6%±0.7%	**100.0%**±0.0%
	average	**71.2%** ± 11.3%	56.4%±12.8%	46.4%±3.5%

Table 13. Comparison between **UDS / CDS+UDS** and the **ACL** (Yang & Nachum, 2021) that performs representation learning on the unlabeled data instead of data sharing. Both **UDS** and **CDS+UDS** outperform **ACL** by a significant margin in the average task result, suggesting that sharing the unlabeled data is crucial in improving the performance of multi-task offline RL with unlabeled data compared to only using the data for learning the representation.

natural to wonder how these representation learning approaches compare to our approach, UDS, that runs offline RL with the lowest possible reward, with an optional reweighting scheme.

To investigate this, we run experiments comparing our UDS and CDS+UDS approaches to state-of-the-art representation learning approaches for utilizing unlabeled data. First, we compare UDS and CDS+UDS to the ACL (Yang & Nachum, 2021) approach on the multi-task Meta-World domain. ACL pretrains a representation using a contrastive loss on both the labeled and unlabeled offline datasets and then runs standard multi-task offline RL using the pretrained representation with **No Sharing**. To handle the multi-task Meta-World domain, we use the version of ACL without inputting reward labels. ACL can be viewed as an alternative to our unlabeled sharing data scheme, which leverages unlabeled data for representation learning rather than sharing it directly. We show the comparison to ACL in Table 13. UDS and CDS+UDS outperform ACL in the average task performance while ACL is only proficient on drawer-open and drawer-close, and it cannot solve door-open or door-close. This indicates that sharing the unlabeled data, even when labeled with the minimum possible reward is important for offline RL performance, while pretraining representations on the whole multi-task offline dataset might have limited benefit. Also note that, in principle, UDS / CDS+UDS are complementary to ACL and these approaches can be combined together to further improve performance, which we leave as future work.

We also compare the results of UDS and CDS + UDS in the single-task AntMaze medium-play and large-play domains (shown in Table 2 in the main paper) to imitation learning methods, TRAIL (Yang et al., 2021) and other baselines from Yang et al. (2021), that also utilize unlabeled data. These methods train a representation using the unlabeled dataset, and then run behavioral cloning to imitate the expert trajectories, which are limited in number. For example, the TRAIL method learns a state-conditioned action representation, by fitting an energy-based model to the transition dynamics of the unlabeled dataset, and then performs downstream imitation learning using the labeled expert dataset. In contrast to these prior approaches, UDS and CDS+UDS do not make any assumptions about how expert the labeled dataset is, and so they are not technically comparable to these imitation learning methods directly. Moreover, any specialized representation learning objective can also be combined with UDS and CDS+UDS to further improve performance. Nevertheless, we hope that a direct comparison to representation learning on unlabeled data combined with downstream imitation will provide informative insights about the potential of UDS and CDS+UDS to effectively leverage unlabeled data. Comparing Table 2 to Figure 3 in Yang et al. (2021), we find that CDS+UDS outperforms the best method from Yang et al. (2021), TRAIL, on the antmaze-medium-play task and performs a bit worse on the antmaze-large-play task. CDS+UDS also outperforms all the other methods in Figure 3 of (Yang et al., 2021). Overall, this implies that UDS and CDS+UDS methods can perform comparably to state-of-the-art representation learning approaches with downstream imitation, without needing any specialized representation learning.

G Details of UDS and CDS+UDS

In this section, we include the details of training UDS and CDS+UDS in Appendix G.1 as well as details on the environment and datasets used in our experiments in Appendix G.2. Finally, we discuss the compute information of UDS and CDS+UDS in Appendix G.3.

G.1 Details on the training procedure

We first present our practical implementation of UDS optimizes the following objectives for the Q-functions and the policy in the *single-task offline RL setting*:

$$\hat{Q}^{k+1} \leftarrow \arg\min_{\hat{Q}} \beta \left(\mathbb{E}_{\mathbf{s}\sim\mathcal{D}_{\mathrm{L}}\cup\mathcal{D}_{\mathrm{U}},\mathbf{a}\sim\mu(\cdot|\mathbf{s})} \left[\hat{Q}(\mathbf{s},\mathbf{a})\right] - \mathbb{E}_{\mathbf{s},\mathbf{a}\sim\mathcal{D}_{\mathrm{L}}\cup\mathcal{D}_{\mathrm{U}}} \left[\hat{Q}(\mathbf{s},\mathbf{a})\right] \right) + \frac{1}{2}\mathbb{E}_{(\mathbf{s},\mathbf{a},\mathbf{s}')\sim\mathcal{D}_{\mathrm{L}}\cup\mathcal{D}_{\mathrm{U}}} \left[\left(\hat{Q}(\mathbf{s},\mathbf{a}) - \left(r(\mathbf{s},\mathbf{a}) + \gamma Q(\mathbf{s}',\mathbf{a}')\right) \right)^2 \right],$$

and

$$\pi \leftarrow \arg\max_{\pi'} \mathbb{E}_{\mathbf{s}\sim\mathcal{D}_{\mathrm{L}}\cup\mathcal{D}_{\mathrm{U}},\mathbf{a}\sim\pi'(\cdot|\mathbf{s})} \left[\hat{Q}^{\pi}(\mathbf{s},\mathbf{a})\right],$$

Moreover, CDS+UDS optimizes the following objectives for training the critic and the policy with a soft weight:

$$\hat{Q}^{k+1} \leftarrow \arg\min_{\hat{Q}} \beta \left(\mathbb{E}_{\mathbf{s}\sim\mathcal{D}_{\mathrm{L}}\cup\mathcal{D}_{\mathrm{U}},\mathbf{a}\sim\mu(\cdot|\mathbf{s})} \left[w_{\mathrm{CDS}}(\mathbf{s},\mathbf{a};\mathrm{U}\to\mathrm{L})\hat{Q}(\mathbf{s},\mathbf{a})\right] - \mathbb{E}_{\mathbf{s},\mathbf{a}\sim\mathcal{D}_{\mathrm{L}}\cup\mathcal{D}_{\mathrm{U}}} \left[w_{\mathrm{CDS}}(\mathbf{s},\mathbf{a};\mathrm{U}\to\mathrm{L})\hat{Q}(\mathbf{s},\mathbf{a})\right] \right) + \frac{1}{2}\mathbb{E}_{(\mathbf{s},\mathbf{a},\mathbf{s}')\sim\mathcal{D}_{\mathrm{L}}\cup\mathcal{D}_{\mathrm{U}}} \left[w_{\mathrm{CDS}}(\mathbf{s},\mathbf{a};\mathrm{U}\to\mathrm{L}) \left(\hat{Q}(\mathbf{s},\mathbf{a}) - \left(r(\mathbf{s},\mathbf{a}) + \gamma Q(\mathbf{s}',\mathbf{a}')\right) \right)^2 \right],$$

and

$$\pi \leftarrow \arg\max_{\pi'} \mathbb{E}_{\mathbf{s}\sim\mathcal{D}_{\mathrm{L}}\cup\mathcal{D}_{\mathrm{U}},\mathbf{a}\sim\pi'(\cdot|\mathbf{s})} \left[w_{\mathrm{CDS}}(\mathbf{s},\mathbf{a};\mathrm{U}\to\mathrm{L})\hat{Q}^{\pi}(\mathbf{s},\mathbf{a})\right],$$

where β is the coefficient of the CQL penalty on distribution shift, μ is an action sampling distribution that covers the action bound as in CQL. On the hopper domain, when the unlabeled data is random, we use the version of CQL that does not maximize the term $\mathbb{E}_{\mathbf{s},\mathbf{a}\sim\mathcal{D}_{\mathrm{L}}\cup\mathcal{D}_{\mathrm{U}}} \left[\hat{Q}(\mathbf{s},\mathbf{a})\right]$ to prevent overestimating Q-values on low-quality random data and use $\beta = 1.0$. We use $\beta = 5.0$ in the other settings in the hopper domain. On the AntMaze domain, following prior works (Kumar et al., 2020; Yu et al., 2021a), we use the Lagrange version of CQL, where the coefficient β is automatically tuned against a pre-specified constraint value on the CQL loss equal to $\tau = 10.0$. We adopt other hyperparameters used in (Yu et al., 2021a). We adapt the CDS weight to the single-task setting as follows: $w_{\mathrm{CDS}}(\mathbf{s},\mathbf{a};\mathrm{U}\to\mathrm{L}) := \sigma\left(\frac{\Delta(\mathbf{s},\mathbf{a};\mathrm{U}\to\mathrm{L})}{\tau}\right)$ where $\Delta(\mathbf{s},\mathbf{a};\mathrm{U}\to\mathrm{L}) = \hat{Q}^{\pi}(\mathbf{s},\mathbf{a}) - P_{k\%}\left\{\hat{Q}^{\pi}(\mathbf{s}',\mathbf{a}')\text{: }\mathbf{s}',\mathbf{a}'\sim\mathcal{D}_{\mathrm{L}}\right\}$ for $(\mathbf{s},\mathbf{a})\sim\mathcal{D}_{\mathrm{U}}$. We use $k = 50$ in all single-task domains.

Similarly in the *multi-task offline RL* setting, our practical implementation of UDS optimizes the following objectives for the Q-functions and the policy:

$$\hat{Q}^{k+1} \leftarrow \arg\min_{\hat{Q}} \mathbb{E}_{i\sim[N]} \left[\beta \left(\mathbb{E}_{j\sim[N]} \left[\mathbb{E}_{\mathbf{s}\sim\mathcal{D}_j,\mathbf{a}\sim\mu(\cdot|\mathbf{s},i)} \left[\hat{Q}(\mathbf{s},\mathbf{a},i)\right] - \mathbb{E}_{\mathbf{s},\mathbf{a}\sim\mathcal{D}_j} \left[\hat{Q}(\mathbf{s},\mathbf{a},i)\right] \right] \right) + \frac{1}{2}\mathbb{E}_{j\sim[N],(\mathbf{s},\mathbf{a},\mathbf{s}')\sim\mathcal{D}_j} \left[\left(\hat{Q}(\mathbf{s},\mathbf{a},i) - \left(r(\mathbf{s},\mathbf{a},i)\mathbb{1}_{\{j=i\}} + \gamma Q(\mathbf{s}',\mathbf{a}')\right) \right)^2 \right] \right],$$

and

$$\pi \leftarrow \arg\max_{\pi'} \mathbb{E}_{i\sim[N]} \left[\mathbb{E}_{j\sim[N],\mathbf{s}\sim\mathcal{D}_j,\mathbf{a}\sim\pi'(\cdot|\mathbf{s},i)} \left[\hat{Q}^{\pi}(\mathbf{s},\mathbf{a},i)\right] \right],$$

We also present the objective of CDS+UDS in the multi-task task as follows:

$$\hat{Q}^{k+1} \leftarrow \arg\min_{\hat{Q}} \mathbb{E}_{i\sim[N]} \left[\beta \left(\mathbb{E}_{j\sim[N]} \left[\mathbb{E}_{\mathbf{s}\sim\mathcal{D}_j,\mathbf{a}\sim\mu(\cdot|\mathbf{s},i)} \left[w_{\mathrm{CDS}}(\mathbf{s},\mathbf{a};j\to i)\hat{Q}(\mathbf{s},\mathbf{a},i)\right] - \mathbb{E}_{\mathbf{s},\mathbf{a}\sim\mathcal{D}_j} \left[w_{\mathrm{CDS}}(\mathbf{s},\mathbf{a};j\to i)\hat{Q}(\mathbf{s},\mathbf{a},i)\right] \right] \right) + \frac{1}{2}\mathbb{E}_{j\sim[N],(\mathbf{s},\mathbf{a},\mathbf{s}')\sim\mathcal{D}_j} \left[w_{\mathrm{CDS}}(\mathbf{s},\mathbf{a};j\to i) \left(\hat{Q}(\mathbf{s},\mathbf{a},i) - \left(r(\mathbf{s},\mathbf{a},i)\mathbb{1}_{\{j=i\}} + \gamma Q(\mathbf{s}',\mathbf{a}')\right) \right)^2 \right] \right],$$

and

$$\pi \leftarrow \arg\max_{\pi'} \mathbb{E}_{i\sim[N]} \left[\mathbb{E}_{j\sim[N],\mathbf{s}\sim\mathcal{D}_j,\mathbf{a}\sim\pi'(\cdot|\mathbf{s},i)} \left[w_{\mathrm{CDS}}(\mathbf{s},\mathbf{a};j\to i)\hat{Q}^{\pi}(\mathbf{s},\mathbf{a},i)\right] \right],$$

where we use $k = 90$ in the multi-task Meta-World domain and $k = 50$ in the other multi-task domains.

To compute the weight w_{CDS}, we pick τ, i.e. the temperature term, using the exponential running average of $\Delta(\mathbf{s}, \mathbf{a}; \text{U} \to \text{L})$ in the single-task setting or $\Delta(\mathbf{s}, \mathbf{a}; j \to i)$ in the multi-task setting with decay 0.995 following (Yu et al., 2021a). Following (Yu et al., 2021a) again, we clip the automatically chosen τ with a minimum and maximum threshold, which we directly use the values from (Yu et al., 2021a). We use $[1, \infty]$ as the minimum and maximum threshold for all state-based single-task and multi-task domains whereas the vision-based robotic manipulation domain does not require such clipping.

In the single-task experiments, we use a total batch size of 256 and balance the number of transitions sampled from $\mathcal{D}_{\text{L}}$ and $\mathcal{D}_{\text{U}}$ in each batch. In the multi-task experiments, following the training protocol in (Yu et al., 2021a), for experiments with low-dimensional inputs, we use a stratified batch with 128 transitions for each task to train the Q-functions and the policy. We also balance the numbers of transitions sampled from the original task and the number of transitions drawn from other task data. Specifically, for each task i, we sample 64 transitions from $\mathcal{D}_i$ and the remaining 64 transitions from $\cup_{j \neq i} \mathcal{D}_{j \to i}$. In CDS+UDS, for each task $i \in [N]$, we only apply $w_{\text{CDS+UDS}}$ to data shared from other tasks on multi-task Meta-World environments and multi-task vision-based robotic manipulation tasks while we also apply the relabeling weight to transitions sampled from the original task dataset $\mathcal{D}_i$ with 50% probability in the multi-task AntMaze domain.

Regarding the choices of the architectures, for state-based domains, we use 3-layer feedforward neural networks with 256 hidden units for both the Q-networks and the policy. In the multi-task domains, we condition the policy and the Q-functions on a one-hot task ID, which is appended to the input state. In domains with high-dimensional image inputs, we adopt the multi-headed convolutional neural networks used in (Kalashnikov et al., 2021; Yu et al., 2021a). We use images with dimension $472 \times 472 \times 3$, extra state features $(g_{\text{robot\_status}}, g_{\text{height}})$ and the one-hot task vector as the observations similar (Kalashnikov et al., 2021; Yu et al., 2021a). Following the set-up in (Kalashnikov et al., 2018; 2021; Yu et al., 2021a), we use Cartesian space control of the end-effector of the robot in 4D space (3D position and azimuth angle) along with two binary actions to open/close the gripper and terminate the episode respectively to represent the actions. For more details, see (Kalashnikov et al., 2018; 2021).

G.2 Details on the environment and the datasets

In this subsection, we include the discussion of the details the environment and datasets used for evaluating UDS and CDS+UDS. Note that all of our single-task datasets and environments are from the standard benchmark D4RL (Fu et al., 2020) while all of our multi-task environment and offline datasets are from prior work (Yu et al., 2021a). We will nonetheless discuss the details to make our work self-contained. We acknowledge that all datasets with low-dimensional inputs are under the MIT License.

Single-task hopper domains. We use the `hopper` environment and datasets from D4RL (Fu et al., 2020). We consider the following three datasets, `hopper-random`, `hopper-medium` and `hopper-expert`. We construct the seven combinations of different data compositions using the three datasets as discussed in Table 5. To construct the combination, we take the first 10k transitions from labeled dataset and concatenate these labeled transitions with the entire unlabeled dataset with 1M transitions.

Single-task AntMaze domains. We use the `AntMaze` environment and datasets from D4RL (Fu et al., 2020) where we consider two datasets, `antmaze-medium-play` and `antmaze-large-play`. These two datasets only contain 1M sub-optimal transitions that navigates to random or fixed locations that are different from the target task during evaluation. We use these two datasets as unlabeled datasets. For the labeled dataset, we use 10 expert demonstrations of solving the target task used in prior work (Yang et al., 2021).

Multi-task Meta-World domains. We use the `door open`, `door close`, `drawer open` and `drawer close` environments introduced in (Yu et al., 2021a) from the public Meta-World (Yu et al., 2020b) repo[1]. In this multi-task Meta-World environment, a door and a drawer are put on the same scene, which ensures that all four tasks share the same state space. The environment uses binary rewards for each task, which are adapted from the success condition defined in the Meta-World public repo. In this case, the robot gets a reward of 1 if it solves the target task and 0 otherwise. We use a fixed 200 timesteps for each episode and do not terminate the episode when receiving a reward of 1 at an intermediate timestep. We use large datasets with wide coverage of the state space and 152K transitions for the `door open` and `drawer close` tasks and datasets with limited (2K transitions), but optimal demonstrations for the `door close` and `drawer open` tasks.

[1]The Meta-World environment can be found at the open-sourced repo `https://github.com/rlworkgroup/metaworld`

We direct use the offline datasets constructed in (Yu et al., 2021a), which are generated by training online SAC policies for each task with the dense reward defined in the Meta-World repo for 500 epochs. The medium-replay datasets use the whole replay buffer of the online SAC agent until 150 epochs while the expert datasets are collected by the final online SAC policy.

Multi-task AntMaze domains. Following (Yu et al., 2021a), we use the `antmaze-medium-play` and `antmaze-large-play` datasets from D4RL (Fu et al., 2020) and partitioning the datasets into multi-task datasets in an undirected way defined in (Yu et al., 2021a). Specifically, the dataset is randomly splitted into chunks with equal size, and then each chunk is assigned to a randomly chosen task. Therefore, under such a task construction scheme, the task data for each task is of low success rate for the particular task it is assigned to and it is imperative for the multi-task offline RL algorithm to leverage effective data sharing strategy to achieve good performance. In AntMaze, we also use a binary reward, which provides the agent a reward of +1 when the ant reaches a position within a 0.5 radius of the task goal, which is also the reward used default by Fu et al. (2020). The terminal of an episode is set to be true when a reward of +1 is observed. We terminate the episode upon seeing a reward of 1 with the maximum possible 1000 transitions per episode. Following (Yu et al., 2021a), we modify the datasets introduced by Fu et al. (2020) by equally dividing the large dataset into different parts for different tasks, where each task corresponds to a different goal position.

Multi-task walker domains. We have presented the details of the multi-task walker environment in Appendix D.

Multi-task image-based robotic picking and placing domains. Following (Kalashnikov et al., 2021; Yu et al., 2021a), we use sparse rewards for each task. That is, reward 1 is assigned to episodes that meet the success conditions and 0 otherwise. The success conditions are defined in (Kalashnikov et al., 2021). We directly use the dataset used in (Yu et al., 2021a). Such a dataset is collected by first training a policy for each individual task using QT-Opt (Kalashnikov et al., 2018) until the success rate reaches 40% and 80% for picking tasks and placing tasks respectively and then combine the replay buffers of all tasks as the multi-task offline dataset. Note that the success rate of placing is higher because the robot is already holding the object at the start of the placing tasks, making the placing easier to solve. The dataset consists of a total number of 100K episodes with 25 transitions for each episode.

G.3 Computation Complexity

We train UDS and CDS+UDS on a single NVIDIA GeForce RTX 2080 Ti for one day on the state-based domains. For the vision-based robotic picking and placing experiments, it takes 3 days to train it on 16 TPUs.