

A Q-Values Learned via UDS

Proposition A.1. For a policy π , let $\hat{Q}_{UDS}^\pi(\cdot, \cdot, i)$ denote the fixed point of Eqn. 6. Then, $\hat{Q}_{UDS}^\pi(\mathbf{s}, \mathbf{a})$ lower-bounds the Q-function $\hat{Q}_{Sharing\ All}^\pi$ that would be obtained had we used the true rewards.

Proof. Note that $\hat{Q}_{UDS}^\pi(\mathbf{s}, \mathbf{a})$ at iteration $k+1$ is defined in Eq. 6, which we restate it for convenience:

$$\hat{Q}_{UDS}^{k+1}(\mathbf{s}, \mathbf{a}, i) \leftarrow \hat{r}(\mathbf{s}, \mathbf{a}, i) + \gamma \mathbb{E}_{\mathbf{s}' \sim \hat{P}(\mathbf{s}'|\mathbf{s}, \mathbf{a}), \pi(\mathbf{a}'|\mathbf{s}', i)} \left[\hat{Q}_{UDS}^k(\mathbf{s}', \mathbf{a}', i) \right] - \alpha \left(\frac{\pi(\mathbf{a}|\mathbf{s}, i)}{\hat{\pi}_\beta(\mathbf{a}|\mathbf{s}, i)} - 1 \right), \quad (8)$$

whereas $\hat{Q}_{Sharing\ All}^\pi(\mathbf{s}, \mathbf{a})$ at iteration $k+1$ is defined as

$$\hat{Q}_{Sharing\ All}^{k+1}(\mathbf{s}, \mathbf{a}, i) \leftarrow r(\mathbf{s}, \mathbf{a}, i) + \gamma \mathbb{E}_{\mathbf{s}' \sim \hat{P}(\mathbf{s}'|\mathbf{s}, \mathbf{a}), \pi(\mathbf{a}'|\mathbf{s}', i)} \left[\hat{Q}_{Sharing\ All}^k(\mathbf{s}', \mathbf{a}', i) \right] \quad (9)$$

$$- \alpha \left(\frac{\pi(\mathbf{a}|\mathbf{s}, i)}{\hat{\pi}_\beta(\mathbf{a}|\mathbf{s}, i)} - 1 \right). \quad (10)$$

Assume $\hat{Q}_{Sharing\ All}^0 = \hat{Q}_{UDS}^0$, i.e. same Q-value initialization and $\hat{Q}_{Sharing\ All}^k \geq \hat{Q}_{UDS}^k$. Using such induction hypothesis and the fact that $\hat{r}(\mathbf{s}, \mathbf{a}, i) \leq r(\mathbf{s}, \mathbf{a}, i)$ for all \mathbf{s}, \mathbf{a} , we can conclude that $\hat{Q}_{UDS}^\pi(\mathbf{s}, \mathbf{a}) \leq \hat{Q}_{Sharing\ All}^\pi(\mathbf{s}, \mathbf{a})$. Therefore, UDS learns the Q-value that lower-bounds the Q-values learned data sharing all tasks with the ground-truth rewards. \square

B Details of UDS and CUDS

In this section, we include the details of training UDS and CUDS in Appendix B.1 as well as details on the environment and datasets used in our experiments in Appendix B.2. Finally, we discuss the compute information of UDS and CUDS in Appendix B.3. For additional details, please see our anonymous website: <https://sites.google.com/view/uds-cuds/>.

B.1 Details on the training procedure

Our practical implementation of UDS optimizes the following objectives for the Q-functions and the policy:

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

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

Similarly, CUDS optimizes the following objectives for training the critic and the policy with a soft weight:

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

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

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. We follow all the CQL hyperparameters used in [80].

To compute the weight $w_{CUDS}(\mathbf{s}, \mathbf{a}; j \rightarrow i) := \sigma \left(\frac{\Delta(\mathbf{s}, \mathbf{a}; j \rightarrow i)}{\tau} \right)$, we pick τ , i.e. the temperature term, using the exponential running average of $\Delta(\mathbf{s}, \mathbf{a}; j \rightarrow i)$ with decay 0.995 for each task following

[80]. Following [80] again, we clip the automatically chosen τ with a minimum and maximum threshold, which we directly use the values from [80]. We use $[1, 50]$ and $[10, \infty]$ as the minimum and maximum threshold for the multi-task Meta-World and AntMaze domains respectively whereas the vision-based robotic manipulation domain does not require such clipping.

Following the training protocol in [80], 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 $\cup_{j \neq i} \mathcal{D}_{j \rightarrow i}$. In CUDS, for each task $i \in [N]$, we only apply w_{CUDS} 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. We condition the policy 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 [30, 80]. 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 [30, 80]. Following the set-up in [29, 30, 80], 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 [29, 30].

B.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 CUDS. Note that all of our environment and offline datasets are from prior work [80]. 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.

Multi-task Meta-World domains. We use the door open, door close, drawer open and drawer close environments introduced in [80] from the public Meta-World [78] repo¹. 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 direct use the offline datasets constructed in [80], 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 [80], we use the antmaze-medium-play and antmaze-large-play datasets from D4RL [20] and partitioning the datasets into multi-task datasets in an undirected way defined in [80]. 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. [20]. The terminal of an episode is set to be true when a reward of +1 is observed.

Multi-task image-based robotic picking and placing domains. Following [30, 80], 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 [30]. We directly use the dataset used in [80]. Such a dataset is collected by first training a policy for each individual task using QT-Opt [29] 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. The dataset consists of a total number of 100K episodes with 25 transitions for each episode.

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

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

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 6: Performance of UDS under different actual success rates of the relabeled data.

B.3 Computation Complexity

We train UDS and CUDS 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.

C Additional details on the quality of data shared from other tasks

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 5. 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 CUDS 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.

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 1 where the success rate of relabeled data is 59.6% as shown in Table 5. The full results are in Table 6. 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 CUDS in more general dense reward settings

In this section, we evaluate UDS and CUDS in the dense reward setting in order to test if UDS and CUDS work in more general reward settings and are not limited to binary rewards. We pick the multi-task walker environment as used in prior work [80], 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 $\text{init } z$ denotes the initial z-position. In UDS and CUDS, we relabel the rewards routed from other

Environment	Tasks / Dataset type	CUDS (ours)	UDS (ours)	No Sharing	CDS (oracle)	Sharing All (oracle)
walker2d	run forward / medium-replay	880.1 \pm 108.8	665.0 \pm 84.9	590.1 \pm 48.6	1057.9 \pm 121.6	701.4 \pm 47.0
	run backward / medium	717.8 \pm 78.3	689.3 \pm 16.3	614.7 \pm 87.3	564.8 \pm 47.7	756.7 \pm 76.7
	jump / expert	1487.7 \pm 177.6	1036.0 \pm 247.1	1575.2 \pm 70.9	1418.2 \pm 138.4	885.1 \pm 152.9
	average	1028.6 \pm 76.8	796.7 \pm 106.3	926.6 \pm 37.7	1013.6 \pm 71.5	781.0 \pm 100.8

Table 7: Results for multi-task walker experiment with dense rewards. CUDS and UDS are able to outperform No Sharing while attaining competitive results compared to CDS and Sharing All with oracle rewards. This suggests that CUDS and UDS are able to solve more general problems where rewards are not binary.

Environment	Tasks	CUDS (ours)	UDS (ours)	COMBO [81]
Meta-World	door open	61.3% \pm 7.9%	51.9% \pm 25.3%	0.0% \pm 0.0%
	door close	54.0% \pm 42.5%	12.3% \pm 27.6%	1.1% \pm 1.6%
	drawer open	73.5% \pm 9.6%	61.8% \pm 16.3%	15.7% \pm 15.2%
	drawer close	99.3% \pm 0.7%	99.6% \pm 0.7%	85.7% \pm 13.3%
	average	71.2% \pm 11.3%	56.4% \pm 12.8%	25.6% \pm 6.2%

Table 8: On the multi-task Meta-World domain, we compare CUDS and UDS to the model-based offline RL method COMBO [81] that trains a dynamics model on all of the data and performs model-based offline training using the learned model. CUDS and UDS are able to outperform COMBO by a large margin.

tasks with the minimum reward value in the offline dataset of the target task. As shown in Table 7, CUDS and UDS outperform No Sharing by a large margin while also performing comparably to CDS and Sharing All. Therefore, CUDS and UDS are not limited to settings with binary rewards but are able to be applied to more general cases, in particular, environments with dense rewards.

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

In this section, we compare CUDS and UDS to a recent, state-of-the-art model-based offline RL method COMBO [81] 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 performing vanilla multi-task offline training without data sharing using the model learned with all of the data. As shown in Table 8, CUDS 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 Theoretical Analysis of UDS and CUDS

In this section, we will theoretically analyze UDS and CUDS to 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.

F.1 Notation and Assumptions

Let $\pi_\beta(\mathbf{a}|\mathbf{s})$ denote the behavior policy for task i (note that index i was dropped from $\pi_\beta(\mathbf{a}|\mathbf{s}; i)$ for brevity). The dataset, \mathcal{D}_i 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 $d_{\mathcal{D}}^\pi$ as the state marginal distribution introduced by the dataset \mathcal{D} under π . For our analysis, we will abstract offline RL algorithms into a generic constrained policy optimization problem [35]:

$$\pi^*(\mathbf{a}|\mathbf{s}) := \arg \max_{\pi} J_{\mathcal{D}}(\pi) - \frac{\alpha}{1-\gamma} D(\pi, \pi_\beta). \quad (11)$$

$J_{\mathcal{D}}(\pi)$ 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 [27, 70], MMD distance [34] or D_{CQL} [35]) between the learned policy π and the behavior policy π_β . 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 [35, 80] have shown that the optimal policy π_i^* that optimizes Equation 11 attains a high probability safe-policy improvement guarantee, i.e., $J(\pi_i^*) \geq J(\pi_\beta) - \zeta_i$, where ζ_i is:

$$\zeta_i = \mathcal{O} \left(\frac{1}{(1-\gamma)^2} \right) \mathbb{E}_{\mathbf{s} \sim d_{\mathcal{D}_i}^{\pi_i^*}} \left[\sqrt{\frac{D_{\text{CQL}}(\pi_i^*, \pi_\beta)(\mathbf{s}) + 1}{|\mathcal{D}_i(\mathbf{s})|}} \right] - \frac{\alpha}{1-\gamma} D(\pi_i^*, \pi_\beta). \quad (12)$$

The first term in Equation 12 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 $\mathbf{s} \sim d_{\mathcal{D}_i}^{\pi_i^*}$. We will show that UDS and CUDS enjoy safe policy improvement. In our analysis, we assume $r(\mathbf{s}, \mathbf{a}) \in [0, 1]$. Finally, as discussed in Section 3, let $\mathcal{D}_i^{\text{eff}}$ 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 [35, 80] we assume that the empirical rewards and dynamics concentrate towards their mean.

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

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

Similar to prior work [35, 80], we also make a coverage assumption, i.e., we assume that each state-action pair is observed in the dataset \mathcal{D}_i , 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 [10]), 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 illustrative analysis in a tabular setting here.

F.2 Theoretical Results

We first provide a performance guarantee for UDS which is then used to show that under certain conditions on the sizes of the labeled \mathcal{D}_i and the effective dataset, $\mathcal{D}_i^{\text{eff}}$, UDS attains a better policy improvement guarantee than naïve no sharing. We first briefly discuss a novel component of our proof technique, then present the theoretical results, and then interpret it.

F.2.1 Our Proof Technique

While there are several techniques to provide guarantees for offline RL algorithms, we will build on the line of safe-policy improvement bounds, previously used in Kumar et al. [35], Yu et al. [80]. However, naïvely applying these guarantees to our UDS setting will give rise to very weak bounds, since a number of these guarantees utilize a bound on the value difference of the policy in the empirical MDP and the actual MDP (term (i)) as shown below:

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

Typically, term (i) depends on the sampling error on states that are visited by the learned policy π , and decays to 0 with infinite samples, but UDS can learn quite pessimistic Q-values due to the reward labeling procedure. However, this may not affect the policy performance since the relative ordering of actions might still be the same. This is not accounted for in any prior analysis we are aware of.

Therefore, we introduce a novel analysis tool that, rather than decomposing $J(\pi) - J(\pi_\beta)$ naïvely using the return in the empirical MDP, decomposes it using the return of the policy π in the best empirical MDP that still produces the policy π as its optimal policy. One simple way to obtain this best empirical MDP is via affine transformations on the reward function that preserve optimality. So, for strengthening our bound, we shall compute the bound similar to the above equation for different affine transformations of the reward and pick the one that gives the tightest bound.

Formally, let $g(\cdot)$ be an affine function: $g(x) = u \cdot x + v$ for some $u > 0, u \in \mathbb{R}$ and $v \in \mathbb{R}$. Then our decomposition looks like:

$$J(\pi) - J(\pi_\beta) := \underbrace{J(\pi) - g(\hat{J}(\pi))}_{(i)} + \underbrace{g(\hat{J}(\pi)) - g(\hat{J}(\pi_\beta))}_{(ii)} + \underbrace{g(\hat{J}(\pi_\beta)) - J(\pi_\beta)}_{(iii)},$$

Then, to obtain a strong lower bound on $J(\pi) - J(\pi_\beta)$, we can first bound each of the terms for a given choice of $g = (u, v)$, and then take the supremum over u and v . This is reflected in the performance guarantee we present next.

F.2.2 Performance Guarantee for UDS

Proposition F.1 (Policy improvement guarantee for UDS). *Let π_{UDS}^* denote the policy learned by UDS for a given task i , and let $\pi_\beta^{\text{eff}}(\mathbf{a}|\mathbf{s}, i)$ denote the behavior policy for the combined dataset for task i , $\mathcal{D}_i^{\text{eff}}$. Then with high probability $\geq 1 - \delta$, π_{UDS}^* is a ζ -safe policy improvement over π_β^{eff} , i.e., $J(\pi_{UDS}^*) \geq J(\pi_\beta^{\text{eff}}) - \zeta$, where ζ is:*

$$\begin{aligned} \zeta = \min_{u>0, v} \quad & \zeta_{u,v} \\ \zeta_{u,v} = & \underbrace{\frac{1}{1-\gamma} \left| \mathbb{E}_{\mathbf{s}, \mathbf{a} \sim d_{\mathcal{D}_i^{\text{eff}}}^{\pi_\beta}} [1 - f(\mathbf{s}, \mathbf{a})] - v \right|}_{(a): \text{reward bias, but modified for the best } u} - \underbrace{\frac{\alpha u}{1-\gamma} D(\pi_{UDS}^*, \pi_\beta^{\text{eff}})}_{(b): \text{policy improvement}} \\ & + \underbrace{\frac{2C_{P,\delta}\gamma}{(1-\gamma)^2} \left[\sqrt{\frac{D_{CQL}(\pi_{UDS}^*, \pi_\beta^{\text{eff}})(\mathbf{s}) + 1}{|\mathcal{D}_i^{\text{eff}}(\mathbf{s})|}} \right]}_{(c): \text{dynamics sampling error}} + \underbrace{\frac{2uC_{r,\delta}}{(1-\gamma)} \mathbb{E}_{\mathbf{s}, \mathbf{a} \sim d_{\mathcal{D}_i}^\pi} \left[\frac{f(\mathbf{s}, \mathbf{a})}{\sqrt{|\mathcal{D}_i(\mathbf{s}, \mathbf{a})|}} \right]}_{(d): \text{reward sampling error, but scaled down}}, \end{aligned}$$

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

A proof of Proposition F.1 is provided in Appendix F.3. To intuitively interpret the various terms that appear, we note that term (b) corresponds to the standard policy improvement that arises as a result of using an offline RL algorithm, term (c) corresponds to sampling error that arises as a result of performing offline RL on the dynamics induced by a finite dataset, but note that this term depends on the size of the effective dataset, $\mathcal{D}_i^{\text{eff}}$ and not only the labeled dataset \mathcal{D}_i for the task. Term (a) corresponds to the bias incurred as a result of labeling various transitions with a 0 reward in the data, and term (d) corresponds to the sampling error in the reward function, under the assumption of a stochastic reward function.

F.2.3 How does UDS compare to No Sharing?

In the setting when no data is shared across tasks, we attain the guarantee shown in Equation 12. Comparing Proposition F.1 to this guarantee, we note that under some scenarios, UDS yields a tighter bound compared to No Sharing. Two such scenarios are given by:

1. **Long-horizon tasks:** Consider a scenario where tasks have a long horizon $H = \frac{1}{1-\gamma}$ and $|\mathcal{D}_i^{\text{eff}}(\mathbf{s})| = H^2|\mathcal{D}_i(\mathbf{s})|$. In this case, dynamics sampling error term (term (c)) consists of one less factor of H when UDS is utilized, compared to when it is not. Since the dynamics sampling error grows quadratically in the horizon, whereas other terms grow linearly, a reduction in this term by increasing sample size (i.e., denominator) can lead to a stronger guarantee for UDS than No Sharing. This reasoning does not even consider term (d), which can be trivially upper-bounded by the corresponding term for No Sharing, even though UDS reduces this term as well.
2. **The fraction $f(\mathbf{s}, \mathbf{a})$ is identical for all state-action pairs in the labeled \mathcal{D}_i , i.e., the unlabeled dataset consists of equal proportions state-action pairs as the labeled dataset.** Consider an extreme case when the unlabeled dataset consists of the trajectories in the labeled dataset such that $f(\mathbf{s}, \mathbf{a}) = c_0$ for all state-action tuples, just not annotated with rewards. In this case, reward bias takes on a constant value across all the transitions in the dataset, and by virtue of utilizing u and v in our bound in Proposition F.1, we note that the overall effect of this reward bias disappears, since v can compensate for this bias.

F.2.4 Extension to CUDS

Finally, we extend Proposition F.1 to a performance guarantee for CUDS by integrating the technique above with the analysis from Yu et al. [80]. To analyze CUDS, we consider the abstract model of the conservative data sharing scheme developed by Yu et al. [80]. This model suggests that CUDS approximates the following optimization in the empirical MDP generated by the relabeled dataset:

$$(\pi^*(\mathbf{a}|\mathbf{s}, i), \pi_\beta^*(\mathbf{a}|\mathbf{s}, i)) := \arg \max_{\pi, \pi_\beta \in \Pi_{\text{relabel}}} \hat{J}_{\mathcal{D}^{\text{eff}}}(\pi) - \frac{\alpha}{1-\gamma} D(\pi, \pi_\beta). \quad (13)$$

Now, utilizing Proposition F.1 and Proposition 5.1 from Yu et al. [80], we obtain the following guarantee for CUDS:

Corollary F.1. *Let $\pi_{\text{CUDS}}^*(\mathbf{a}|\mathbf{s}, i)$ be the optimal policy found by CUDS (Equation 13) and let $\pi_\beta^*(\mathbf{a}|\mathbf{s}, i)$ denote the behavior policy that optimizes Equation 13 for task $i \in [N]$. Then, with high probability $\geq 1 - \delta$, π_{CUDS}^* is a ζ -safe policy improvement over π_β^* , i.e., $J(\pi_{\text{CUDS}}^*) \geq J(\pi_\beta^*) - \zeta_{\text{CUDS}}$, where ζ_{CUDS} is given by:*

$$\begin{aligned} \zeta_{\text{CUDS}} = \min_{u>0, v} \zeta_{u,v} \\ \zeta_{u,v} = \underbrace{\frac{1}{1-\gamma} \left| \mathbb{E}_{\mathbf{s}, \mathbf{a} \sim d_{\mathcal{D}_i^{\text{eff}}}^{\pi_\beta}} [1 - f(\mathbf{s}, \mathbf{a})] - v \right|}_{(a): \text{reward bias}} - \underbrace{\frac{\alpha u}{1-\gamma} D(\pi_{\text{CUDS}}^*, \pi_\beta^*)}_{(b): \text{policy improvement}} \\ + \underbrace{\frac{2C_{P,\delta}\gamma}{(1-\gamma)^2} \left[\sqrt{\frac{D_{\text{CQL}}(\pi_{\text{CUDS}}^*, \pi_\beta^*)(\mathbf{s}) + 1}{|\mathcal{D}_i^{\text{eff}}(\mathbf{s})|}} \right]}_{(c): \text{dynamics sampling error}} + \underbrace{\frac{2uC_{r,\delta}}{(1-\gamma)} \mathbb{E}_{\mathbf{s}, \mathbf{a} \sim d_{\mathcal{D}_i}^{\pi_\beta}} \left[\frac{f(\mathbf{s}, \mathbf{a})}{\sqrt{|\mathcal{D}_i(\mathbf{s}, \mathbf{a})|}} \right]}_{(d): \text{reward sampling error, but scaled down}}. \end{aligned}$$

Proof. The proof of this proposition follows directly from the proof of Proposition F.1 with the exception that this argument must be applied against the optimized behavior policy π_β^* . \square

Comparing the bounds for CUDS and UDS. We now interpret the bound in Corollary F.1 comparatively against the bound in Proposition F.1. First note that since the abstract model of CUDS (Equation 13) optimizes the behavior policy π_β^* , we first note from Equation 14 of Yu et al. [80] that for any other behavior policy π' ,

$$D(\pi_{\text{CUDS}}^*, \pi_\beta^*) \leq D(\pi_{\text{CUDS}}^*, \pi'). \quad (14)$$

This means that the numerator of the sampling error term (term (c)) is smaller when CUDS is utilized as compared to when UDS is utilized. In addition, since CUDS relabels unlabeled data from Equation 13, this scheme also increases the dataset size, increasing the denominator of term (c). On the other hand, note that while UDS increases the denominator $|\mathcal{D}_i^{\text{eff}}|$, it may also increase the distributional shift $D(\pi^*, \pi_\beta^{\text{eff}})$ appearing in the numerator of the sampling error term. Our practical version of CUDS (Equation 2), which approximates Equation 13 by relabeling only the top k percentile of the unlabeled data based on the objective in Equation 2, gives us a control over the effective dataset size after relabeling $|\mathcal{D}_i^{\text{eff}}|$, while still ensuring a reduced value of $D(\pi^*, \pi_\beta^*)$, and is thus expected to reduce ζ compared to UDS.

Intuitively, note that the bounds in Proposition F.1 and Corollary F.1, guarantee safe-policy improvement over different base policies π_β^{eff} vs π_β^* . Intuitively, we would expect that $J(\pi_\beta^*) \geq J(\pi_\beta^{\text{eff}})$ in practice, especially for a large α , since CUDS optimizes the behavior policy towards high return, compared to simply relabeling all unlabeled transitions. Therefore, CUDS not only reduces ζ compared to UDS, but also, in practice, is expected to improve over π_β^* , which performs better than π_β^{eff} . Thus, we would expect CUDS to be better in practice compared to UDS.

F.3 Proof of Proposition F.1

As mentioned in the beginning of Section F.2.1, to strengthen the conventional safe policy improvement bound, we instead utilize a different form of loss decomposition of the improvement of the learned policy relative to the behavior policy with the affine transformation g :

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

Now we will discuss how to bound each of the terms: terms (i) and (ii) correspond to the divergence between a transformed 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 $(\mathbf{s}, \mathbf{a}, r, \mathbf{s}') \in \mathcal{D}_i^{\text{eff}}$, as \hat{r}_i^{eff} , which considers contributions from both the reward $\hat{r}(\mathbf{s}, \mathbf{a})$ observed in dataset \mathcal{D}_i , and the contribution of 0 reward from the relabeled dataset:

$$\hat{r}_i^{\text{eff}}(\mathbf{s}, \mathbf{a}) = \frac{|\mathcal{D}_i(\mathbf{s}, \mathbf{a})| \cdot \hat{r}(\mathbf{s}, \mathbf{a}) + |\mathcal{D}_i^{\text{eff}}(\mathbf{s}, \mathbf{a}) \setminus \mathcal{D}_i(\mathbf{s}, \mathbf{a})| \cdot 0}{|\mathcal{D}_i^{\text{eff}}(\mathbf{s}, \mathbf{a})|} \quad (15)$$

Define $f(\mathbf{s}, \mathbf{a}) := \frac{|\mathcal{D}_i(\mathbf{s}, \mathbf{a})|}{|\mathcal{D}_i^{\text{eff}}(\mathbf{s}, \mathbf{a})|}$ for notation compactness. Equation 15 and the form of the reward transformation $g(x) = u \cdot x + v$ can then be used to derive the following difference against the true rewards:

$$u\hat{r}_i^{\text{eff}}(\mathbf{s}, \mathbf{a}) + v - r(\mathbf{s}, \mathbf{a}) = uf(\mathbf{s}, \mathbf{a})(\hat{r}(\mathbf{s}, \mathbf{a}) - r(\mathbf{s}, \mathbf{a})) + (1 - uf(\mathbf{s}, \mathbf{a})) \cdot (0 - r(\mathbf{s}, \mathbf{a})) + v \quad (16)$$

$$\begin{aligned} &\leq uf(\mathbf{s}, \mathbf{a}) \cdot \frac{C_{r,\delta}}{\sqrt{|\mathcal{D}_i(\mathbf{s}, \mathbf{a})|}} - (1 - f(\mathbf{s}, \mathbf{a})u) \cdot r(\mathbf{s}, \mathbf{a}) + v \\ &\leq uf(\mathbf{s}, \mathbf{a}) \cdot \frac{C_{r,\delta}}{\sqrt{|\mathcal{D}_i(\mathbf{s}, \mathbf{a})|}}, \end{aligned} \quad (17)$$

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

$$\begin{aligned} u\hat{r}_i^{\text{eff}}(\mathbf{s}, \mathbf{a}) + v - r(\mathbf{s}, \mathbf{a}) &= uf(\mathbf{s}, \mathbf{a}) \cdot (\hat{r}(\mathbf{s}, \mathbf{a}) - r(\mathbf{s}, \mathbf{a})) + (1 - f(\mathbf{s}, \mathbf{a})u) \cdot (-r(\mathbf{s}, \mathbf{a})) + v \\ &\geq -uf(\mathbf{s}, \mathbf{a}) \cdot \frac{C_{r,\delta}}{\sqrt{|\mathcal{D}_i(\mathbf{s}, \mathbf{a})|}} - (1 - f(\mathbf{s}, \mathbf{a})u) + v, \end{aligned} \quad (18)$$

where the last step follows from the fact that $r(\mathbf{s}, \mathbf{a}) \leq 1$. To highlight the significance of this reward transformation, note that in the last step, if $\forall \mathbf{s}, \mathbf{a}$, $f(\mathbf{s}, \mathbf{a}) = c_0$, then the best reward transformation would choose $v = 1 - c_0$, and that completely eliminates the excess bias induced in the bound.

Upper bounding $g(\hat{J}_i(\pi)) - J_i(\pi)$. 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

$$\begin{aligned} g(\hat{J}_i(\pi)) - J_i(\pi) &= \frac{1}{1-\gamma} \sum_{\mathbf{s}, \mathbf{a}} \left(\hat{d}_{\mathcal{D}_i^{\text{eff}}}^\pi(\mathbf{s}) \pi(\mathbf{a}|\mathbf{s}) g(\hat{r}_i^{\text{eff}}(\mathbf{s}, \mathbf{a})) - d_i^\pi(\mathbf{s}) \pi(\mathbf{a}|\mathbf{s}) r(\mathbf{s}, \mathbf{a}) \right) \\ &\leq \underbrace{\frac{1}{1-\gamma} \sum_{\mathbf{s}, \mathbf{a}} \hat{d}_{\mathcal{D}_i^{\text{eff}}}^\pi(\mathbf{s}) \pi(\mathbf{a}|\mathbf{s}) (g(\hat{r}_i^{\text{eff}}(\mathbf{s}, \mathbf{a})) - r(\mathbf{s}, \mathbf{a}))}_{:=\Delta_1} + \underbrace{\frac{1}{1-\gamma} \sum_{\mathbf{s}, \mathbf{a}} (\hat{d}_{\mathcal{D}_i^{\text{eff}}}^\pi(\mathbf{s}) - d_i^\pi(\mathbf{s})) \pi(\mathbf{a}|\mathbf{s}) r(\mathbf{s}, \mathbf{a})}_{:=\Delta_2} \end{aligned} \quad (19)$$

Following Kumar et al. [35] (Theorem 3.6), we can bound the second term Δ_2 using:

$$|\Delta_2| \leq \frac{\gamma C_{P,\delta}}{1-\gamma} \mathbb{E}_{\mathbf{s} \sim \hat{d}_{\mathcal{D}_i^{\text{eff}}}^\pi(\mathbf{s})} \left[\frac{\sqrt{|\mathcal{A}|}}{\sqrt{|\mathcal{D}_i^{\text{eff}}(\mathbf{s})|}} \sqrt{D(\pi, \hat{\pi}_\beta^{\text{eff}})(\mathbf{s}) + 1} \right]. \quad (20)$$

To upper bound Δ_1 , we utilize the reward upper bound from Equation 16:

$$\Delta_1 = \sum_{\mathbf{s}} \hat{d}_{\mathcal{D}_i^{\text{eff}}}^\pi(\mathbf{s}) \left(\sum_{\mathbf{a}} \pi(\mathbf{a}|\mathbf{s}) (u \hat{r}_i^{\text{eff}}(\mathbf{s}, \mathbf{a}) + v - r(\mathbf{s}, \mathbf{a})) \right) \quad (21)$$

$$\leq \underbrace{\sum_{\mathbf{s}} \hat{d}_{\mathcal{D}_i^{\text{eff}}}^\pi(\mathbf{s}) \sum_{\mathbf{a}} u f(\mathbf{s}, \mathbf{a}) \frac{C_{r,\delta}}{\sqrt{|\mathcal{D}_i(\mathbf{s})|}} \frac{\pi(\mathbf{a}|\mathbf{s})}{\sqrt{\hat{\pi}_\beta(\mathbf{a}|\mathbf{s})}}}_{=\Delta'_1}. \quad (22)$$

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

$$J_i(\pi) \geq g(\hat{J}_i(\pi)) - \frac{|\Delta_2|}{1-\gamma} - \frac{|\Delta'_1|}{1-\gamma}. \quad (23)$$

Lower bounding $g(\hat{J}_i(\pi)) - J_i(\pi)$. To lower bound this quantity, we follow the step shown in Equation 19, and lower bound the term Δ_2 by using the negative of the RHS of Equation 20, and lower bound Δ_1 by upper bounding its absolute value as shown below:

$$\begin{aligned} |\Delta_1| &= \left| \sum_{\mathbf{s}} \hat{d}_{\mathcal{D}_i^{\text{eff}}}^\pi(\mathbf{s}) \left(\sum_{\mathbf{a}} \pi(\mathbf{a}|\mathbf{s}) (u \hat{r}_i^{\text{eff}}(\mathbf{s}, \mathbf{a}) + v - r(\mathbf{s}, \mathbf{a})) \right) \right| \\ &\leq \underbrace{\sum_{\mathbf{s}} \hat{d}_{\mathcal{D}_i^{\text{eff}}}^\pi(\mathbf{s}) \sum_{\mathbf{a}} u f(\mathbf{s}, \mathbf{a}) \frac{C_{r,\delta}}{\sqrt{|\mathcal{D}_i(\mathbf{s})|}} \frac{\pi(\mathbf{a}|\mathbf{s})}{\sqrt{\hat{\pi}_\beta(\mathbf{a}|\mathbf{s})}}}_{=\Delta'_1} + \left| \sum_{\mathbf{s}} \hat{d}_{\mathcal{D}_i^{\text{eff}}}^\pi(\mathbf{s}) \sum_{\mathbf{a}} \pi(\mathbf{a}|\mathbf{s}) \cdot (1 - f(\mathbf{s}, \mathbf{a})u) - v \right|. \end{aligned} \quad (24)$$

$$(25)$$

This gives rise to the complete lower bound:

$$g(\hat{J}_i(\pi)) \geq J_i(\pi) - \frac{|\Delta_2|}{1-\gamma} - \frac{1}{1-\gamma} \left| \sum_{\mathbf{s}, \mathbf{a}} \hat{d}_{\mathcal{D}_i^{\text{eff}}}^\pi(\mathbf{s}) \pi(\mathbf{a}|\mathbf{s}) (1 - f(\mathbf{s}, \mathbf{a})u) - v \right| - \frac{\Delta'_1}{1-\gamma}. \quad (26)$$

Policy improvement term (ii). Finally, the missing piece that needs to be bounded is the policy improvement term (ii) in the decomposition of $g(J(\pi)) - g(J(\pi_\beta))$. Utilizing the abstract form of offline RL (Equation 11, we note that term (ii) is lower bounded as:

$$\text{term (ii)} \geq \frac{\alpha u}{1-\gamma} D(\pi, \pi_\beta). \quad (27)$$

Putting it all together. To obtain the final expression of Proposition F.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_β^{eff} . Applying Equation 26 for term (i) on policy π , Equation 27 for term (ii), and Equation 23 for the behavior policy π_β^{eff} , we obtain the following:

$$\begin{aligned}
J(\pi) - J(\pi_\beta^{\text{eff}}) &= J(\pi) - g\left(\widehat{J}(\pi)\right) + g\left(\widehat{J}(\pi)\right) - g\left(\widehat{J}(\pi_\beta^{\text{eff}})\right) + g\left(\widehat{J}(\pi_\beta^{\text{eff}})\right) - J(\pi_\beta^{\text{eff}}) \\
&\geq -\frac{2\gamma C_{P,\delta}}{(1-\gamma)^2} \mathbb{E}_{\mathbf{s} \sim \widehat{d}_{\mathcal{D}_i^{\text{eff}}}^\pi(\mathbf{s})} \left[\frac{\sqrt{|\mathcal{A}|}}{\sqrt{|\mathcal{D}^{\text{eff}}(\mathbf{s})|}} \sqrt{D(\pi, \widehat{\pi}_\beta^{\text{eff}})(\mathbf{s}) + 1} \right] - \frac{2u C_{r,\delta}}{1-\gamma} \mathbb{E}_{\mathbf{s}, \mathbf{a} \sim \widehat{d}_{\mathcal{D}_i^{\text{eff}}}^\pi} \left[\frac{f(\mathbf{s}, \mathbf{a})}{\sqrt{|\mathcal{D}_i(\mathbf{s}, \mathbf{a})|}} \right] \\
&\quad - \frac{1}{1-\gamma} \underbrace{\left| \mathbb{E}_{\mathbf{s}, \mathbf{a} \sim d_{\mathcal{D}_i^{\text{eff}}}^{\pi_\beta}} [1 - f(\mathbf{s}, \mathbf{a})u] - v \right|}_{:= \Delta_3} + \frac{\alpha u}{1-\gamma} D(\pi, \pi_\beta^{\text{eff}}).
\end{aligned}$$

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 π_β 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.

Finally to finish the proof, we can take the supremum over the best choice of (u, v) . Thus, we obtain the desired bound in Proposition F.1.

G Empirical analysis of the reason that CUDS and UDS work

In this section, we perform an empirical study on the Meta-World domain to better understand the reason that UDS and CUDS 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.

G.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**).

We share the same data from the other three tasks, door close, drawer open and drawer close as in Table 1, which are . As shown in Table 9, both UDS and CUDS 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 CUDS significantly outperform No Sharing in the limited expert and medium data setting whereas in the medium-replay setting with broader coverage, CUDS outperforms No sharing but UDS fails to achieve non-zero success rate. Such results suggest that UDS and CUDS 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.

G.2 D4RL Hopper Data Quality + Coverage Diagnostic Study

To further understand the sensitivity of UDS to the data coverage and the data quality of both target task data (i.e. with reward labels) and relabeled data (i.e. without reward labels), we perform another empirical study using the hopper environment from the D4RL [20] benchmark. We consider the following 6 different combinations varying the quality and amount of the labeled and unlabeled datasets:

Environment	Dataset type / size	CUDS (ours)	UDS (ours)	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 9: 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. CUDS 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 CUDS 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.

1. 10k labeled data from hopper-expert + unlabeled 1M data hopper-random (i.e., **high-quality + narrow labeled data, low-quality + broad unlabeled data**)
2. 10k labeled data from hopper-expert + unlabeled 1M data from hopper-medium (i.e., **high-quality + narrow labeled data, medium-quality + narrow unlabeled data**)
3. 10k labeled data from hopper-medium + unlabeled 1M data from hopper-random (i.e., **medium-quality + narrow labeled data, low-quality + broad unlabeled data**)
4. 10k labeled data from hopper-medium + unlabeled 1M data from hopper-expert (i.e., **medium-quality + narrow labeled data, high-quality + narrow unlabeled data**)
5. 10k labeled data from hopper-random + unlabeled 1M data from hopper-medium (i.e., **low-quality + broad labeled data, medium-quality + narrow unlabeled data**)
6. 10k labeled data from hopper-random + unlabeled 1M data from hopper-expert (i.e., **low-quality + broad labeled data, high-quality + narrow unlabeled data**)

Results. In cases (1) and (2), adding the unlabeled random or medium data, should increase coverage, since the labeled data only consists of expert transitions. Moreover, the induced reward bias 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. Therefore, we expect the benefits of coverage to outweigh any reward bias, and as shown in Table 10, we find that UDS does help.

In cases (4), (5) and (6), when the relabeled data is better compared to the labeled data (i.e., expert or medium), we find that even if the rewards on these transitions are incorrect, behavior regularization properties induced by offline RL algorithms allow UDS to attain better performance than no sharing by utilizing the unlabeled data.

In case (3), we find that UDS hurts compared to No Sharing. This is because the target task 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.

G.3 Summary of empirical analysis

Given our results in Table 9 and Table 10, we summarize the applicability of UDS under different scenarios in Table 11 below.

Environment	Labeled dataset type / size	Unlabeled dataset type / size	UDS (ours)	No Sharing
D4RL hopper [20]	expert / 10k transitions	random / 1M transitions	90.8	77.1
	expert / 10k transitions	medium / 1M transitions	87.6	77.1
	medium / 10k transitions	random / 1M transitions	9.8	28.7
	medium / 10k transitions	expert / 1M transitions	106.1	28.7
	random / 10k transitions	medium / 1M transitions	51.9	9.6
	random / 10k transitions	expert / 1M transitions	97.0	9.6

Table 10: We perform an empirical analysis on the hopper environment from the D4RL [20] 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 three random seeds. UDS outperforms No Sharing in 5 out of 6 settings, suggesting that UDS is robust in different combinations of data quality and coverage of both labeled and unlabeled data. Note that UDS fails in the setting where the labeled data is of medium data quality and the unlabeled data is random, suggesting that sharing data in settings where the labeled data is limited and of low quality and the unlabeled data is also of poor quality is not useful.

Scenarios	UDS	Intuition
L: limited + high-quality + narrow, U: abundant + low-quality + broad	✓	increase coverage
L: limited + high-quality + narrow, U: abundant + medium-quality + narrow	✓	more negatives
L: limited + medium-quality + narrow, U: abundant + low-quality + broad	✗	reward bias outweighs high coverage
L: abundant + medium-quality + broad, U: abundant + medium-quality + broad	✗	reward bias outweighs high coverage
L: limited + medium-quality + narrow, U: abundant + high-quality + narrow	✓	increase data quality
L: limited + low-quality + broad, U: abundant + medium-quality + narrow	✓	increase data quality
L: limited + low-quality + broad, U: abundant + high-quality + narrow	✓	increase data quality

Table 11: 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 (note that these are not absolute numbers and hard to precisely quantify without access to the problem domain, but a highly skewed ratio of the amount of labeled and unlabeled data might help characterize it as limited/abundant). 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.