DexDICE: Reconciling Safety and Performance via Dual-Expert Offline Imitation Learning

Seokin Seo[1,2], **Sung Kuk Shyn**[2], **Minji Seo**[1,3],
Yisoo Lee[1†], **Jongmin Lee**[4†], **Hongseok Yang**[5†], **Kee-Eung Kim**[2†]

[1]Korea Institute of Science and Technology (KIST),
[2]Korea Advanced Institute of Science and Technology (KAIST),
[3]Hyundai Motor Company, [4]Yonsei University,
[5]Korea Institute for Advanced Study (KIAS)

Abstract

Ensuring high returns while providing strong safety guarantees in offline imitation learning (IL) is a fundamental challenge: existing approaches either rely on explicitly specified cost functions, which are difficult to design for complex real-world safety constraints, or require expert demonstrations that are both safe and high-performing, which are often too costly to collect. We introduce *dual-expert imitation learning*, which leverages two complementary data sources: (i) safety-compliant demonstrations that faithfully follow safety guidelines but may be suboptimal, and (ii) performance-oriented demonstrations that achieve high returns but may violate safety guidelines. To address this problem, we propose DexDICE (**D**ual-**ex**pert offline imitation learning via stationary **DI**stribution **C**orrection **E**stimation). Instead of conventional distribution constraints, DexDICE combines DICE-style offline learning with safe support constraints, enabling agents to exploit high-return behaviors while remaining within safety-compliant regions. We prove that the ideal solution satisfying this constraint exactly has zero unsafe-region visitation with high probability. We evaluate DexDICE on a re-curated DSRL benchmark and validate it on a real-world mobile-robot navigation task trained directly from human teleoperation demonstrations, where it substantially outperforms baselines and remains robust to data scarcity and quality degradation.[1]

1 Introduction

Imitation learning (IL) is a powerful paradigm for sequential decision-making that learns directly from expert demonstrations [21, 25, 67], bypassing the reward specification required in reinforcement learning (RL). IL has enabled sophisticated behaviors in real-world domains such as robotic manipulation [17, 30, 63] and autonomous driving [19, 37]. In many such domains, however, collecting additional on-policy interaction data can be expensive or impractical due to hardware wear-and-tear, limited access to the system, and slow or costly resets, motivating offline IL methods that learn entirely from fixed datasets [27, 58]. Since expert demonstrations are often scarce, recent offline IL methods also leverage auxiliary unlabeled trajectories that, while potentially suboptimal, provide valuable coverage of transition information [31, 45, 64, 66].

Separately, when IL policies are deployed in safety-critical applications, ensuring safety remains an open challenge. A common strategy is to encode safety requirements via explicit cost functions

[†]Equal advising.
[1]Code: `https://github.com/tzs930/dexdice`

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

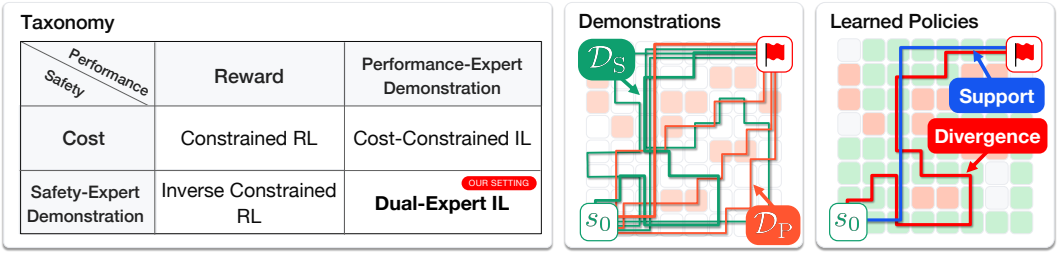


Figure 1: (left) Taxonomy of safe sequential decision-making problems by available performance and safety signals. (middle) Dual-expert IL instance: $\mathcal{D}_P$ prioritizes fast goal reaching but is possibly unsafe, while $\mathcal{D}_S$ stays safe but possibly suboptimal. (right) Prior offline IL relies on *divergence regularization* (e.g., behavior regularization), biasing the policy toward $\mathcal{D}_S$'s suboptimal behaviors (red path); DexDICE instead imposes a *support constraint* from $\mathcal{D}_S$ (blue path), recovering $\mathcal{D}_P$'s near-optimal route within the safe support.

and solve a constrained optimization problem [14, 59, 65]. However, specifying such costs can be as difficult as reward design: safety desiderata are often nuanced, context-dependent, and hard to instrument. Inverse constraint learning (ICL) [24, 40, 46] offers an alternative by inferring safety constraints from demonstrations, but many ICL methods still rely on access to reward signals that IL does not assume. These limitations motivate methods that can jointly optimize safety and performance solely from demonstration data, without requiring an explicitly specified cost function.

In practice, demonstrations that are simultaneously safe and high-performing are difficult to obtain from a single expert. Moreover, simply pooling demonstrations from multiple experts (e.g., some prioritizing safety and others prioritizing performance) does not resolve the issue, since it introduces the challenging problem of disentangling safety and performance signals within the resulting dataset. A more realistic strategy is therefore to separately collect different types of expert demonstrations, one focused on safety and the other focused on performance [35]. This separation provides clearer supervision for each objective, making it easier for offline IL algorithms to leverage both effectively. Such dual-source datasets are common in practice: for example, autonomous driving data contain high-speed but risky maneuvers, alongside safe and conservative driving behaviors [3, 10]; similarly, in robotics and healthcare, some demonstrations are efficiency-oriented while others explicitly prioritize safety [34, 36, 47, 49].

Motivated by these observations, we introduce a novel problem setting, **dual-expert imitation learning**, in which the learner has access to two types of expert demonstrations: *safe demonstrations*, which adhere to safety constraints but may be suboptimal in task performance, and *performant demonstrations*, which achieve high returns but may violate safety constraints. We further consider **dual-expert offline IL**, a more restrictive setting that forbids online interaction during training; in lieu of online rollouts, we incorporate *unlabeled demonstrations* that provide broader transition coverage, following prior work on offline IL [26, 31, 64].

To address this setting, we propose *DexDICE* (**D**ual-**ex**pert offline imitation learning via stationary **DI**stribution **C**orrection **E**stimation), which extends the DICE framework [31] with a *support constraint* derived from the safe demonstrations. The resulting policy is the closest approximation to the performant expert's state-action distribution within the estimated safe support, with a high-probability guarantee on zero unsafe-region visitation for the ideal support-constrained policy (Theorem 3).

Our main contributions are summarized below:

- We formalize *dual-expert offline imitation learning*, a problem class that disentangles safety and performance objectives through complementary demonstrations alone, without the explicit reward or cost labels required by prior methods.
- We propose *DexDICE*, the first DICE-based offline IL algorithm to replace distributional regularization toward d_S with an explicit *safe-support constraint*, decoupling the learned policy's performance from the safe expert's suboptimality.
- We empirically validate DexDICE on a re-curated DSRL benchmark [44], where it achieves superior (i) safety-performance trade-off, (ii) data efficiency, and (iii) robustness to variations in data quality, and further demonstrate (iv) real-robot applicability with human teleoperation demonstrations.

2 Related Work

Offline Safe RL Offline safe RL (OSRL) [18, 39, 42, 43, 70] requires per-transition reward $r(s, a)$ and cost $c(s, a)$, which the dual-expert IL setting does not provide. To naively apply these methods, we would need to estimate reward and costs from the given demonstrations. However, such reward/cost estimates are unreliable especially in our setting where $\mathcal{D}_{\mathrm{U}}$ is a latent mixture of safe and unsafe transitions, turning reward/cost recovery into a hard PU-learning problem [5] (Appendix B.2). We evaluate recent OSRL methods CDT [43] and TraC [18] with pseudo-labels $\hat{r}, \hat{c}$ estimated from the $(\mathcal{D}_{\mathrm{P}}, \mathcal{D}_{\mathrm{S}}, \mathcal{D}_{\mathrm{U}})$ partition as matched-information baselines. For an empirical comparison with OSRL methods under their original (ground-truth) information access, see Appendix G.3.

Safe Imitation Learning Existing safe IL methods assume either explicit cost or constraint signals [11, 14, 59] or model-based safety constructs such as Hamilton-Jacobi reachability [4] and control barrier functions [2], both of which require system dynamics or hand-specified constraints unavailable in our offline setting. Beyond cost-aware methods, some safe IL approaches sidestep cost signals by leveraging undesirable demonstrations [9, 23, 26], while an online dual-expert method such as Safe CoR [35] shares our two-set demonstration partition but additionally requires environment interaction with explicit reward and cost signals.

Offline Imitation Learning Offline IL is essential when environment interaction is restricted and reward specification is difficult, and mixed-quality datasets are commonly leveraged to supplement scarce transition samples [22, 31, 41, 64, 66]. DWBC [64] reweights behavioral cloning via a PU-learning discriminator, while DICE-based methods such as DemoDICE [31] and RelaxDICE [66] cast IL as convex distribution matching over stationary state-action distributions. The closest cost-aware comparator is the offline safe IL variant of OSRL method FISOR [70] (henceforth *FISOR-IL*), which uses per-transition cost labels; we evaluate it under matched-information access by supplying pseudo-cost $\hat{c}$ derived from the $(\mathcal{D}_{\mathrm{P}}, \mathcal{D}_{\mathrm{S}}, \mathcal{D}_{\mathrm{U}})$ partition.

Inverse Constrained Learning (ICL) ICL methods [24, 33, 40, 46, 60] and their offline extension ICSDICE [51] infer a latent cost function from demonstrations and then optimize against it. This indirection is inherently ill-posed due to non-uniqueness of admissible cost functions; we instead learn safe policies directly from dual-expert demonstrations, bypassing cost recovery entirely.

Figure 1 positions our contribution. Prior DICE-based offline IL enforces safety via *divergence regularization*, which can bias the policy toward suboptimal safe behaviors (red path). DexDICE instead enforces safety via an explicit *support constraint* (blue path), and to our knowledge is the first such formulation among DICE-based offline IL methods. See Appendix B for extended discussion.

3 Problem Definition

3.1 Markov Decision Processes

We consider a Markov decision process (MDP) defined as a tuple $\mathcal{M} := \langle \mathcal{S}, \mathcal{A}, T, \rho_0, r, \gamma \rangle$, where $\mathcal{S}$ and $\mathcal{A}$ denote the state and action spaces respectively, $T : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ is the transition probability function, $\rho_0 \in \Delta(\mathcal{S})$ is the initial state distribution, $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, and $\gamma \in (0, 1)$ is the discount factor. We define the policy space as $\Pi := \{\pi : \mathcal{S} \to \Delta(\mathcal{A})\}$. Given policy $\pi \in \Pi$, the stationary distribution d_π is defined as:

$$d_\pi(s, a) := (1 - \gamma) \sum_{t=0}^{\infty} \gamma^t \mathbb{P}(s_t = s, a_t = a),$$

where $s_0 \sim \rho_0$, $a_t \sim \pi(s_t)$, and $s_{t+1} \sim T(s_t, a_t)$. Then, d_π satisfies the Bellman flow constraint:

$$\sum_a d_\pi(s, a) = (1 - \gamma)\rho_0(s) + \gamma \sum_{\bar{s}, \bar{a}} T(s|\bar{s}, \bar{a}) d_\pi(\bar{s}, \bar{a}), \forall s \in \mathcal{S}.$$

A constrained Markov decision process (CMDP) [1] extends the standard MDP framework to incorporate safety constraints. We define a CMDP as $\mathcal{M}_c := \langle \mathcal{M}, c, \delta_c \rangle$, where $\mathcal{M}$ is the unconstrained MDP, $c : \mathcal{S} \times \mathcal{A} \to \mathbb{R}_{\geq 0}$ is a non-negative cost function, and $\delta_c \geq 0$ is a cost limit. Since we only observe demonstrations, neither the reward function r nor the cost function c is available in our setting.

3.2 Imitation Learning

Imitation learning (IL) aims to recover an expert policy from demonstrations in the absence of reward signals. We denote the demonstration dataset as $\mathcal{D} = \{(s_i, a_i)\}_{i=1}^N$, a collection of state-action pairs sampled from expert trajectories. A straightforward approach to IL is behavioral cloning (BC), which treats the problem as supervised learning and solves it by minimizing the negative log-likelihood:

$$\min_{\pi} \quad \mathbb{E}_{(s,a)\sim\mathcal{D}} \left[-\log \pi(a|s) \right].$$

However, BC suffers from compounding errors due to covariate shift, particularly over long horizons [53]. To address this limitation, recent IL algorithms typically minimize a distribution matching objective:

$$\min_{\pi} \quad \mathbb{D}(d_\pi(s,a), d_{\mathcal{D}}(s,a)),$$

where $\mathbb{D}$ is a divergence measure (e.g., f-divergences [21, 31, 68] and integral probability metrics [15, 32]) and $d_{\mathcal{D}}$ denotes the state-action distribution in $\mathcal{D}$.

In many practical applications, on-policy interaction with the environment can be prohibitively expensive, giving rise to *offline IL*. Since expert data is often scarce, offline IL methods leverage auxiliary unlabeled demonstrations $\mathcal{D}_{\mathrm{U}}$, which may be suboptimal but provide valuable transition information [31, 45, 64]. While standard IL assumes demonstrations from a single expert, real-world scenarios often involve separate sources for safety and performance, motivating the dual-expert formulation introduced in the next section.

3.3 Dual-Expert Imitation Learning

We consider *dual-expert IL* from safety-performance expert demonstrations, consisting of a set of safety-expert (*safe*) demonstrations $\mathcal{D}_{\mathrm{S}}$, which satisfy the underlying constraints but may be suboptimal, and a set of performance-expert (*performant*) demonstrations $\mathcal{D}_{\mathrm{P}}$, which achieve high returns but may violate unknown safety constraints.

Denote the underlying state-action distributions of $\mathcal{D}_{\mathrm{S}}, \mathcal{D}_{\mathrm{P}}$ as $d_{\mathrm{S}}, d_{\mathrm{P}}$. Since $\mathcal{D}_{\mathrm{S}}$ may contain suboptimal demonstrations, we aim to recover only its safety-compliant support of state-action pairs rather than matching the distribution d_{S}. Accordingly, we seek a policy π that aligns with performant behaviors while remaining within the safe support. Formally, the goal of dual-expert IL is to recover a policy π whose stationary distribution d_π matches d_{P} with a support constraint derived from d_{S}:

$$\min_{\pi} \quad \mathbb{D}(d_\pi \| d_{\mathrm{P}}) \quad \text{s.t.} \quad \mathrm{supp}(d_\pi) \subseteq \mathrm{supp}(d_{\mathrm{S}}), \tag{1}$$

where $\mathrm{supp}(d) = \{(s,a) \mid d(s,a) > 0\}$. We refer to this constraint as the *safe support constraint*.

To ensure the optimization problem is well-posed, we assume that there exists a policy π whose stationary distribution d_π satisfies $\mathrm{supp}(d_\pi) \subseteq \mathrm{supp}(d_{\mathrm{S}}) \cap \mathrm{supp}(d_{\mathrm{P}})$. This requirement is weaker than assuming access to demonstrations that are simultaneously safe and performant, which corresponds to the special case $d_{\mathrm{S}} = d_{\mathrm{P}}$ (practical examples in Appendix A). When the overlap $\mathrm{supp}(d_{\mathrm{S}}) \cap \mathrm{supp}(d_{\mathrm{P}})$ is small, the optimum may be far from d_{P} in divergence, but the formulation still returns the closest safe approximation to d_{P}.

Equivalently, this is a hard cost-constrained IL problem [59] with $c(s,a) = 0$ on $\mathrm{supp}(d_{\mathrm{S}})$ and $c(s,a) > 0$ otherwise; the support constraint enforces $\mathbb{E}_{d_\pi}[c(s,a)] = 0$ implicitly (see Appendix A).

4 DexDICE: Dual-Expert Offline IL with Safe Support Constraint

We study the offline version of the dual-expert IL problem, where on-policy interaction is prohibited during training. In addition to $\mathcal{D}_{\mathrm{S}}$ and $\mathcal{D}_{\mathrm{P}}$, we assume access to a more abundant unlabeled dataset $\mathcal{D}_{\mathrm{U}}$ sampled from distribution d_{U}, which provides coverage of transition information. Following the standard positive-unlabeled convention, $\mathcal{D}_{\mathrm{U}}$ is constructed to include $\mathcal{D}_{\mathrm{S}}$ and $\mathcal{D}_{\mathrm{P}}$ as subsets in our experiments. Accordingly, we assume $d_{\mathrm{S}} \ll d_{\mathrm{U}}$ and $d_{\mathrm{P}} \ll d_{\mathrm{U}}$ at the population level, so that the density ratios $d_{\mathrm{S}}/d_{\mathrm{U}}$ and $d_{\mathrm{P}}/d_{\mathrm{U}}$ used below are well-defined d_{U}-almost everywhere. Here d_{U} is neither a safety nor a performance target but a common offline reference: it supplies transition coverage for learning the Bellman flow constraints offline without safety or performance annotation [31], places both expert sources on a common scale through the ratios $d_{\mathrm{S}}/d_{\mathrm{U}}$ and $d_{\mathrm{P}}/d_{\mathrm{U}}$ (Section 4.2), and defines the DICE correction $w = d/d_{\mathrm{U}}$, which requires $d^\pi \ll d_{\mathrm{U}}$ (Assumption D).

4.1 Support-Constrained DICE Objective

Since direct sampling from d_π is not allowed in the offline setting, we adopt the DICE (Distribution Correction Estimation) framework [38, 48], which formulates policy optimization as convex optimization over stationary distributions. The key idea is to introduce a correction ratio $w(s,a) := d(s,a)/d_{\mathrm{U}}(s,a)$, enabling off-policy optimization using only static datasets [31, 39]. Following this, we optimize over d instead of π.

Unlike conventional DICE methods [31] that use a divergence regularization (e.g., $D_{\mathrm{KL}}(d\|d_{\mathrm{S}})$) and may bias the policy toward suboptimal safe behaviors, our support constraint restricts only *where* the policy can act, not *how often*, decoupling safety from suboptimality (validated in a tabular experiment in Appendix C).

To enforce the safe support constraint $\mathrm{supp}(d) \subseteq \mathrm{supp}(d_{\mathrm{S}})$ in the objective (1), we reformulate the constraint as an equality condition:

$$\Big(\mathrm{supp}(d) \subseteq \mathrm{supp}(d_{\mathrm{S}})\Big) \Longleftrightarrow \sum_{s,a} \mathbb{1}_{d_{\mathrm{S}}}(s,a) d(s,a) = 1,$$

where $\mathbb{1}_{d_{\mathrm{S}}}(s,a) = 1$ if $(s,a) \in \mathrm{supp}(d_{\mathrm{S}})$ and 0 otherwise; the equivalence uses $\sum_{s,a} d(s,a) = 1$ since d is a probability distribution. Then, our objective is to find a stationary distribution d that minimizes divergence to d_{P} while satisfying the safe support constraint expressed as above. To ensure that d corresponds to a valid policy, we impose Bellman flow constraints and arrive at the following optimization problem:

$$\begin{aligned} \min_{d} \; D_{\mathrm{KL}}(d\|d_{\mathrm{P}}) \qquad \text{s.t.} \qquad & \sum_{s,a} \mathbb{1}_{d_{\mathrm{S}}}(s,a) d(s,a) = 1 \\ & (\mathcal{B}_* d)(s) = (1-\gamma)\rho_0(s) + \gamma(\mathcal{T}_* d)(s), \qquad \forall s \in \mathcal{S} \\ & d(s,a) \geq 0, \qquad \forall s \in \mathcal{S}, a \in \mathcal{A}, \end{aligned} \tag{2}$$

where $(\mathcal{B}_* d)(s) := \sum_a d(s,a)$ is a marginalization operator and $(\mathcal{T}_* d)(s) := \sum_{\bar{s},\bar{a}} T(s|\bar{s},\bar{a}) d(\bar{s},\bar{a})$ is a transposed Bellman operator.

To unconstrain the problem, we introduce Lagrange multipliers λ for the safe-support constraint and $\nu(s)$ for the Bellman flow constraints, flip the sign, and apply the change of variables $w(s,a) := d(s,a)/d_{\mathrm{U}}(s,a)$ together with the density ratio $r_{\mathrm{P}}(s,a) := d_{\mathrm{P}}(s,a)/d_{\mathrm{U}}(s,a)$. This yields the dual objective (full derivation in Appendix D.1):

$$L(w,\nu,\lambda) := (1-\gamma)\mathbb{E}_{s\sim\rho_0}[\nu(s)] - \lambda + \mathbb{E}_{(s,a)\sim d_{\mathrm{U}}}\big[w(s,a)\cdot(A_{\nu,\lambda}(s,a) - \log w(s,a))\big], \tag{3}$$

with advantage $A_{\nu,\lambda}(s,a) := \gamma(\mathcal{T}\nu)(s,a) - \nu(s) + \log r_{\mathrm{P}}(s,a) + \lambda \mathbb{1}_{d_{\mathrm{S}}}(s,a)$, where $(\mathcal{T}\nu)(s,a) := \mathbb{E}_{s'\sim T(s,a)}[\nu(s')]$. Intuitively, $A_{\nu,\lambda}$ captures the temporal-difference structure and encourages matching d_{P} via the performance proxy $\log r_{\mathrm{P}}$, while $\lambda \mathbb{1}_{d_{\mathrm{S}}}$ relatively up-weights w^* inside $\mathrm{supp}(d_{\mathrm{S}})$ by a factor e^λ through the closed form $w^*_{\nu,\lambda} = \exp(A_{\nu,\lambda} - 1)$ (Eq. 6). In practice, $A_{\nu,\lambda}$ is estimated from (s,a,s') tuples in $\mathcal{D}_{\mathrm{U}}$ via $(\mathcal{T}\nu)(s,a) \approx \nu(s')$, while r_{P} and $\mathbb{1}_{d_{\mathrm{S}}}$ are obtained via discriminator training, which we revisit in Section 4.2.

Then, our optimization objective becomes:

$$\min_{\nu,\lambda} \max_{w\geq 0} L(w,\nu,\lambda). \tag{4}$$

Proposition 1. *(Strong Duality) Under the constraint qualification stated in Appendix D, $L(w,\nu,\lambda)$ satisfies strong duality, which ensures:*

$$\max_{w\geq 0} \min_{\nu,\lambda} L(w,\nu,\lambda) = \min_{\nu,\lambda} \max_{w\geq 0} L(w,\nu,\lambda). \tag{5}$$

Proposition 2. *(Closed-form solution; Lee et al. [39]) The inner maximization on the RHS of Eq. (5) admits a closed-form solution:*

$$w^*_{\nu,\lambda}(s,a) = \exp(A_{\nu,\lambda}(s,a) - 1). \tag{6}$$

See Appendix D for the proofs of these propositions. Hence, the problem is converted into the following minimization problem w.r.t ν, λ:

$$\begin{aligned} &\min_{\nu,\lambda} \; \mathbb{E}_{(s,a)\sim d_{\mathrm{U}}}[w^*(s,a)\cdot(-\log w^*(s,a) + A_{\nu,\lambda}(s,a))] + (1-\gamma)\mathbb{E}_{s\sim\rho_0}[\nu(s)] - \lambda \\ =&\min_{\nu,\lambda} \; \mathbb{E}_{(s,a)\sim d_{\mathrm{U}}}[\exp(A_{\nu,\lambda}(s,a) - 1)] + (1-\gamma)\mathbb{E}_{s\sim\rho_0}[\nu(s)] - \lambda =: L(\nu,\lambda). \end{aligned} \tag{7}$$

We refer to the method that optimizes the objective in Eq. (7) as DexDICE, short for **D**ual-**ex**pert offline IL via stationary **DI**stribution **C**orrection **E**stimation.

4.2 Support Estimation via Density Ratios

To optimize the objective in Eq. (7), we need an estimator $\hat{\mathbb{1}}_{d_\mathrm{S}}$ of the support indicator $\mathbb{1}_{d_\mathrm{S}}$. Directly estimating the density d_S is intractable in high-dimensional state-action spaces; we instead estimate a bounded density *ratio* via classification, which scales more favorably with dimensionality [20].

The Bayes-optimal classifier of safe vs. unsafe state-action pairs is a likelihood-ratio test on $d_\mathrm{S}/d_\mathrm{N}$, where d_N is the unsafe distribution. However, d_N is not directly observable in our setting. We therefore estimate the *safe-unlabeled* ratio $d_\mathrm{S}/d_\mathrm{U}$ from $\mathcal{D}_\mathrm{S}$ and $\mathcal{D}_\mathrm{U}$, and relate it to $d_\mathrm{S}/d_\mathrm{N}$ via the PU-learning mixture [57]: writing $d_\mathrm{U} = \eta\, d_\mathrm{S} + (1-\eta)\, d_\mathrm{N}$ with class prior η, rearrangement yields

$$\frac{d_\mathrm{N}(s,a)}{d_\mathrm{S}(s,a)} = \frac{1}{(1-\eta)}\frac{d_\mathrm{U}(s,a)}{d_\mathrm{S}(s,a)} - \frac{\eta}{1-\eta}, \tag{8}$$

showing that the unsafe-safe ratio is a monotone transformation of the unlabeled-safe ratio, rendering the two ratios equivalent as decision functions up to a threshold shift.

Following Kim et al. [31], we train two discriminators: $\hat{C}_\mathrm{S}$ estimates $\hat{r}_\mathrm{S} \approx d_\mathrm{S}/d_\mathrm{U}$ (used to construct $\hat{\mathbb{1}}_{d_\mathrm{S}}$), and $\hat{C}_\mathrm{P}$ estimates $\hat{r}_\mathrm{P} \approx d_\mathrm{P}/d_\mathrm{U}$ (the performance proxy in $A_{\nu,\lambda}$). Training details are in Appendix F.1. Compared to estimators that use only safe demonstrations, this safe-vs-unlabeled discriminator empirically yields more robust support estimates (Appendix G.7).

We do not directly invert Eq. (8) since the class prior η is notoriously hard to identify [13, 16]. Instead, we exploit *rank consistency*: the monotone relationship in Eq. (8) implies that $\hat{r}_\mathrm{S}$ alone induces the same ordering over state-action pairs as the unsafe-safe ratio, so thresholding $\hat{r}_\mathrm{S}$ suffices. To this end, we construct a non-trivial support estimator $\hat{\mathbb{1}}_{d_\mathrm{S}}$ of $\mathbb{1}_{d_\mathrm{S}}$ by specifying a target false-negative rate (FNR) $\kappa \in (0,1)$, i.e., the fraction of safe pairs incorrectly rejected. Specifically, we set the threshold τ_κ as the κ-quantile of $\hat{r}_\mathrm{S}$ over $\mathcal{D}_\mathrm{S}$:

$$\tau_\kappa := \mathrm{Quantile}_\kappa\big(\{\hat{r}_\mathrm{S}(s,a) : (s,a) \in \mathcal{D}_\mathrm{S}\}\big) = \inf\Big\{\tau : \frac{1}{|\mathcal{D}_\mathrm{S}|}\sum_{(s,a)\in\mathcal{D}_\mathrm{S}} \mathbb{1}\big(\hat{r}_\mathrm{S}(s,a) \le \tau\big) \ge \kappa\Big\}. \tag{9}$$

We then define the support estimator by thresholding $\hat{r}_\mathrm{S}$: $\hat{\mathbb{1}}_{d_\mathrm{S}}(s,a) := \mathbb{1}\big\{\hat{r}_\mathrm{S}(s,a) \ge \tau_\kappa\big\}$, so that a pair is accepted when $\hat{r}_\mathrm{S}(s,a) \ge \tau_\kappa$ and rejected when $\hat{r}_\mathrm{S}(s,a) < \tau_\kappa$. By the definition of the empirical quantile, the empirical FNR on $\mathcal{D}_\mathrm{S}$ (the fraction of $\mathcal{D}_\mathrm{S}$ with $\hat{r}_\mathrm{S} < \tau_\kappa$) is strictly below κ.

4.3 Theoretical Analysis

Although the support estimator $\hat{\mathbb{1}}_{d_\mathrm{S}}$ in Section 4.2 is calibrated to a target FNR by construction, this does not by itself imply that a policy constrained to the estimated support avoids the unsafe region at deployment. We now show that this calibration translates to zero unsafe-region visitation by the ideal support-constrained policy.

Setup. Let $\mathbf{S} = \mathrm{supp}(d_\mathrm{S})$ and $\mathbf{N} = \mathrm{supp}(d_\mathrm{U}) \setminus \mathbf{S}$ denote the safe and (reachable) unsafe regions, and let $r^*(s,a) := d_\mathrm{S}/(d_\mathrm{S}+d_\mathrm{U})$ be the Bayes-optimal classifier of $\mathbf{S}$ vs. $\mathbf{N}$. The estimated safe support at threshold τ is $\hat{\mathbf{S}}_\tau = \{(s,a) : \hat{r}_\mathrm{S}(s,a) \ge \tau\}$, with false-negative and false-positive rates $\mathrm{FNR}(\tau), \mathrm{FPR}(\tau)$ (edge cases on $\mathbf{N}$ vs. $\mathbf{S}^c \setminus \mathrm{supp}(d_\mathrm{U})$ are in Appendix E).

Assumption. We assume the demonstrated safe support is safety-compliant, so that staying within $\mathrm{supp}(d_\mathrm{S})$ implies environmental safety (pointwise soundness; Appendix E). We consider an ideal support-constrained policy π_κ with $\mathrm{supp}(d^{\pi_\kappa}) \subseteq \hat{\mathbf{S}}_{\tau_\kappa}$, where τ_κ is the empirical κ-quantile in Eq. (9) for a user-specified $\kappa \in (0,1)$, computed from calibration pairs in $\mathcal{D}_\mathrm{S}$ drawn i.i.d. from d_S (Assumption B). Let $\varepsilon > 0$ bound the worst-case discriminator error over all state-action pairs, and let $\delta \in (0,1)$ be the total failure probability of the guarantee, split equally between the discriminator and the finite calibration sample. We assume $\|\hat{r}_\mathrm{S} - r^*\|_\infty \le \varepsilon$ with probability at least $1 - \delta/2$ over the training data (Assumption C). For fixed δ, ε vanishes as the sample size grows.

The following theorem characterizes the unsafe-region visitation $d^{\pi_\kappa}(\mathbf{N})$ in terms of the discriminator error ε, the failure probability δ, and user-specified κ.

Theorem 3 (Safety guarantee of DexDICE; informal)**.** *Under i.i.d. calibration sampling and standard regularity conditions on the discriminator estimation error, the DICE weights, and a safe-margin condition (Assumptions B–E in Appendix E), if $\kappa > \kappa_0 + O(|\mathcal{D}_{\mathrm{S}}|^{-1/2})$, the support-constrained policy π_κ visits the unsafe region with zero probability on a high-probability event:*

$$d^{\pi_\kappa}(\mathbf{N}) = 0 \quad \textit{with probability } \geq 1 - \delta.$$

The *boundary mass* κ_0 is the d_{S}-probability of safe pairs with low Bayes-optimal discriminator scores (formal definition in Appendix E). The condition $\kappa > \kappa_0 + O(|\mathcal{D}_{\mathrm{S}}|^{-1/2})$ requires κ to exceed this mass by a finite-sample slack that vanishes as $|\mathcal{D}_{\mathrm{S}}| \to \infty$.

Key findings. **(i) Exact-zero safety with high probability.** Once the discriminator error ε falls below half the safe margin, unsafe visitation drops to *exact zero*, since $r^* \equiv 0$ on $\mathbf{N}$ (Lemma G) converts sub-margin error into a hard zero. This does not require perfect discriminator estimation: under the stated assumptions, $\tau_\kappa > \varepsilon$ excludes every pair in $\mathbf{N}$ from the estimated support, yielding zero visitation for the ideal support-constrained policy.

(ii) Single-knob safety calibration. By construction, κ bounds the empirical FNR on $\mathcal{D}_{\mathrm{S}}$ strictly below κ. Under the conditions of Theorem 3, the same calibration yields $\mathrm{FPR}(\tau_\kappa) = 0$ with high probability (Corollary I). Within the range covered by the theorem, varying κ adjusts safe-pair retention while the FPR remains zero.

Detailed proofs and supporting lemmas are deferred to Appendix E.

Practical algorithm. In summary, $\hat{r}_{\mathrm{P}}, \hat{r}_{\mathrm{S}}$ are trained from $\mathcal{D}_{\mathrm{P}}, \mathcal{D}_{\mathrm{S}}, \mathcal{D}_{\mathrm{U}}$, and $\hat{\mathbb{1}}_{d_{\mathrm{S}}}$ is built from $\hat{r}_{\mathrm{S}}$ via FNR calibration. Plugging these into $L(\nu, \lambda)$ (Eq. 7), we jointly optimize ν, λ to obtain $w^*_{\nu,\lambda}$ (Eq. 6) and recover the policy from $w^*_{\nu,\lambda}$ via weighted BC, following prior DICE frameworks [31, 39] (Appendix F.3; full algorithm in Appendix F.4). This step introduces a theory–practice gap: Theorem 3 concerns the ideal solution, whereas the extracted weighted-BC policy is a Gaussian with full support over the action space and hence cannot be hard-supported on $\hat{\mathbf{S}}_{\tau_\kappa}$. A deployment-time projection of each proposed action onto $\hat{\mathbf{S}}_{\tau_\kappa}$ substantially reduces the measured support mismatch but cannot eliminate it in general, since a reached state need not admit any in-support action and a finite-budget search may fail to find one even when it exists; the residual mismatch is therefore reported separately and is not covered by the exact guarantee (Appendix G.11).

5 Experiments

Our experiments aim to answer the following questions:

- **Q1**: How effective are offline IL methods in the dual-expert demonstration setting?
- **Q2**: How data-efficient are these methods with respect to the sizes of $\mathcal{D}_{\mathrm{S}}$ and $\mathcal{D}_{\mathrm{P}}$?
- **Q3**: How robust are they to variations in demonstration quality (i.e., the proportion of safe-performant demonstrations) within $\mathcal{D}_{\mathrm{S}}$ and $\mathcal{D}_{\mathrm{P}}$?
- **Q4**: Does DexDICE remain effective on a real-world robot when trained from human-collected demonstrations?

Because dual-expert imitation learning is a problem setting we newly propose, no dedicated public benchmark currently exists. We therefore re-curate the offline safe RL benchmark DSRL [44] by relabeling each trajectory along two binary axes (performant vs. non-performant; safe vs. unsafe) based on predefined return and cost thresholds, and refer to this adapted dataset as the *re-curated DSRL benchmark*. These labels are used solely to construct $\mathcal{D}_{\mathrm{P}}$ and $\mathcal{D}_{\mathrm{S}}$; DexDICE itself never consumes reward or cost signals during training. For **Q1**, we use this re-curated DSRL benchmark, which spans a broad range of performance and safety levels. For **Q2** and **Q3**, which require precise control over dataset size and quality, we construct *controlled datasets* on Safety Gymnasium [28] by systematically mixing four distinct demonstration types. For **Q4**, we collect human teleoperation demonstrations on a physical mobile robot for real-world navigation evaluation.

Since dual-expert offline IL is a new problem setting with no directly comparable baselines, we adapt a range of established offline IL and offline safe RL methods with matched information access:

Task Group	Metric	BC	FBC	JFBC	DemoDICE-S	DemoDICE-U	DWBC-S	FISOR-IL ($\hat{c}$)	CDT ($\hat{r}, \hat{c}$)	TraC ($\hat{r}, \hat{c}$)	DexDICE (Ours)
Safety Gymnasium (21 tasks)	Avg return	0.46	0.13	0.59	0.27	0.47	0.45	0.58	0.43	0.19	0.37
	Avg cost	1.99	1.44	2.29	0.68	1.84	1.74	2.25	1.46	0.86	0.71
	# Safe	3/21	7/21	2/21	18/21	10/21	5/21	3/21	7/21	14/21	17/21
	# Best among safe	0/21	1/21	2/21	4/21	5/21	2/21	3/21	2/21	3/21	**9/21**
Bullet Safety Gym (8 tasks)	Avg return	0.57	0.44	0.73	0.46	0.63	0.62	0.60	0.44	0.35	0.59
	Avg cost	1.47	0.98	1.62	1.02	1.41	1.41	1.41	0.99	0.53	0.63
	# Safe	1/8	4/8	1/8	5/8	2/8	2/8	1/8	4/8	6/8	7/8
	# Best among safe	0/8	2/8	1/8	0/8	0/8	1/8	1/8	2/8	3/8	**5/8**
MetaDrive (9 tasks)	Avg return	0.76	0.25	0.70	0.33	0.71	0.74	0.62	0.23	0.23	0.50
	Avg cost	1.68	0.50	1.85	0.28	1.79	1.69	1.68	0.04	0.04	0.69
	# Safe	1/9	8/9	0/9	9/9	0/9	0/9	0/9	9/9	9/9	7/9
	# Best among safe	1/9	2/9	0/9	**5/9**	0/9	0/9	0/9	4/9	3/9	**5/9**

Table 1: Safety–performance comparison on the re-curated DSRL benchmark (38 tasks), grouped by environment family. Return and cost are normalized and averaged over 5 seeds and over the tasks in each group (safe when cost ≤ 1; a lower cost is not preferred once the constraint is met). # Safe counts tasks satisfying the cost constraint, and # Best among safe counts tasks on which a method attains the highest return among safe methods (ties within one standard error). All methods use $N_{\mathcal{D}_\mathrm{S}}{=}N_{\mathcal{D}_\mathrm{P}}{=}10$; DexDICE uses a single fixed $\kappa = 0.01$ for all tasks without per-task tuning. Gray: group-averaged cost > 1 (constraint violated). **Bold**: row maximum in # Best among safe. Per-task results are reported in Table H.

- **BC**: Standard BC minimizing $\mathbb{E}_{(s,a)\sim d_\mathrm{P}}[-\log \pi(a|s)]$.
- **Filtered BC (FBC)**: BC restricted to samples retained by the estimated support $\hat{\mathbb{1}}_{d_\mathrm{S}}$.
- **Jointly Filtered BC (JFBC)**: BC restricted to samples accepted by both $\hat{\mathbb{1}}_{d_\mathrm{S}}$ and $\hat{\mathbb{1}}_{d_\mathrm{P}}$.
- **DemoDICE-S/-U**: Two variants of DemoDICE [31] that minimize $D_\mathrm{KL}(d\|d_\mathrm{P}) + \alpha D_\mathrm{KL}(d\|d_\mathrm{S})$, parametrized via $w = d/d_\mathrm{S}$ ($\mathcal{D}_\mathrm{S}$ substituted for $\mathcal{D}_\mathrm{U}$) and $w = d/d_\mathrm{U}$ (using $\mathcal{D}_\mathrm{U}$), respectively; both serve as ablations of DexDICE that replace its support constraint with distribution matching to d_S.
- **DWBC-S**: DWBC [64] with $\mathcal{D}_\mathrm{S}$ substituted for $\mathcal{D}_\mathrm{U}$: expert-like samples are filtered from $\mathcal{D}_\mathrm{S}$ via a PU-learning-based discriminator f that approximates $\mathcal{D}_\mathrm{P}$, and a policy is then trained by discriminator-weighted BC, minimizing $\mathbb{E}_{(s,a)\sim d_\mathrm{S}}[-f(s,a)\log \pi(a \mid s)]$ (Appendix B.1).
- **FISOR-IL** [70], **CDT** [43], and **TraC** [18]: Offline safe IL/RL methods fed with binary pseudo-labels $(\hat{r}, \hat{c}) \in \{0,1\}^2$ in lieu of ground-truth reward and cost, ensuring matched information access (Appendix B.2).

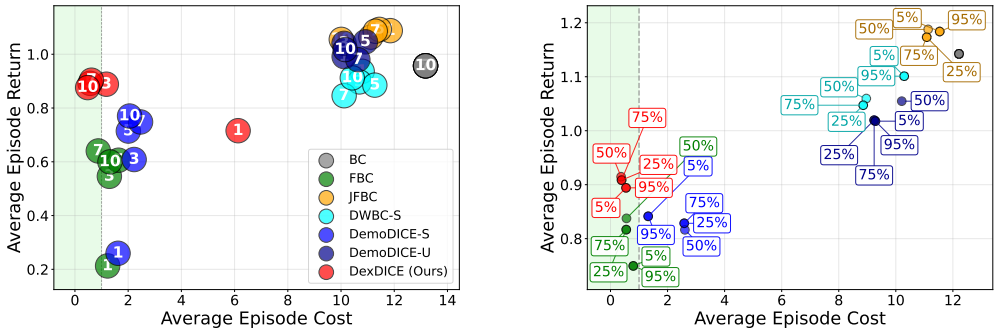
Information access is matched. BC, FBC, JFBC, DemoDICE-S/-U, and DWBC-S share DexDICE's $(\mathcal{D}_\mathrm{S}, \mathcal{D}_\mathrm{P}, \mathcal{D}_\mathrm{U})$ partition; FISOR-IL, CDT, and TraC receive binary pseudo-labels $(\hat{r}, \hat{c}) \in \{0,1\}^2$ obtained by thresholding two discriminators trained on $\mathcal{D}_\mathrm{P}$ vs. $\mathcal{D}_\mathrm{U}$ and $\mathcal{D}_\mathrm{S}$ vs. $\mathcal{D}_\mathrm{U}$, instead of ground-truth reward and cost labels.

For each task, the episode return is used as the performance metric and normalized to $[0, 1]$ using the minimum and maximum returns, and the episode cost is measured as the safety metric and normalized to $[0, 1]$ by the cost upper bound δ_c (a method is deemed safe when its mean normalized cost is ≤ 1). Each result averages 50 rollouts at the last checkpoint, across 5 random seeds.

5.1 Effectiveness on Re-curated DSRL Benchmark

To address **Q1**, we evaluate all methods on the 38 tasks of the re-curated DSRL benchmark, grouped into Safety Gymnasium (21 tasks), Bullet Safety Gym (8 tasks), and MetaDrive (9 tasks). To construct $\mathcal{D}_\mathrm{P}$ and $\mathcal{D}_\mathrm{S}$ appropriate for the dual-expert IL problem, we label each trajectory as performant or non-performant, and as safe or unsafe, according to predefined return and cost thresholds (see Appendix G.1; sensitivity analysis to these thresholds in Appendix G.10). Note that these are used solely for dataset construction; DexDICE does not require reward or cost during training. We use 10 randomly sampled performant trajectories to form $\mathcal{D}_\mathrm{P}$, and another 10 safe trajectories to form $\mathcal{D}_\mathrm{S}$, while setting $\mathcal{D}_\mathrm{U}$ to the entire offline dataset.

Table 1 reports group-wise aggregates (per-task results in Table H). DexDICE keeps the group-averaged cost below the constraint and attains the highest mean episode return among safe methods on all three benchmarks; per task, it satisfies the cost constraint on 31 of 38 tasks and is the best safe method on 19. BC, JFBC, DWBC-S, FISOR-IL, and DemoDICE-U frequently violate safety;



(a) Data efficiency on $N_{\mathcal{D}_\mathrm{S}}$

(b) Dependency on **quality**$(\mathcal{D}_S)$

(c) Data efficiency on $N_{\mathcal{D}_\mathrm{P}}$

(d) Dependency on **quality**$(\mathcal{D}_P)$

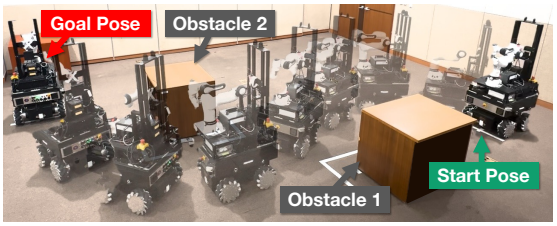
Figure 2: Data efficiency (a) and sample quality (b) on the safe demonstration set $\mathcal{D}_\mathrm{S}$ across methods, mirroring counterpart plots on the performant demonstration set $\mathcal{D}_\mathrm{P}$ (c), (d) respectively. Results are averaged over 6 tasks. Each subplot examines: (a) varying $N_{\mathcal{D}_\mathrm{S}} = 2k$, $k \in \{1, 3, 5, 7, 10\}$, with $N_{\mathcal{D}_\mathrm{P}} = 10$ fixed; (b) varying **quality**$(\mathcal{D}_\mathrm{S}) \in \{5\%, 25\%, 50\%, 75\%, 95\%\}$ with $N_{\mathcal{D}_\mathrm{P}} = N_{\mathcal{D}_\mathrm{S}} = 20$ and **quality**$(\mathcal{D}_\mathrm{P}) = 50\%$ fixed; (c), (d) the counterparts on $\mathcal{D}_\mathrm{P}$ with the roles of $\mathcal{D}_\mathrm{S}$ and $\mathcal{D}_\mathrm{P}$ exchanged. The green shaded region denotes the cost constraint-satisfying area.

CDT and TraC become safer under pseudo-labels but at substantial return cost. DexDICE's margin over DemoDICE-S/-U confirms that distribution matching biases the policy toward suboptimal safe behaviors, whereas the support constraint preserves high performance within safe regions (further comparisons against SafeDICE, COptiDICE, ICSDICE in Appendix G.3).

5.2 Data Efficiency and Quality Robustness

To address **Q2** (data efficiency) and **Q3** (quality robustness), we construct *controlled datasets* on six Safety Gymnasium tasks spanning the Circle and Velocity families by mixing four policy types (`safe-high`, `safe-medium`, `unsafe-high`, `unsafe-medium`): $\mathcal{D}_\mathrm{S}$ combines `safe-high`/`safe-medium`, $\mathcal{D}_\mathrm{P}$ combines `safe-high`/`unsafe-high`, and $\mathcal{D}_\mathrm{U}$ contains all four (Appendix G.2). For **Q2**, we sweep $N_{\mathcal{D}_\mathrm{P}}$ or $N_{\mathcal{D}_\mathrm{S}} \in \{2, 6, 10, 14, 20\}$ with the other at 10; for **Q3**, we fix $N_{\mathcal{D}_\mathrm{P}}{=}N_{\mathcal{D}_\mathrm{S}}{=}20$ and sweep **quality**$(\mathcal{D}_\mathrm{P})$ or **quality**$(\mathcal{D}_\mathrm{S}) \in \{5\%, 25\%, 50\%, 75\%, 95\%\}$ with the other at 50%, where **quality**$(\mathcal{D})$ is the `safe-high` fraction. Each controlled dataset requires training four policies per task, limiting the ablation to this representative subset.

Figure 2-(a),(c) report six-task averages as $N_{\mathcal{D}_\mathrm{S}}$ and $N_{\mathcal{D}_\mathrm{P}}$ vary, respectively. DexDICE maintains safety and attains the highest episode return among safe methods, struggling only at $N_{\mathcal{D}_\mathrm{S}}{=}2$ where the safe support cannot be reliably estimated. Figure 2-(b),(d) show that as **quality**$(\mathcal{D}_\mathrm{S})$ or **quality**$(\mathcal{D}_\mathrm{P})$ degrades, baselines deteriorate sharply while DexDICE sustains both safety and high return, except when the overlap between d_P and d_S becomes extreme (e.g., **quality**$(\mathcal{D}_\mathrm{P}){\leq}5\%$). Joint $(\mathcal{D}_\mathrm{S}, \mathcal{D}_\mathrm{P})$ scaling is reported in Figure B. Per-task analyses are in Appendices G.5.2–G.6.2.



(a) Navigation arena with obstacles.

Method	Success Rate (%)	Time-to-goal(s)	Violation Rate (%)
BC-Safe	8/10 (80)	36.0 ± 6.4	3.81 ± 1.56
FBC	6/10 (60)	15.8 ± 0.7	6.38 ± 1.62
DemoDICE-U	8/10 (80)	15.9 ± 0.1	0.69 ± 0.20
FISOR-IL ($\hat{c}$)	8/10 (80)	16.9 ± 1.3	4.89 ± 1.53
DexDICE (ours)	**10/10 (100)**	16.9 ± 0.5	$\mathbf{0.00 \pm 0.00}$

(b) Per-method results (10 trials).

Figure 3: Real-world mobile-robot navigation. (a) Arena: start (green) to goal (red) avoiding two obstacles. (b) Per-method results over 10 trials. *Success*: reached goal without collision. *Time-to-goal*: on successful trials. *Violation rate*: fraction of timesteps within an unsafe distance to obstacles.

In addition, we observe that DexDICE is also robust to κ in a reasonable range (Appendix G.8) and to noisy safe demonstrations under moderate corruption (Appendix G.9).

5.3 Real-World Mobile Robot Navigation from Human Demonstrations

To address **Q4**, we next ask whether DexDICE can learn from *human-collected* demonstrations and survive deployment on real hardware beyond simulation. We use a SUMMIT-XL-Steel mobile robot with a 2D LiDAR for goal-reaching navigation with intermediate obstacles. A single operator records $N{=}114$ joystick teleoperation episodes: 10 *aggressive* runs (fast, possibly unsafe) form $\mathcal{D}_{\mathrm{P}}$, 10 *conservative* runs (safe, slow) form $\mathcal{D}_{\mathrm{S}}$, and the remaining 94 episodes together with $\mathcal{D}_{\mathrm{P}} \cup \mathcal{D}_{\mathrm{S}}$ form $\mathcal{D}_{\mathrm{U}}$. To smooth control outputs and prevent actuator stress, we apply action chunking [69] uniformly across methods. Hardware, MDP, training, and evaluation details are in Appendix H.

Despite hardware-induced uncertainty and only 20 directed human episodes, DexDICE is the only method that reaches the goal in every trial with *zero* safety violations (Figure 3(b)). BC-Safe, which uses safe demonstrations only, is safer than FBC/FISOR-IL but its conservatism more than doubles time-to-goal; FBC and FISOR-IL match expert speed but violate safety frequently; DemoDICE-U closely tracks DexDICE yet misses the goal in $2/10$ trials. These results show that the advantages observed on the simulated benchmarks carry over to a noisy, human-driven physical setting (per-trial analysis in Figure J in Appendix H).

6 Conclusion

We proposed DexDICE, a framework for learning safe and performant policies from dual-expert demonstrations in the offline regime, without relying on explicit reward or cost signals. By replacing distributional regularization toward d_{S} with a safe support constraint, DexDICE decouples safety from the safe expert's suboptimality: it attains the highest return among constraint-satisfying methods on all three families of the re-curated DSRL benchmark, remains data-efficient and robust to demonstration quality, and transfers to a real mobile robot trained from human teleoperation, reaching the goal in every trial without a safety violation. The support constraint further admits a high-probability zero-unsafe-visitation guarantee for the ideal support-constrained policy, calibrated by a single parameter.

Our approach has two main limitations. First, as in any constrained formulation, the dual-expert IL problem requires a feasibility condition: non-trivial overlap between safe and performant demonstrations and a separable safe support. Our controlled studies show graceful degradation over a wide range of overlap levels, with failures only under extreme scarcity. Second, the exact guarantee concerns the ideal policy yet the weighted BC policy in practice need not be hard-supported. This gap cannot be eliminated in general, but deployment-time projection substantially reduces it at a modest cost in return (Appendix G.11). The conditions of Theorem 3 are population-level and are monitored through empirical diagnostics rather than certified (Appendix G.7), and results depend on the dataset-construction thresholds, to which we report sensitivity (Appendix G.10). A promising direction for future work is to extend our framework to handle multiple safety constraints or multi-objective settings.

Acknowledgments and Disclosure of Funding

We thank the reviewers and the area chairs for their valuable feedback. We also acknowledge support from the Google.org GCP Program in the form of Google Cloud credits. S. Seo, M. Seo and Y. Lee were supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. RS-2025-25448259). J. Lee was supported by the Institute of Information & communications Technology Planning & Evaluation (IITP) grants funded by the Korea government (MSIT) (No. RS-2020-II201361, RS-2024-00457882), the National Research Foundation of Korea (NRF) grants funded by the Korea government (MSIT) (No. RS-2026-25505230, RS-2026-25619452), and a grant from the Institute for AI and Social Innovation at Yonsei University. H. Yang was supported by the National Research Foundation of Korea (NRF) grant funded by the Korean Government (MSIT) (No. RS-2023-00279680), and a KIAS Individual Grant (CG109601) at the Korea Institute for Advanced Study. S. Shyn and K. Kim were supported by the Institute of Information & communications Technology Planning & Evaluation (IITP) grants funded by the Korea government (MSIT) (No. RS-2024-00457882, RS-2022-II220311, RS-2020-II200940, RS-2019-II190075). The authors declare no competing financial interests.

References

[1] Eitan Altman. *Constrained Markov decision processes*. Chapman and Hall/CRC, 1999.

[2] Aaron D. Ames, Samuel Coogan, Magnus Egerstedt, Gennaro Notomista, Koushil Sreenath, and Paulo Tabuada. Control barrier functions: Theory and applications. In *18th European Control Conference (ECC)*, 2019.

[3] Mayank Bansal, Alex Krizhevsky, and Abhijit S. Ogale. ChauffeurNet: Learning to drive by imitating the best and synthesizing the worst. In *Robotics: Science and Systems XV*, 2019.

[4] Somil Bansal, Mo Chen, Sylvia Herbert, and Claire J. Tomlin. Hamilton–Jacobi reachability: A brief overview and recent advances. In *IEEE 56th Annual Conference on Decision and Control (CDC)*, 2017.

[5] Jessa Bekker and Jesse Davis. Learning from positive and unlabeled data: A survey. *Machine Learning*, 109, 2020.

[6] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[7] Stephen P Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[8] Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. LOF: Identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, 2000.

[9] Returaj Burnwal, Nirav Pravinbhai Bhatt, and Balaraman Ravindran. SafeMIL: Learning offline safe imitation policy from non-preferred trajectories. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, 2026.

[10] Eamon T Campolettano, Kristofer D Kusano, and Trent Victor. Potential safety benefits associated with speed limit compliance in San Francisco and Phoenix. *Traffic Injury Prevention*, 26, 2025.

[11] Shengfan Cao, Eunhyek Joa, and Francesco Borrelli. A simple approach to constraint-aware imitation learning with application to autonomous racing. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2025.

[12] Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Misha Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

[13] Marthinus Christoffel, Gang Niu, and Masashi Sugiyama. Class-prior estimation for learning from positive and unlabeled data. In *Proceedings of the 7th Asian Conference on Machine Learning*, volume 45, 2016.

[14] Yusuf Umut Ciftci, Darren Chiu, Zeyuan Feng, Gaurav S Sukhatme, and Somil Bansal. SAFE-GIL: Safety guided imitation learning for robotic systems. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2025.

[15] Robert Dadashi, Leonard Hussenot, Matthieu Geist, and Olivier Pietquin. Primal Wasserstein imitation learning. In *The Ninth International Conference on Learning Representations*, 2021.

[16] Charles Elkan and Keith Noto. Learning classifiers from only positive and unlabeled data. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2008.

[17] Bin Fang, Shidong Jia, Di Guo, Muhua Xu, Shuhuan Wen, and Fuchun Sun. Survey of imitation learning for robotic manipulation. *International Journal of Intelligent Robotics and Applications*, 3, 2019.

[18] Ze Gong, Akshat Kumar, and Pradeep Varakantham. Offline safe reinforcement learning using trajectory classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, 2025.

[19] Jeffrey Hawke, Richard Shen, Corina Gurau, Siddharth Sharma, Daniele Reda, Nikolay Nikolov, Przemysław Mazur, Sean Micklethwaite, Nicolas Griffiths, Amar Shah, et al. Urban driving with conditional imitation learning. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2020.

[20] Shohei Hido, Yuta Tsuboi, Hisashi Kashima, Masashi Sugiyama, and Takafumi Kanamori. Statistical outlier detection using direct density ratio estimation. *Knowledge and Information Systems*, 26, 2011.

[21] Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. In *Advances in Neural Information Processing Systems*, volume 29, 2016.

[22] Huy Hoang, Tien Mai, and Pradeep Varakantham. SPRINQL: Sub-optimal demonstrations driven offline imitation learning. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

[23] Huy Hoang, Tien Mai, and Pradeep Varakantham. No experts, no problem: Avoidance learning from bad demonstrations. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

[24] Adriana Hugessen, Harley Wiltzer, and Glen Berseth. Simplifying constraint inference with inverse reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

[25] Ahmed Hussein, Mohamed Medhat Gaber, Eyad Elyan, and Chrisina Jayne. Imitation learning: A survey of learning methods. *ACM Computing Surveys (CSUR)*, 50, 2017.

[26] Youngsoo Jang, Geon-Hyeong Kim, Jongmin Lee, Sungryull Sohn, Byoungjip Kim, Honglak Lee, and Moontae Lee. SafeDICE: Offline safe imitation learning with non-preferred demonstrations. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

[27] Daniel Jarrett, Ioana Bica, and Mihaela van der Schaar. Strictly batch imitation learning by energy-based distribution matching. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

[28] Jiaming Ji, Borong Zhang, Jiayi Zhou, Xuehai Pan, Weidong Huang, Ruiyang Sun, Yiran Geng, Yifan Zhong, Josef Dai, and Yaodong Yang. Safety Gymnasium: A unified safe reinforcement learning benchmark. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

[29] Jiaming Ji, Jiayi Zhou, Borong Zhang, Juntao Dai, Xuehai Pan, Ruiyang Sun, Weidong Huang, Yiran Geng, Mickel Liu, and Yaodong Yang. OmniSafe: An infrastructure for accelerating safe reinforcement learning research. *Journal of Machine Learning Research*, 25, 2024.

[30] Edward Johns. Coarse-to-fine imitation learning: Robot manipulation from a single demonstration. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

[31] Geon-Hyeong Kim, Seokin Seo, Jongmin Lee, Wonseok Jeon, HyeongJoo Hwang, Hongseok Yang, and Kee-Eung Kim. DemoDICE: Offline imitation learning with supplementary imperfect demonstrations. In *The Tenth International Conference on Learning Representations*, 2022.

[32] Kee-Eung Kim and Hyun Soo Park. Imitation learning via kernel mean embedding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[33] Konwoo Kim, Gokul Swamy, Zuxin Liu, Ding Zhao, Sanjiban Choudhury, and Steven Z Wu. Learning shared safety constraints from multi-task demonstrations. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

[34] Matthieu Komorowski, Leo A Celi, Omar Badawi, Anthony C Gordon, and A Aldo Faisal. The artificial intelligence clinician learns optimal treatment strategies for sepsis in intensive care. *Nature Medicine*, 24, 2018.

[35] Hyeokjin Kwon, Gunmin Lee, Junseo Lee, and Songhwai Oh. Safe CoR: A dual-expert approach to integrating imitation learning and safe reinforcement learning using constraint rewards. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2024.

[36] Przemyslaw A Lasota, Gregory F Rossano, and Julie A Shah. Toward safe close-proximity human-robot interaction with standard industrial robots. In *IEEE International Conference on Automation Science and Engineering (CASE)*, 2014.

[37] Luc Le Mero, Dewei Yi, Mehrdad Dianati, and Alexandros Mouzakitis. A survey on imitation learning techniques for end-to-end autonomous vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 23, 2022.

[38] Jongmin Lee, Wonseok Jeon, Byungjun Lee, Joelle Pineau, and Kee-Eung Kim. OptiDICE: Offline policy optimization via stationary distribution correction estimation. In *Proceedings of the 38th International Conference on Machine Learning*, 2021.

[39] Jongmin Lee, Cosmin Paduraru, Daniel J Mankowitz, Nicolas Heess, Doina Precup, Kee-Eung Kim, and Arthur Guez. COptiDICE: Offline constrained reinforcement learning via stationary distribution correction estimation. In *The Tenth International Conference on Learning Representations*, 2022.

[40] Guiliang Liu, Sheng Xu, Shicheng Liu, Ashish Gaurav, Sriram Ganapathi Subramanian, and Pascal Poupart. A comprehensive survey on inverse constrained reinforcement learning: Definitions, progress and challenges. *Transactions on Machine Learning Research*, 2025.

[41] Qiang Liu, Huiqiao Fu, Kaiqiang Tang, Chunlin Chen, and Daoyi Dong. PN-GAIL: Leveraging non-optimal information from imperfect demonstrations. In *The Thirteenth International Conference on Learning Representations*, 2025.

[42] Zifan Liu, Xinran Li, and Jun Zhang. C2IQL: Constraint-conditioned implicit Q-learning for safe offline reinforcement learning. In *Proceedings of the 42nd International Conference on Machine Learning*, 2025.

[43] Zuxin Liu, Zijian Guo, Yihang Yao, Zhepeng Cen, Wenhao Yu, Tingnan Zhang, and Ding Zhao. Constrained decision transformer for offline safe reinforcement learning. In *Proceedings of the 40th International Conference on Machine Learning*, 2023.

[44] Zuxin Liu, Zijian Guo, Haohong Lin, Yihang Yao, Jiacheng Zhu, Zhepeng Cen, Hanjiang Hu, Wenhao Yu, Tingnan Zhang, Jie Tan, et al. Datasets and benchmarks for offline safe reinforcement learning. *Journal of Data-centric Machine Learning Research*, 2024.

[45] Yecheng Ma, Andrew Shen, Dinesh Jayaraman, and Osbert Bastani. Versatile offline imitation from observations and examples via regularized state-occupancy matching. In *Proceedings of the 39th International Conference on Machine Learning*, 2022.

[46] Shehryar Malik, Usman Anwar, Alireza Aghasi, and Ali Ahmed. Inverse constrained reinforcement learning. In *Proceedings of the 38th International Conference on Machine Learning*, 2021.

[47] Ajay Mandlekar, Danfei Xu, Josiah Wong, Soroush Nasiriany, Chen Wang, Rohun Kulkarni, Li Fei-Fei, Silvio Savarese, Yuke Zhu, and Roberto Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation. In *Proceedings of the 5th Conference on Robot Learning*, volume 164, 2021.

[48] Ofir Nachum, Yinlam Chow, Bo Dai, and Lihong Li. DualDICE: Behavior-agnostic estimation of discounted stationary distribution corrections. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

[49] Shamim Nemati, Mohammad M Ghassemi, and Gari D Clifford. Optimal medication dosing from suboptimal clinical examples: A deep reinforcement learning approach. In *38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 2016.

[50] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. PointNet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[51] Guorui Quan, Zhiqiang Xu, and Guiliang Liu. Learning constraints from offline demonstrations via superior distribution correction estimation. In *Proceedings of the 41st International Conference on Machine Learning*, 2024.

[52] Alex Ray, Joshua Achiam, and Dario Amodei. Benchmarking safe exploration in deep reinforcement learning. *arXiv preprint arXiv:1910.01708*, 2019.

[53] Stephane Ross and Drew Bagnell. Efficient reductions for imitation learning. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics*, 2010.

[54] Lukas Ruff, Robert Vandermeulen, Nico Goernitz, Lucas Deecke, Shoaib Ahmed Siddiqui, Alexander Binder, Emmanuel Müller, and Marius Kloft. Deep one-class classification. In *Proceedings of the 35th International Conference on Machine Learning*, 2018.

[55] Bernhard Schölkopf, John C Platt, John Shawe-Taylor, Alex J Smola, and Robert C Williamson. Estimating the support of a high-dimensional distribution. *Neural Computation*, 13, 2001.

[56] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[57] Clayton Scott and Gilles Blanchard. Novelty detection: Unlabeled data definitely help. In *Proceedings of the 12th International Conference on Artificial Intelligence and Statistics*, 2009.

[58] Seokin Seo, Byung-Jun Lee, Jongmin Lee, HyeongJoo Hwang, Hongseok Yang, and Kee-Eung Kim. Mitigating covariate shift in behavioral cloning via robust stationary distribution correction. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

[59] Qian Shao, Pradeep Varakantham, and Shih-Fen Cheng. Imitating cost-constrained behaviors in reinforcement learning. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 34, 2024.

[60] Sriram Ganapathi Subramanian, Guiliang Liu, Mohammed Elmahgiubi, Kasra Rezaee, and Pascal Poupart. Confidence aware inverse constrained reinforcement learning. In *Proceedings of the 41st International Conference on Machine Learning*, 2024.

[61] Lei Tai, Giuseppe Paolo, and Ming Liu. Virtual-to-real deep reinforcement learning: Continuous control of mobile robots for mapless navigation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017.

[62] Xin Tong. A plug-in approach to Neyman-Pearson classification. *Journal of Machine Learning Research*, 14, 2013.

[63] Fan Xie, Alexander Chowdhury, M De Paolis Kaluza, Linfeng Zhao, Lawson Wong, and Rose Yu. Deep imitation learning for bimanual robotic manipulation. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

[64] Haoran Xu, Xianyuan Zhan, Honglei Yin, and Huiling Qin. Discriminator-weighted offline imitation learning from suboptimal demonstrations. In *Proceedings of the 39th International Conference on Machine Learning*, 2022.

[65] He Yin, Peter Seiler, Ming Jin, and Murat Arcak. Imitation learning with stability and safety guarantees. *IEEE Control Systems Letters*, 6, 2022.

[66] Lantao Yu, Tianhe Yu, Jiaming Song, Willie Neiswanger, and Stefano Ermon. Offline imitation learning with suboptimal demonstrations via relaxed distribution matching. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, 2023.

[67] Maryam Zare, Parham M Kebria, Abbas Khosravi, and Saeid Nahavandi. A survey of imitation learning: Algorithms, recent developments, and challenges. *IEEE Transactions on Cybernetics*, 54, 2024.

[68] Xin Zhang, Yanhua Li, Ziming Zhang, and Zhi-Li Zhang. f-GAIL: Learning f-divergence for generative adversarial imitation learning. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

[69] Tony Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *Robotics: Science and Systems XIX*, 2023.

[70] Yinan Zheng, Jianxiong Li, Dongjie Yu, Yujie Yang, Shengbo Eben Li, Xianyuan Zhan, and Jingjing Liu. Safe offline reinforcement learning with feasibility-guided diffusion model. In *The Twelfth International Conference on Learning Representations*, 2024.

Appendix

Contents

A Problem Definition **18**

B Detailed Discussions on Related Work **20**

B.1 Conceptual Comparison with DICE Methods 20

B.2 Detailed Discussion of Offline Safe RL Baselines 21

C Tabular Experiment: Distribution vs. Support Constraint **22**

C.1 Environment Setup 22

C.2 Results 22

D Proofs of Propositions **23**

D.1 Derivation of the Lagrangian Objective $L(w, \nu, \lambda)$ 23

D.2 Strong Duality and Closed-Form Solution 23

E Safety Guarantee of DexDICE **24**

F Details on Practical Algorithm **26**

F.1 Discriminator-based Estimation 26

F.2 Numerical Stability 26

F.3 Policy Extraction 27

F.4 Algorithm Summary 27

G Details on Simulation Experiments **27**

G.1 Re-curated DSRL Datasets 27

G.2 Controlled Datasets 32

G.3 Empirical Comparison with Offline Safe Methods in Different Settings 33

G.4 Hyperparameters 36

G.5 Data Efficiency Analysis 36

G.5.1 Data Efficiency with Joint Scaling 36

G.5.2 Data Efficiency on $\mathcal{D}_{\mathrm{S}}$ 38

G.5.3 Data Efficiency on $\mathcal{D}_{\mathrm{P}}$ 39

G.6 Data Quality Analysis 40

G.6.1 Data Quality on $\mathcal{D}_{\mathrm{S}}$ 40

G.6.2 Data Quality on $\mathcal{D}_{\mathrm{P}}$ 41

G.7 Comparison on Support Estimators 42

G.8 Sensitivity Analysis on Target FNR κ 43

G.9 Robustness on Noisy Safe Demonstrations 43
G.10 Sensitivity to Dataset Construction Thresholds 43
G.11 Deployment-Time Projection onto the Estimated Safe Support 44

H Details of Real-World Robot Experiment **45**
H.1 Hardware Specification 45
H.2 MDP Specification 45
H.3 Training Setup 46
H.4 Evaluation Protocol 46
H.5 Per-Trial Visualization 46

I Compute Resources **48**

J Broader Impacts **48**

A Problem Definition

For any function $f : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, we denote the expected value under the stationary distribution as:

$$J(\pi; f) := \mathbb{E}_{(s,a)\sim d_\pi}[f(s,a)].$$

Given a cost upper bound $\delta_c \in \mathbb{R}_{\geq 0}$ and a performance lower bound $\delta_r \in \mathbb{R}$, we define Π_{S} and Π_{P} to be the following sets of policies:

$$\Pi_{\mathrm{S},\delta_c} := \{\pi_{\mathrm{S}} \in \Pi \mid J(\pi_{\mathrm{S}}; c) \leq \delta_c\} \quad \text{and} \quad \Pi_{\mathrm{P},\delta_r} := \{\pi_{\mathrm{P}} \in \Pi \mid J(\pi_{\mathrm{P}}; r) \geq \delta_r\}.$$

Intuitively, Π_{S} is the set of safe policies, and Π_{P} that of performant policies. Also, for a distribution d on state-action pairs, we write $\mathrm{supp}(d)$ for the support of d, i.e., $\mathrm{supp}(d) = \{(s,a) \mid d(s,a) > 0\}$.

Definition A (Dual-Expert Imitation Learning)**.** Let $\delta_c \in \mathbb{R}_{\geq 0}$ and $\delta_r \in \mathbb{R}$ denote a cost upper bound and a performance lower bound, respectively. Let $\pi_{\mathrm{S}} \in \Pi_{\mathrm{S},\delta_c}$ and $\pi_{\mathrm{P}} \in \Pi_{\mathrm{P},\delta_r}$ denote a safe policy and a performant policy, with stationary distributions d_{S} and d_{P}. We assume the feasibility condition

$$\overline{\Pi} := \{\pi \in \Pi \mid \mathrm{supp}(d_\pi) \subseteq \mathrm{supp}(d_{\mathrm{S}}) \cap \mathrm{supp}(d_{\mathrm{P}})\} \neq \emptyset, \tag{10}$$

where d_π is the stationary distribution of policy π. Given finite demonstration datasets

$$\mathcal{D}_{\mathrm{S}} = \{\tau_{\mathrm{S}}^{(1)}, \ldots, \tau_{\mathrm{S}}^{(m)}\}, \quad \tau_{\mathrm{S}}^{(i)} \sim \pi_{\mathrm{S}}, \qquad \mathcal{D}_{\mathrm{P}} = \{\tau_{\mathrm{P}}^{(1)}, \ldots, \tau_{\mathrm{P}}^{(n)}\}, \quad \tau_{\mathrm{P}}^{(i)} \sim \pi_{\mathrm{P}},$$

where each trajectory $\tau = \{s_0, a_0, s_1, a_1, \ldots\}$ is generated by $s_0 \sim \rho_0$, $a_t \sim \pi(s_t)$, and $s_{t+1} \sim T(s_t, a_t)$, the *dual-expert IL problem* seeks a policy π that minimizes

$$\min_{\pi \in \Pi} \mathbb{D}(d_\pi \| d_{\mathrm{P}}) \quad \text{subject to} \quad \mathrm{supp}(d_\pi) \subseteq \mathrm{supp}(d_{\mathrm{S}}). \tag{11}$$

The non-emptiness of $\overline{\Pi}$ ensures that some π satisfies the support constraint in the problem and also has a finite value when $\mathbb{D}$ is the KL divergence; these imply the feasibility of the above optimization problem in this case. If interaction with the environment is allowed during optimization, the problem is referred to as the online setting; otherwise, it is the offline setting.

Note that dual-expert IL is closely related to cost-constrained IL [59]. In particular, a solution to dual-expert IL corresponds to a feasible solution of the cost-constrained IL problem with a hard constraint, where the cost function is non-negative ($c \geq 0$) and the upper bound is fixed to $\delta_c = 0$:

$$\min_{\pi} \quad \mathbb{D}(d_\pi \| d_{\mathrm{P}}) \quad \text{subject to} \quad J(\pi; c) = 0.$$

This is because the safety condition $J(\pi; c) = 0$ follows from our support constraint $\mathrm{supp}(d_\pi) \subseteq \mathrm{supp}(d_{\mathrm{S}})$. To see why, note that $J(\pi_{\mathrm{S}}; c) \leq \delta_c = 0$. Together with the non-negativity of c, this implies that $c(s,a) = 0$ for all $(s,a) \in \mathrm{supp}(d_{\mathrm{S}})$. Thus, also by our support constraint, we have $c(s,a) = 0$ for all $(s,a) \in \mathrm{supp}(d_\pi)$, which gives $J(\pi; c) = \mathbb{E}_{d_\pi}[c(s,a)] = 0$, that is, π satisfies the hard constraint. The intermediate statement $c(s,a) = 0$ on $\mathrm{supp}(d_{\mathrm{S}})$ is precisely the pointwise soundness condition $\mathrm{supp}(d_{\mathrm{S}}) \cap \mathcal{N}_{\mathrm{env}} = \emptyset$ with $\mathcal{N}_{\mathrm{env}} := \{(s,a) : c(s,a) > 0\}$, under which the safety analysis of Appendix E is stated.

In practice, demonstrations may be collected from multiple safe policies and multiple performant policies, instead of just one π_{S} and one π_{P}. This general case is also covered by our setup. A finite set of demonstrations from multiple policies can be viewed as a demonstration set of a single policy obtained by a convex combination of the original multiple policies. Furthermore, if all those original policies are safe (i.e., they are all in $\Pi_{\mathrm{S},\delta_c}$), so is their convex combination. Similarly, if the policies are performant (i.e., they belong to $\Pi_{\mathrm{P},\delta_r}$), so is their convex combination. Thus, the general case can be understood as an instance of our problem where π_{S} and π_{P} are appropriate convex combinations of multiple safe and performant policies, respectively.

Dual-Expert IL with a trivial safe-support When d_{S} has full support, i.e., $d_{\mathrm{S}}(s,a) > 0$ for all $(s,a) \in \mathcal{S} \times \mathcal{A}$, the support constraint in Eq. 11 becomes vacuous and dual-expert IL collapses to unconstrained stationary-distribution matching, $\min_{d \in \mathcal{Q}} \mathbb{D}(d \,\|\, d_{\mathrm{P}})$, i.e., DemoDICE without behavior regularization. Analogously to constrained RL with a trivial cost constraint, this reflects that the safe demonstrations carry no safety signal, which underscores the practical importance of collecting informative safe demonstrations.

Dual-Expert IL with no support overlap If the safe and performant stationary distributions, d_{S} and d_{P}, have

$$\mathrm{supp}(d_{\mathrm{S}}) \cap \mathrm{supp}(d_{\mathrm{P}}) = \emptyset,$$

then the feasible region of dual-expert IL becomes severely restricted. Under the dual-expert IL formulation

$$\min_{d \in \mathcal{Q}} \mathbb{D}(d \,\|\, d_{\mathrm{P}}) \quad \text{s.t.} \quad \mathrm{supp}(d) \subseteq \mathrm{supp}(d_{\mathrm{S}}),$$

any feasible d must satisfy $d(s,a) = 0$ outside $\mathrm{supp}(d_{\mathrm{S}})$, while d_{P} assigns mass exclusively outside this region. Hence no feasible d is absolutely continuous with respect to d_{P}, and the feasibility condition (10) is violated. Under the KL divergence used throughout this paper, this means $D_{\mathrm{KL}}(d \,\|\, d_{\mathrm{P}}) = +\infty$ for every feasible d, so the optimization is degenerate and admits no finite minimizer in this regime. For broader choices of $\mathbb{D}$ that remain finite without absolute continuity (e.g., total variation or Wasserstein), the problem is well-posed but reduces to selecting the feasible d that minimizes $\mathbb{D}(d \,\|\, d_{\mathrm{P}})$ within $\mathrm{supp}(d_{\mathrm{S}})$, a degenerate solution whose support is disjoint from d_{P} and therefore cannot meaningfully imitate it. In either case, dual-expert IL collapses to imitating only the safe demonstrations, even when d_{P} lies entirely outside the safe support. This limitation is also inherent to constrained optimization and is directly analogous to constrained RL: when the constraint is overly strict, no policy that attains sufficiently high return can satisfy the constrained RL objective.

Practical interpretation of the overlap condition The feasibility condition does not require a dataset of entire trajectories that are jointly safe and performant; it requires the separately collected sources to share enough state-action support to admit a Bellman-flow-feasible policy, which can hold even when the two demonstrators differ substantially at the trajectory level. Representative cases include: (i) autonomous driving, where human driver logs contain both fast, risky maneuvers and rule-compliant driving [3, 10], and the overlap consists of efficient routes that still respect the limits; (ii) human-robot collaboration, where a manipulator moves fast when the workspace is clear but slows near a person [36], so that efficient-and-safe motions populate the overlap; and (iii) changing safety regulations, where a revised rule (e.g., updated speed limits, no-go zones, or clearance standards) redefines the safe region, so that only the safe demonstrations need re-collecting or re-labeling while the performant demonstrations remain reusable, and the overlap is the efficient behavior that still complies. The last case is precisely where labeling demonstrations is cheaper than re-engineering a reward or cost function. Our real-robot navigation experiment (Section 5.3) is a further instance: safe demonstrations keep clearance from obstacles, performant ones take faster routes, and the overlap is the set of fast routes that pass through safe corridors.

B Detailed Discussions on Related Work

DICE Method	Target Offline Problem	Reward	Cost	$\mathcal{D}_{\mathrm{P}}$	$\mathcal{D}_{\mathrm{S}}$	$\mathcal{D}_{\mathrm{N}}$	Safety	Constraints
OptiDICE [38]	RL	✓	–	–	–	–	–	Distribution
DemoDICE [31], SMODICE [45], RelaxDICE [66]	IL	–	–	✓	–	–	–	Distribution
SafeDICE [26]	IL	–	–	–	–	✓	✓	Distribution
COptiDICE [39]	Constrained RL	✓	✓	–	–	–	✓	Distribution, Cost (true)
ICSDICE [51]	Inverse Constrained RL	✓	–	–	✓	–	✓	Distribution, Cost (inferred)
DexDICE (Ours)	Dual-expert IL	–	–	✓	✓	–	✓	Support

Table A: Structural comparison of DICE methods for offline problems in terms of required learning signals, safety constraints, and dataset requirements. $\mathcal{D}_{\mathrm{N}}$ indicates a set of non-preferred demonstrations, which corresponds to unsafe demonstrations in our setting.

In this section, we provide a detailed discussion of DICE-based approaches developed for offline learning problems that are closely related to our setting. Table A summarizes representative methods in terms of required learning signals, safety considerations, and dataset requirements. DICE methods have been widely studied in both RL and IL because they bypass explicit sequential modeling of MDP dynamics by optimizing over stationary distributions constrained to satisfy the Bellman flow condition. This perspective leads to linear programming objectives with tractable convex dual forms, making DICE formulations particularly appealing for offline settings.

B.1 Conceptual Comparison with DICE Methods

A representative DICE method for offline RL is OptiDICE [38], whose objective is

$$\max_{d \in \mathcal{Q}} \ \mathbb{E}_d[r(s,a)] \ - \ \alpha \, D_f(d \,\|\, d_{\mathrm{U}}),$$

where $\mathcal{Q}$ denotes the set of stationary distributions. The f-divergence regularizer plays a central role by enforcing conservatism, preventing the learned distribution from drifting outside the support of the offline dataset, which is a standard requirement for stable offline RL. COptiDICE [39] extends OptiDICE to the CMDP setting. DemoDICE [31] adapts this idea to offline IL using the objective

$$\max_{d \in \mathcal{Q}} \ - \, D_{\mathrm{KL}}(d \,\|\, d_{\mathrm{P}}) \ - \ \alpha \, D_{\mathrm{KL}}(d \,\|\, d_{\mathrm{U}}),$$

which encourages the learned stationary distribution to remain close to both the performant demonstrations d_{P} and the offline dataset d_{U}. As shown in Table A, DemoDICE also relies on distributional proximity to d_{U}, which may be problematic in dual-expert IL; if the offline dataset contains suboptimal or unsafe behavior, forcing closeness to d_{U} can degrade either safety or performance. SafeDICE [26] leverages non-preferred demonstrations $\mathcal{D}_{\mathrm{N}}$ and assumes that $\mathcal{D}_{\mathrm{U}}$ is a mixture of expert and non-preferred data. ICSDICE [51] assumes that d_{S} is induced from an optimal CMDP policy and deems (s,a) risky if it is more likely under a superior stationary distribution, namely, the stationary distribution of a policy that outperforms d_{S}. As discussed in Section 2, replacing d_{U} with d_{S} (safe demonstrations) in DemoDICE can partially mitigate safety concerns, but still requires staying close to possibly low-quality safe demonstrations, making the method sensitive to data quality.

To address this limitation, DexDICE explicitly replaces the distributional regularizer with a support constraint:

$$\max_{d \in \mathcal{Q}} \ -D_{\mathrm{KL}}(d \,\|\, d_{\mathrm{P}}) \quad \text{subject to} \quad \mathrm{supp}(d) \subseteq \mathrm{supp}(d_{\mathrm{S}}).$$

This formulation seeks the stationary distribution that is closest to d_{P} while remaining entirely within the safe support defined by d_{S}. Among the DICE methods in Table A, DexDICE alone enforces safety through an explicit support constraint without requiring reward signals, cost signals, or non-preferred demonstrations. In the tabular comparison of Appendix C, this avoids the return degradation caused by distributional penalties toward suboptimal d_{S}.

Although it is not a DICE-based method, DWBC [64] is closely related to our setting as an offline IL approach that leverages unlabeled demonstrations. In its original formulation, DWBC filters expert-like samples from unlabeled data by optimizing the discriminator f by using the following PU-learning objective:

$$\min_f \ \mathbb{E}_{(s,a)\sim d_{\mathrm{P}}}[-\log f(s,a)] + \mathbb{E}_{(s,a)\sim d_{\mathrm{N}}}[-\log(1-f(s,a))]$$
$$= \min_f \ \eta\,\mathbb{E}_{(s,a)\sim d_{\mathrm{P}}}[-\log f(s,a)] + \mathbb{E}_{(s,a)\sim d_{\mathrm{U}}}[-\log(1-f(s,a))] - \eta\,\mathbb{E}_{(s,a)\sim d_{\mathrm{P}}}[-\log(1-f(s,a))],$$

where η denotes the class prior. The policy is then obtained via discriminator-weighted behavior cloning, i.e., $\min_\pi \mathbb{E}_{(s,a)\sim\mathcal{D}_{\mathrm{U}}}[-f(s,a)\log\pi(a|s)]$. In the context of safe IL, we compare against DWBC-S, a variant of DWBC that substitutes $\mathcal{D}_{\mathrm{S}}$ for $\mathcal{D}_{\mathrm{U}}$ throughout, so that expert-like samples are filtered from $\mathcal{D}_{\mathrm{S}}$ rather than from $\mathcal{D}_{\mathrm{U}}$ and the policy is trained by minimizing $\mathbb{E}_{(s,a)\sim\mathcal{D}_{\mathrm{S}}}[-f(s,a)\log\pi(a|s)]$.

B.2 Detailed Discussion of Offline Safe RL Baselines

This section elaborates on the offline safe RL methods discussed in Section 2, why they are not directly applicable to the dual-expert IL setting, and how we adapt them for the comparison reported in Section 5.1.

Offline safe RL methods. These methods take a constrained MDP as input, with explicit per-transition reward $r(s,a)$ and cost $c(s,a)$ provided in the offline dataset, and learn a policy that maximizes return subject to a cost budget.

- **COptiDICE** [39] extends OptiDICE [38] to constrained MDPs by introducing a Lagrangian term for the cost expectation.
- **CDT** [43] conditions a Decision Transformer [12] on both return-to-go and cost-to-go targets computed from each trajectory's reward and cost signals.
- **FISOR** [70] casts safety as a reachability problem, learning a cost value function V_c from $c(s,a)$ and using it to mask infeasible actions for a guided diffusion policy.
- **C2IQL** [42] extends IQL with a Lagrangian dual: separate Q-functions for reward and cost are learned via expectile regression and balanced by a learnable multiplier.
- **TraC** [18] reframes offline safe RL as trajectory classification: trajectories are scored by predicted return and total cost, then weighted behavioral cloning recovers the policy.

All five methods consume per-transition (r,c) supervision, which our dual-expert IL setting does not provide.

Why naive labeling fails. A natural attempt to apply these methods to our dual-expert IL setting is to construct labels from the dataset identity alone, e.g.,

$$\hat{r}(s,a) = \mathbb{1}[(s,a)\in\mathcal{D}_{\mathrm{P}}], \qquad \hat{c}(s,a) = \mathbb{1}[(s,a)\notin\mathcal{D}_{\mathrm{S}}].$$

This labeling is biased in two ways. First, $\mathbb{1}[(s,a)\notin\mathcal{D}_{\mathrm{S}}]$ marks every transition outside $\mathcal{D}_{\mathrm{S}}$ as unsafe, including a substantial number of safe transitions in $\mathcal{D}_{\mathrm{U}}$, since $\mathcal{D}_{\mathrm{U}}$ is a latent mixture of safe and unsafe behavior. Second, $\mathcal{D}_{\mathrm{S}}$ contains only positive (safe) examples while $\mathcal{D}_{\mathrm{U}}$ is unlabeled rather than negative; this is exactly the positive–unlabeled (PU) learning regime [5, 16], in which naive binary classification systematically over-estimates the negative class. Cost Q-functions trained on such biased labels inflate the unsafe region, leading either to overly conservative behavior or to unreliable feasibility masks that admit unsafe actions.

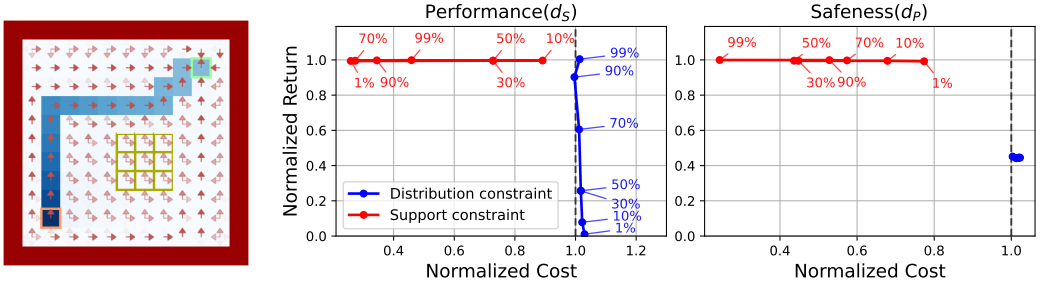
Pseudo-label adaptation. For empirical comparison under matched information access, we threshold DexDICE's learned discriminators $\hat{r}_{\mathrm{P}}$ and $\hat{r}_{\mathrm{S}}$ to obtain binary reward and cost pseudo-labels. Let τ_{P} and τ_{S} denote the corresponding discriminator-score thresholds:

$$\hat{r}(s,a) := \mathbb{1}\{\hat{r}_{\mathrm{P}}(s,a)\geq\tau_{\mathrm{P}}\}, \qquad \hat{c}(s,a) := \mathbb{1}\{\hat{r}_{\mathrm{S}}(s,a)<\tau_{\mathrm{S}}\}.$$

These labels assign $(\hat{r},\hat{c})\in\{0,1\}^2$ to transitions in $\mathcal{D}_{\mathrm{U}}$ using only $(\mathcal{D}_{\mathrm{P}},\mathcal{D}_{\mathrm{S}},\mathcal{D}_{\mathrm{U}})$, without ground-truth reward or cost signals. Section 5.1 reports the resulting performance for FISOR, CDT, and TraC under this pseudo-label scheme, and Appendix G.3 under the oracle (r,c) scheme; DexDICE outperforms the pseudo-label variants on the majority of tasks and remains competitive with the oracle variants despite using strictly less information.

C Tabular Experiment: Distribution vs. Support Constraint

This section provides a controlled comparison between distribution constraints and support constraints using a tabular domain where exact solutions can be computed. The key question is: *how do these two constraint types behave when the quality of d_{S} or d_{P} varies?*



(a) Constrained gridworld.

(b) Return vs. cost under varying d_{S} and d_{P} quality.

Figure A: An illustration on tabular experiment. (a) The agent must reach the goal while avoiding unsafe zones. (b) Each point shows the mean return and cost (averaged over 50 seeds) for a given demonstration quality (indicated by labels). The support constraint remains in the high return, low cost region across all settings, while the distribution constraint degrades when d_{S} is suboptimal.

C.1 Environment Setup

We design an 11×11 constrained gridworld (Figure Aa) where the agent navigates from start (orange) to goal (green) while avoiding 3×3 unsafe zones (olive) along the diagonal. The environment has stochastic transitions (0.99 success), $\gamma = 0.95$, reward 1 at the goal, cost 1 in unsafe zones, and cost threshold $\delta_c = 10^{-4}$. This tabular domain enables exact computation of stationary distributions, isolating constraint effects without function approximation.

Constructing d_{P} and d_{S} To systematically vary demonstration quality, we define:

- $d^*_{\mathcal{M}}$: stationary distribution of the unconstrained MDP-optimal policy.
- $d^*_{\mathcal{M}_c}$: stationary distribution of the CMDP-optimal policy.
- $d^o_{\mathcal{M}_c}$: stationary distribution of a safe policy with performance level $o \in [0, 1]$.

We construct $d_{\mathrm{P}} = (1 - \xi)\, d^*_{\mathcal{M}} + \xi\, d^*_{\mathcal{M}_c}$ where $\xi \in [0, 1]$ controls safeness, and $d_{\mathrm{S}} = d^o_{\mathcal{M}_c}$ where o controls performance. This allows us to independently control the *safeness* of d_{P} and the *performance* of d_{S}.

Hyperparameters For the distribution constraint, we sweep $\alpha \in [10^{-2}, 10^{2}]$. For the support constraint, we define $\mathbb{1}_{d_{\mathrm{S}}}$ by thresholding low-probability state-action pairs, with thresholds in $[10^{-10}, 10^{-4}]$. We report the highest return among feasible policies (i.e., $J(\pi; c) \leq \delta_c$); if none exist, we report the policy with the lowest cost.

C.2 Results

We systematically vary the safeness of d_{P} and the performance of d_{S}, then solve each objective exactly. From each solution d, we extract a policy via $\pi(a|s) = d(s, a) / \sum_{\bar{a}} d(s, \bar{a})$ and evaluate its return and cost. All results are averaged over 50 random seeds with randomized constraint zones.

Figure Ab reveals a clear behavioral difference between the two constraint types:

- **Distribution constraint** pulls the learned distribution toward d_{S}, causing the policy to inherit suboptimal behaviors when d_{S} has low performance. Even with high-quality d_{P}, the regularization toward d_{S} limits the achievable return.
- **Support constraint** restricts only *where* the policy can act, not *how often*. This allows the policy to align with d_{P} within safe regions, achieving high returns even when d_{S} is suboptimal.

This confirms the key advantage of support constraints: they decouple safety from suboptimality, enabling robust performance across varying demonstration qualities.

D Proofs of Propositions

D.1 Derivation of the Lagrangian Objective $L(w, \nu, \lambda)$

Throughout this section, d ranges over the domain $\mathcal{C}$ of normalized nonnegative state-action distributions that are supported on $\mathrm{supp}(d_{\mathrm{S}})$, absolutely continuous with respect to d_{P}, and have finite $D_{\mathrm{KL}}(d\|d_{\mathrm{P}})$. Every $d \in \mathcal{C}$ therefore assigns zero mass outside $\mathrm{supp}(d_{\mathrm{S}}) \cap \mathrm{supp}(d_{\mathrm{P}})$. Since $d_{\mathrm{P}} \ll d_{\mathrm{U}}$ (Section 4), $d \ll d_{\mathrm{P}}$ implies $d \ll d_{\mathrm{U}}$, so the correction ratio $w = d/d_{\mathrm{U}}$ is well-defined d_{U}-almost everywhere. Starting from the constrained problem in Eq. (2), we introduce Lagrange multipliers $\lambda \in \mathbb{R}$ for the safe-support constraint and $\nu(s) \in \mathbb{R}$ for the Bellman flow constraints:

$$\min_{d \geq 0} \max_{\nu, \lambda} D_{\mathrm{KL}}(d\|d_{\mathrm{P}}) + \lambda \left(1 - \sum_{s,a} \mathbb{1}_{d_{\mathrm{S}}}(s,a) d(s,a)\right) + \sum_{s} \nu(s)\big((\mathcal{B}_* d)(s) - (1-\gamma)\rho_0(s) - \gamma(\mathcal{T}_* d)(s)\big). \tag{12}$$

Flipping the sign and rearranging terms yields the equivalent max-min form

$$\max_{d \geq 0} \min_{\nu, \lambda} (1-\gamma)\mathbb{E}_{s\sim\rho_0}[\nu(s)] - \lambda + \mathbb{E}_{(s,a)\sim d}\Big[\gamma(\mathcal{T}\nu)(s,a) - \nu(s) - \log \frac{d(s,a)}{d_{\mathrm{P}}(s,a)} + \lambda \mathbb{1}_{d_{\mathrm{S}}}(s,a)\Big], \tag{13}$$

where $(\mathcal{T}\nu)(s,a) := \mathbb{E}_{s'\sim T(s,a)}[\nu(s')]$. Applying the change of variables $w(s,a) := d(s,a)/d_{\mathrm{U}}(s,a)$ and defining $r_{\mathrm{P}}(s,a) := d_{\mathrm{P}}(s,a)/d_{\mathrm{U}}(s,a)$, we obtain

$$\begin{aligned}&(1-\gamma)\mathbb{E}_{s\sim\rho_0}[\nu(s)] - \lambda + \mathbb{E}_{(s,a)\sim d_{\mathrm{U}}}\left[w(s,a)\cdot(-\log w(s,a) + A_{\nu,\lambda}(s,a))\right] \\ &\quad =: L(w,\nu,\lambda),\end{aligned} \tag{14}$$

which is Eq. (3) in the main text, and where the advantage term is

$$A_{\nu,\lambda}(s,a) := \gamma(\mathcal{T}\nu)(s,a) - \nu(s) + \log r_{\mathrm{P}}(s,a) + \lambda \mathbb{1}_{d_{\mathrm{S}}}(s,a). \tag{15}$$

The DexDICE objective is then $\min_{\nu,\lambda} \max_{w\geq 0} L(w,\nu,\lambda)$, which by strong duality (Proposition below) and the closed-form solution $w^*_{\nu,\lambda}(s,a) = \exp(A_{\nu,\lambda}(s,a) - 1)$ reduces to the convex minimization in Eq. (7).

D.2 Strong Duality and Closed-Form Solution

Rewrite an objective (7):

$$L(w,\nu,\lambda) = (1-\gamma)\mathbb{E}_{s\sim\rho_0}[\nu(s)] + \mathbb{E}_{(s,a)\sim d_U}\left[w(s,a)(A_{\nu,\lambda}(s,a) - \log w(s,a))\right] - \lambda.$$

Proposition 1. *(Strong Duality) $L(w,\nu,\lambda)$ satisfies strong duality which ensures:*

$$\max_{w\geq 0}\min_{\nu,\lambda} L(w,\nu,\lambda) = \min_{\nu,\lambda}\max_{w\geq 0} L(w,\nu,\lambda)$$

Proof. On the domain $\mathcal{C}$ defined in Appendix D.1, the objective $D_{\mathrm{KL}}(d\|d_{\mathrm{P}})$ is convex in d, the safe-support constraint $\sum_{s,a} \mathbb{1}_{d_{\mathrm{S}}}(s,a)d(s,a) = 1$ holds automatically, and the Bellman flow constraints are affine in d. Problem (2) is therefore a convex program with affine constraints only. We assume the standard constraint qualification for this case [7]: the affine set $\{d : (\mathcal{B}_* d)(s) = (1-\gamma)\rho_0(s) + \gamma(\mathcal{T}_* d)(s) \ \ \forall s\}$ intersects the relative interior of $\mathcal{C}$. Under this condition, strong duality holds, so the primal optimum equals the dual optimum and the max–min and min–max values of $L(w,\nu,\lambda)$ coincide. We note that reachability of every state alone does not provide this condition once the support constraint is active, since a strictly feasible d must additionally be supported on $\mathrm{supp}(d_{\mathrm{S}}) \cap \mathrm{supp}(d_{\mathrm{P}})$; the constraint qualification above makes the required feasibility explicit. □

Proposition 2 (Closed-form solution)**.** *For any dual variables ν and λ, the inner maximization*

$$\max_{w\geq 0} L(w,\nu,\lambda)$$

admits the closed-form solution

$$w^*_{\nu,\lambda}(s,a) = \exp\big(A_{\nu,\lambda}(s,a) - 1\big).$$

Proof. Recall the following lemma (Proposition 1 in [39]):

Lemma. Let f be a convex function (e.g. for KL divergence, $f(x) = x \log x$) with strictly increasing derivative f'. Given ν, λ, and $\alpha \geq 0$, define

$$e_{\nu,\lambda}(s,a) := r(s,a) - \lambda c(s,a) + \gamma\, \mathbb{E}_{s' \sim T(s,a)}[\nu(s')] - \nu(s).$$

Then, the maximization

$$\max_{w \geq 0} \mathbb{E}_{(s,a)\sim d_U} \big[\, w(s,a)\, e_{\nu,\lambda}(s,a) - \alpha\, f(w(s,a))\big]$$

is achieved by

$$w^*_{\nu,\lambda}(s,a) = \big(f'\big)^{-1}\big(\tfrac{1}{\alpha}\, e_{\nu,\lambda}(s,a)\big)_+,$$

where $(\cdot)_+ = \max(0, \cdot)$.

Applying this lemma in our setting: we substitute

$$r = \log r_P, \quad c = -\mathbb{1}_{d_S}, \quad \alpha = 1, \quad \delta_c = -1,$$

which yields $e_{\nu,\lambda} = A_{\nu,\lambda}$ and $L = \mathcal{L}$ in the canonical form. Since for KL divergence one has $f(x) = x \log x$ and thus $(f')^{-1}(y) = \exp(y-1)$, we conclude

$$w^*_{\nu,\lambda}(s,a) = \exp\big(A_{\nu,\lambda}(s,a) - 1\big).$$

□

E Safety Guarantee of DexDICE

This section provides the full statement and proof of the safety guarantee informally stated as Theorem 3 in the main paper, relating the policy's unsafe-region visitation to the discriminator's error rates.

Setup and notation. Consider a constrained MDP with state space $\mathcal{S}$ and action space $\mathcal{A}$. We are given three offline datasets $\mathcal{D}_\mathrm{S}, \mathcal{D}_\mathrm{P}, \mathcal{D}_\mathrm{U}$, with stationary state-action distributions $d_\mathrm{S}, d_\mathrm{P}, d_\mathrm{U}$. Let $\mathcal{N}_\mathrm{env} \subseteq \mathcal{S} \times \mathcal{A}$ denote the set of state-action pairs that are unsafe in the environment. We assume *pointwise soundness* of the safe demonstrations, $\mathrm{supp}(d_\mathrm{S}) \cap \mathcal{N}_\mathrm{env} = \emptyset$, i.e., every demonstrated safe pair is environmentally safe. Under this assumption, agreement with the demonstrated support implies environmental safety, so the zero-visitation guarantee below for the unsafe region $\mathbf{N}$ (the reachable complement of the demonstrated safe support) is a statement about environmental safety; without it, the guarantee concerns support agreement only. Define:

- **Safe region:** $\mathbf{S} = \mathrm{supp}(d_\mathrm{S}) = \{(s,a) : d_\mathrm{S}(s,a) > 0\}$ and **(reachable) unsafe region** $\mathbf{N} = \mathrm{supp}(d_\mathrm{U}) \setminus \mathbf{S} = \{(s,a) : d_\mathrm{S}(s,a) = 0,\ d_\mathrm{U}(s,a) > 0\}$. Restricting $\mathbf{N}$ to $\mathrm{supp}(d_\mathrm{U})$ guarantees $d_\mathrm{U}(s,a) > 0$ on $\mathbf{N}$; the residual set $\mathbf{S}^c \setminus \mathrm{supp}(d_\mathrm{U})$ is never visited under Assumption D (Proposition F).
- **Bayes-optimal discriminator:** $r^*(s,a) := d_\mathrm{S}(s,a)/(d_\mathrm{S}(s,a) + d_\mathrm{U}(s,a))$, well-defined on $\mathbf{S} \cup \mathbf{N} = \mathrm{supp}(d_\mathrm{S}) \cup \mathrm{supp}(d_\mathrm{U})$ since the denominator is strictly positive there. By construction $r^*(s,a) > 0$ on $\mathbf{S}$ and $r^*(s,a) = 0$ for all $(s,a) \in \mathbf{N}$ (where $d_\mathrm{S} = 0$ and $d_\mathrm{U} > 0$).
- **Estimated safe support:** $\hat{\mathbf{S}}_\tau = \{(s,a) : \hat{r}(s,a) \geq \tau\}$, where $\hat{r}$ is the learned discriminator.
- **Conditional unsafe distribution:** $d_\mathbf{N}(s,a) := d_\mathrm{U}(s,a)\, \mathbb{1}[(s,a) \in \mathbf{N}]/d_\mathrm{U}(\mathbf{N})$, the conditional distribution of d_U given $(s,a) \in \mathbf{N}$. We assume $d_\mathrm{U}(\mathbf{N}) > 0$ throughout; if $d_\mathrm{U}(\mathbf{N}) = 0$, then $\mathrm{supp}(d_\mathrm{U}) \subseteq \mathbf{S}$, $\mathbf{N} = \emptyset$, and the safety guarantee holds vacuously.
- **False-positive rate:** $\mathrm{FPR}(\tau) = \mathbb{P}_{(s,a)\sim d_\mathbf{N}}[(s,a) \in \hat{\mathbf{S}}_\tau] = d_\mathrm{U}(\mathbf{N} \cap \hat{\mathbf{S}}_\tau)/d_\mathrm{U}(\mathbf{N})$.

Let $n_S := |\mathcal{D}_\mathrm{S}|$ and $\hat{F}(\tau) := \frac{1}{n_S} \sum_{(s,a)\in\mathcal{D}_\mathrm{S}} \mathbb{1}\{\hat{r}(s,a) \leq \tau\}$ denote the empirical CDF of the scores over $\mathcal{D}_\mathrm{S}$. For a user-specified $\kappa \in (0,1)$, $\tau_\kappa = \inf\{\tau : \hat{F}(\tau) \geq \kappa\}$ is the empirical κ-quantile of Eq. (9). A pair is accepted when $\hat{r}(s,a) \geq \tau_\kappa$ and rejected when $\hat{r}(s,a) < \tau_\kappa$. Our analysis considers an ideal support-constrained policy π_κ whose stationary distribution satisfies $\mathrm{supp}(d^{\pi_\kappa}) \subseteq \hat{\mathbf{S}}_{\tau_\kappa}$; The policy extracted by weighted BC need not satisfy this inclusion exactly (quantified and mitigated in Appendix G.11), and the exact guarantee below is stated for π_κ.

Assumption B (i.i.d. calibration sample)**.** The n_S calibration pairs $\{(s_i, a_i)\}_{i=1}^{n_S}$ used to compute τ_κ are i.i.d. draws from d_{S}.

Assumption C (Uniform estimation error)**.** The learned discriminator $\hat{r}$ satisfies $\|\hat{r} - r^*\|_\infty \le \varepsilon$ with probability at least $1 - \delta/2$, where $\varepsilon = \varepsilon(n)$, $n = |\mathcal{D}_{\mathrm{S}}| + |\mathcal{D}_{\mathrm{U}}|$, and $\varepsilon(n) \to 0$ as $n \to \infty$. Denote this event by $\mathcal{E} := \{\|\hat{r} - r^*\|_\infty \le \varepsilon\}$.

Assumption D (Bounded DICE weights)**.** The learned policy π satisfies $d^\pi \ll d_{\mathrm{U}}$ (absolute continuity) and $d^\pi(s, a) \le C_w \cdot d_{\mathrm{U}}(s, a)$ for all (s, a), where $C_w < \infty$ can be controlled by gradient-penalty regularization in the DICE objective.

Assumption E (Safe margin)**.** Define the *safe margin* $\mu = \inf\{r^*(s, a) : (s, a) \in \mathbf{S},\ d_{\mathrm{S}}(s, a) \ge \rho\}$ for a regularity parameter $\rho > 0$, and the *boundary mass* $\kappa_0 = \mathbb{P}_{d_{\mathrm{S}}}(r^*(s, a) \le \mu)$. The safe margin satisfies $\mu > 2\varepsilon$.

Remark. None of the assumptions requires exact recovery of the safe region, its density, or a cost function. Pointwise soundness requires only that $\mathcal{D}_{\mathrm{S}}$ be uncontaminated, not that it cover the full safe region. Assumption C requires a uniform bound on the discriminator score rather than recovery of either density, and Assumption E requires a margin only on the retained safe interior. Both are standard plug-in conditions for thresholded density-ratio classifiers [62]. Since these are population-level conditions, their empirical diagnostics are reported in Appendices G.7 and G.8.

Proposition F (Unsafe-visitation decomposition)**.** *Let π be any policy supported on $\hat{\mathbf{S}}_\tau$, and suppose Assumption D holds. Then*

$$d^\pi(\mathbf{S}^c) \;=\; d^\pi(\mathbf{N}) \;\le\; C_w \cdot \mathrm{FPR}(\tau) \cdot d_{\mathrm{U}}(\mathbf{N}).$$

Proof. Under Assumption D, $d^\pi \ll d_{\mathrm{U}}$, so d^π assigns zero mass to $\mathbf{S}^c \setminus \mathrm{supp}(d_{\mathrm{U}})$ and $d^\pi(\mathbf{S}^c) = d^\pi(\mathbf{N})$. Since π is supported on $\hat{\mathbf{S}}_\tau$, we have $d^\pi(s, a) = 0$ for $(s, a) \notin \hat{\mathbf{S}}_\tau$. Therefore

$$\begin{aligned} d^\pi(\mathbf{N}) &= \sum_{(s,a) \in \mathbf{N} \cap \hat{\mathbf{S}}_\tau} d^\pi(s, a) \;\le\; C_w \sum_{(s,a) \in \mathbf{N} \cap \hat{\mathbf{S}}_\tau} d_{\mathrm{U}}(s, a) \\ &= C_w \cdot d_{\mathrm{U}}(\mathbf{N} \cap \hat{\mathbf{S}}_\tau) \;=\; C_w \cdot \mathrm{FPR}(\tau) \cdot d_{\mathrm{U}}(\mathbf{N}), \end{aligned}$$

where the inequality uses Assumption D and the last equality is the definition of FPR. □

Lemma G (Structural FPR bound)**.** *On the good event $\mathcal{E}$, for any threshold $\tau > \varepsilon$,* $\mathrm{FPR}(\tau) = 0$.

Proof. For any $(s, a) \in \mathbf{N}$, $r^*(s, a) = 0$ by construction. On $\mathcal{E}$, for every $(s, a) \in \mathbf{N}$,

$$\hat{r}(s, a) = \hat{r}(s, a) - r^*(s, a) \;\le\; |\hat{r}(s, a) - r^*(s, a)| \;\le\; \varepsilon \;<\; \tau.$$

Hence $(s, a) \notin \hat{\mathbf{S}}_\tau$ for every $(s, a) \in \mathbf{N}$, giving $\mathrm{FPR}(\tau) = 0$. □

Lemma H (κ-threshold-FPR connection)**.** *Under Assumptions B and E, if $\kappa > \kappa_0 + \sqrt{\log(2/\delta)/(2n_S)}$, then on the intersection of the good event $\mathcal{E}$ with a calibration event $\mathcal{E}_{\mathrm{cal}}$ of probability at least $1 - \delta/2$ over the draw of $\mathcal{D}_{\mathrm{S}}$,*

$$\tau_\kappa \;\ge\; \mu - \varepsilon \;>\; \varepsilon.$$

Proof. On $\mathcal{E}$, any pair (s, a) with $\hat{r}(s, a) \le \mu - \varepsilon$ satisfies $r^*(s, a) \le \hat{r}(s, a) + \varepsilon \le \mu$, hence lies in the fixed boundary set $B := \{(s, a) : r^*(s, a) \le \mu\}$, whose d_{S}-mass is κ_0 by definition.

Since B depends only on r^* and not on $\hat{r}$, the indicators $\mathbb{1}\{(s_i, a_i) \in B\}$ are i.i.d. Bernoulli variables with mean κ_0 under Assumption B. By Hoeffding's inequality, the event $\mathcal{E}_{\mathrm{cal}} := \left\{\frac{1}{n_S}\sum_{i=1}^{n_S} \mathbb{1}\{(s_i, a_i) \in B\} \le \kappa_0 + \sqrt{\log(2/\delta)/(2n_S)}\right\}$ has probability at least $1 - \delta/2$. On $\mathcal{E} \cap \mathcal{E}_{\mathrm{cal}}$, every sample counted by $\hat{F}(\mu - \varepsilon)$ lies in B by Step 1, so

$$\hat{F}(\mu - \varepsilon) \;\le\; \kappa_0 + \sqrt{\log(2/\delta)/(2n_S)} \;<\; \kappa,$$

where the last inequality is the assumed strict slack condition on κ.

By construction, $\hat{F}(\tau_\kappa) \ge \kappa$. If $\tau_\kappa < \mu - \varepsilon$, monotonicity of $\hat{F}$ would give $\kappa \le \hat{F}(\tau_\kappa) \le \hat{F}(\mu - \varepsilon) < \kappa$, a contradiction. Hence $\tau_\kappa \ge \mu - \varepsilon$, and $\mu > 2\varepsilon$ (Assumption E) yields $\tau_\kappa > \varepsilon$. □

Theorem 3 (Safety guarantee of DexDICE; restated)**.** *Suppose Assumptions B–E hold, with Assumption C at failure level $\delta/2$. Let π_κ be a support-constrained solution with $\mathrm{supp}(d^{\pi_\kappa}) \subseteq \hat{\mathbf{S}}_{\tau_\kappa}$, where τ_κ is the empirical κ-quantile of $\hat{r}$ over $\mathcal{D}_\mathrm{S}$ (Eq. 9) and*

$$\kappa \;>\; \kappa_0 + \sqrt{\log(2/\delta)/(2n_S)}.$$

Then

$$d^{\pi_\kappa}(\mathbf{N}) \;=\; 0 \quad \textit{with probability at least } 1-\delta.$$

Proof. By Assumption C, the good event $\mathcal{E}$ has probability at least $1-\delta/2$, and by Lemma H the calibration event $\mathcal{E}_\mathrm{cal}$ has probability at least $1-\delta/2$; a union bound gives $\Pr(\mathcal{E}\cap\mathcal{E}_\mathrm{cal}) \geq 1-\delta$. On $\mathcal{E}\cap\mathcal{E}_\mathrm{cal}$, Lemma H gives $\tau_\kappa > \varepsilon$, and Lemma G then yields $\mathrm{FPR}(\tau_\kappa) = 0$. Applying Proposition F to π_κ, which is supported on $\hat{\mathbf{S}}_{\tau_\kappa}$,

$$d^{\pi_\kappa}(\mathbf{N}) \;\leq\; C_w \cdot \mathrm{FPR}(\tau_\kappa) \cdot d_\mathrm{U}(\mathbf{N}) \;=\; 0.$$

The square-root term in the condition on κ is a finite-sample slack; as $n_S \to \infty$, the condition reduces to the strict population condition $\kappa > \kappa_0$. □

Corollary I (Simultaneous FNR-FPR control)**.** *Under the conditions of Theorem 3, the FNR parameter κ controls both error rates:*

1. ***FNR control (direct):*** *Under acceptance $\hat{r} \geq \tau_\kappa$, the rejected set is $\{\hat{r} < \tau_\kappa\}$, and the empirical FNR on $\mathcal{D}_\mathrm{S}$, $\frac{1}{n_S}\sum_{(s,a)\in\mathcal{D}_\mathrm{S}} \mathbb{1}\{\hat{r}(s,a) < \tau_\kappa\}$, is strictly below κ by construction; hence $\hat{\mathbf{S}}_{\tau_\kappa}$ retains more than a $(1-\kappa)$-fraction of the calibration pairs in $\mathcal{D}_\mathrm{S}$.*
2. ***FPR control (indirect):*** *On $\mathcal{E}$, $\mathrm{FPR}(\tau_\kappa) = 0$ as a consequence of $\tau_\kappa > \varepsilon$ (Lemma H) and the structural property $r^* \equiv 0$ on $\mathbf{N}$ (Lemma G).*

F Details on Practical Algorithm

F.1 Discriminator-based Estimation

Following Kim et al. [31], we train discriminators $\hat{C}_\mathrm{P}, \hat{C}_S : \mathcal{S}\times\mathcal{A} \to [0,1]$ with neural networks to estimate the density ratio $r_\mathrm{P} = d_\mathrm{P}/d_\mathrm{U}$ and $r_\mathrm{S} = d_\mathrm{S}/d_\mathrm{U}$ via the following objectives:

$$\hat{C}_\mathrm{P} \in \underset{C:\mathcal{S}\times\mathcal{A}\to[0,1]}{\arg\max}\; \mathbb{E}_{d_\mathrm{P}}[\log C(s,a)] + \mathbb{E}_{d_\mathrm{U}}[\log(1-C(s,a))] \tag{16}$$

$$\hat{C}_\mathrm{S} \in \underset{C:\mathcal{S}\times\mathcal{A}\to[0,1]}{\arg\max}\; \mathbb{E}_{d_\mathrm{S}}[\log C(s,a)] + \mathbb{E}_{d_\mathrm{U}}[\log(1-C(s,a))] \tag{17}$$

The solutions correspond to $C^*_\mathrm{P}(s,a) = \frac{d_\mathrm{P}(s,a)}{d_\mathrm{P}(s,a)+d_\mathrm{U}(s,a)}, C^*_\mathrm{S}(s,a) = \frac{d_\mathrm{S}(s,a)}{d_\mathrm{S}(s,a)+d_\mathrm{U}(s,a)}$ respectively. From $\hat{C}_\mathrm{P}, \hat{C}_\mathrm{S}$, we estimate the density ratios $\hat{r}_\mathrm{P}, \hat{r}_\mathrm{S}$ as follows:

$$\hat{r}_\mathrm{P}(s,a) = \frac{d_\mathrm{P}(s,a)}{d_\mathrm{U}(s,a)} = \frac{\hat{C}_\mathrm{P}(s,a)}{1-\hat{C}_\mathrm{P}(s,a)}, \quad \hat{r}_\mathrm{S}(s,a) = \frac{d_\mathrm{S}(s,a)}{d_\mathrm{U}(s,a)} = \frac{\hat{C}_\mathrm{S}(s,a)}{1-\hat{C}_\mathrm{S}(s,a)} \tag{18}$$

F.2 Numerical Stability

To improve numerical stability during optimization arising from summations of exponentials, we adopt an alternative log-sum-exp formulation, as proposed by Kim et al. [31].

Proposition J. *(Kim et al. [31]) Define an alternative objective $\tilde{L}(\nu,\lambda)$ as follows:*

$$\tilde{L}(\nu,\lambda) := (1-\gamma)\mathbb{E}_{\rho_0}[\nu(s)] + \log \mathbb{E}_{(s,a)\sim d_\mathrm{U}}\left[\exp\left(A_{\nu,\lambda}(s,a)\right)\right] - \lambda \tag{19}$$

Then, the following holds:

$$\min_\nu L(\nu,\lambda) = \min_\nu \tilde{L}(\nu,\lambda).$$

F.3 Policy Extraction

In tabular settings, we can directly recover the policy from the optimal stationary distribution d^* as:

$$d^*(s,a) = w^*_{\nu,\lambda}(s,a) d_{\mathrm{U}}(s,a), \quad \pi(a|s) = \frac{d^*(s,a)}{\sum_{\bar{a}\in\mathcal{A}} d^*(s,\bar{a})}.$$

However, in continuous spaces, computing $d^*(s) = \int d^*(s,a)da$ is intractable. Thus, following Kim et al. [31], Lee et al. [38], we extract the policy via weighted behavioral cloning:

$$\min_{\pi} \quad J(\pi; w^*_{\nu,\lambda}) := \mathbb{E}_{(s,a)\sim d_{\mathrm{U}}} \left[-w^*_{\nu,\lambda}(s,a) \log \pi(a|s) \right]. \tag{20}$$

F.4 Algorithm Summary

We summarize a practical version of the DexDICE algorithm in **Algorithm 1**.

Algorithm 1 DexDICE (Safe offline imitation learning from **D**ual **ex**pert demonstrations via **DICE**)

input Safe, performant, unlabeled demonstration sets $\mathcal{D}_{\mathrm{S}}, \mathcal{D}_{\mathrm{P}}, \mathcal{D}_{\mathrm{U}}$, and initial-state set $\mathcal{D}_0$
input Discriminator networks $C_{\mathrm{P}}, C_{\mathrm{S}}$ with parameters $\zeta_{\mathrm{P}}, \zeta_{\mathrm{S}}$, and percentile cutoff κ (e.g., 0.01)
input Value network $\nu_\theta(s)$ and policy network π_ϕ with parameters θ, ϕ, and Lagrange multiplier λ
Initialize $\zeta_{\mathrm{P}}, \zeta_{\mathrm{S}}$.
Pretrain $\hat{C}_{\mathrm{P}}$ and $\hat{C}_{\mathrm{S}}$ by optimizing objectives (16, 17).
Obtain $\hat{r}_{\mathrm{P}}$ and $\hat{r}_{\mathrm{S}}$ from Eq. (18).
Obtain $\hat{\mathbb{1}}_{d_{\mathrm{S}}}$ using κ from Eq. (9).
Initialize θ, ϕ, λ.
for each iteration **do**
  Sample mini-batches from $\mathcal{D}_{\mathrm{P}}, \mathcal{D}_{\mathrm{S}}, \mathcal{D}_{\mathrm{U}}, \mathcal{D}_0$.
  Compute gradients for objective (19):
    $g_\theta \leftarrow \nabla_\theta \tilde{L}(\nu_\theta, \lambda), \quad g_\lambda \leftarrow \nabla_\lambda L(\nu_\theta, \lambda).$
  Compute gradient for objective (20):
    $g_\phi \leftarrow \nabla_\phi J\Big(\pi_\phi; w^*_{\nu_\theta,\lambda}\Big).$
  Perform SGD updates:
    $\theta \leftarrow \theta - \eta g_\theta, \quad \phi \leftarrow \phi - \eta g_\phi, \quad \lambda \leftarrow \lambda - \eta g_\lambda.$
end for
output $\nu_\theta \approx \nu^*, \pi_\phi \approx \pi^*$

G Details on Simulation Experiments

This section details the simulation experiments on 38 re-curated DSRL tasks and six controlled Safety Gymnasium tasks. It covers dataset construction (Sections G.1 and G.2), extended baseline comparisons, hyperparameters, data efficiency and quality, support-estimator comparisons, sensitivity to the target FNR and dataset construction thresholds, robustness to noisy safe demonstrations, and deployment-time projection.

G.1 Re-curated DSRL Datasets

The original DSRL datasets [44] were constructed by aggregating demonstrations from policies trained with a range of safe RL algorithms, resulting in a diverse set of pre-collected trajectories. Each trajectory records both its episode return and cost, which allows us to partition the dataset into dual-expert subsets through thresholding. To define these subsets, we adopted cost thresholds from the original DSRL paper (specifically, one of $\{20, 40, 80\}$) and classified trajectories as *safe* if their cumulative cost was below the corresponding threshold, or *unsafe* otherwise. For the return threshold, we selected the 90th percentile of returns among the safe trajectories. Sensitivity to both thresholds is analyzed in Appendix G.10. With these criteria, each trajectory is categorized into one of four groups: `performant` ($\mathcal{D}_{\mathrm{P}}$), `safe` ($\mathcal{D}_{\mathrm{S}}$), `non-preferred` ($\mathcal{D}_{\mathrm{N}}$), or `unlabeled` ($\mathcal{D}_{\mathrm{U}}$), based on its safety and performance relative to the thresholds. The `non-preferred` set is exclusively used in the

SafeDICE experiments. Table B summarizes the characteristics and statistics for each task category in the re-curated DSRL benchmark. For brevity, task names in the tables and analyses are abbreviated where appropriate by omitting repeated prefixes (e.g., 'Safety', 'Safe') and environment-version suffixes (e.g., '-v0', '-v1').

Task	Obs. Dim	Act. Dim	Cost Threshold	Return Threshold	Max Return	Min Return
Safety Gymnasium						
PointCircle1	28	2	40	44.85	61.73	20.07
PointGoal1	60	2	40	21.67	30.07	0.01
PointButton1	76	2	40	27.12	41.19	0.01
PointPush1	76	2	40	9.91	16.52	0.001
PointCircle2	28	2	80	43.10	54.02	20.48
PointGoal2	60	2	80	20.40	27.72	0.003
PointButton2	76	2	80	29.75	42.90	0.01
PointPush2	76	2	40	8.15	14.69	0.001
CarCircle1	40	2	80	17.47	24.94	8.04
CarGoal1	72	2	40	30.08	39.91	0.01
CarButton1	88	2	80	25.15	44.42	0.003
CarPush1	88	2	40	8.24	16.30	0.01
CarCircle2	40	2	80	18.14	25.95	5.11
CarGoal2	72	2	40	18.89	28.90	0.001
CarButton2	88	2	80	23.46	42.00	0.002
CarPush2	88	2	40	6.46	15.14	0.00
HopperVelocity	11	3	40	1607.84	1911.40	37.05
Walker2dVelocity	17	6	40	2718.04	3418.22	18.67
HalfCheetahVelocity	17	6	40	2563.55	2806.93	5.75
AntVelocity	27	8	40	2743.78	2976.28	6.15
SwimmerVelocity	8	2	40	158.47	238.96	0.07
Bullet Safety Gym						
BallRun	8	2	40	556.19	1327.45	26.34
BallCircle	7	2	40	712.45	881.46	0.38
CarRun	8	2	20	566.21	574.65	204.29
CarCircle	7	2	40	435.21	534.31	3.48
DroneRun	18	4	40	411.47	682.83	10.56
DroneCircle	17	4	40	713.46	996.39	207.79
AntRun	34	8	40	664.83	955.48	0.00
AntCircle	33	8	40	225.71	460.71	0.02
MetaDrive						
MetaDrive-easydense	259	2	40	325.86	425.82	19.71
MetaDrive-easymean	259	2	40	339.41	425.82	22.14
MetaDrive-easysparse	259	2	40	327.70	425.80	20.09
MetaDrive-mediumdense	259	2	20	263.84	269.79	15.10
MetaDrive-mediummean	259	2	20	262.39	269.82	17.03
MetaDrive-mediumsparse	259	2	20	263.81	268.80	17.45
MetaDrive-harddense	259	2	20	283.47	486.64	15.39
MetaDrive-hardmean	259	2	20	309.68	486.81	18.11
MetaDrive-hardsparse	259	2	20	311.44	486.81	17.25

Table B: Descriptions of Tasks in Re-curated DSRL Datasets.

We detail the data labeling process for each demonstration category below:

- `safe`: demonstrations whose cumulative cost remains below the predefined cost threshold, regardless of their return.
- `performant`: demonstrations whose return exceeds the predefined return threshold, regardless of their cost.
- `non-preferred`: demonstrations whose cumulative cost exceeds the cost threshold (i.e., unsafe trajectories).
- `unlabeled`: the full set of offline trajectories.

Tables C, D, E, and F summarize the statistics of the re-curated DSRL datasets across 38 tasks with three different environments, reporting the safe ratio, performant ratio, safe-performant ratio, and the total number of trajectories for each demonstration category.

Task	Dataset	Safe Ratio	Performant Ratio	Safe-Performant Ratio	# traj.
Safety Gymnasium					
PointCircle1	`safe`	1.0	0.10	0.10	214
	`performant`	0.03	1.0	0.03	793
	`non-preferred`	0.0	0.87	0.0	884
	`unlabeled`	0.19	0.72	0.02	1098
PointGoal1	`safe`	1.0	0.10	0.10	1364
	`performant`	0.44	1.0	0.44	313
	`non-preferred`	0.0	0.27	0.0	658
	`unlabeled`	0.67	0.15	0.07	2022
PointButton1	`safe`	1.0	0.10	0.10	783
	`performant`	0.24	1.0	0.24	327
	`non-preferred`	0.0	0.17	0.0	1485
	`unlabeled`	0.35	0.14	0.03	2268
PointPush1	`safe`	1.0	0.10	0.10	1095
	`performant`	0.47	1.0	0.47	235
	`non-preferred`	0.0	0.10	0.0	1284
	`unlabeled`	0.46	0.10	0.05	2379
PointCircle2	`safe`	1.0	0.10	0.10	349
	`performant`	0.09	1.0	0.09	409
	`non-preferred`	0.0	0.68	0.0	546
	`unlabeled`	0.39	0.46	0.04	895
PointGoal2	`safe`	1.0	0.10	0.10	1963
	`performant`	0.38	1.0	0.38	525
	`non-preferred`	0.0	0.22	0.0	1479
	`unlabeled`	0.57	0.15	0.06	3442
PointButton2	`safe`	1.0	0.10	0.10	1513
	`performant`	0.39	1.0	0.39	386
	`non-preferred`	0.0	0.13	0.0	1775
	`unlabeled`	0.46	0.12	0.05	3288
PointPush2	`safe`	1.0	0.10	0.10	1218
	`performant`	0.47	1.0	0.47	260
	`non-preferred`	0.0	0.07	0.0	2024
	`unlabeled`	0.38	0.08	0.04	3242

Table C: Statistics for 4 types of datasets for Point tasks of Safety Gymnasium environment in DSRL Datasets.

Task	Dataset	Safe Ratio	Performant Ratio	Safe-Performant Ratio	# traj.
Safety Gymnasium					
CarCircle1	safe	1.0	0.10	0.10	313
	performant	0.04	1.0	0.04	908
	non-preferred	0.0	0.91	0.0	958
	unlabeled	0.25	0.71	0.03	1271
CarGoal1	safe	1.0	0.10	0.10	905
	performant	0.47	1.0	0.47	193
	non-preferred	0.0	0.13	0.0	766
	unlabeled	0.54	0.12	0.05	1671
CarButton1	safe	1.0	0.10	0.10	1094
	performant	0.30	1.0	0.30	368
	non-preferred	0.0	0.17	0.0	1562
	unlabeled	0.41	0.14	0.04	2656
CarPush1	safe	1.0	0.10	0.10	1459
	performant	0.37	1.0	0.37	399
	non-preferred	0.0	0.18	0.0	1412
	unlabeled	0.51	0.14	0.05	2871
CarCircle2	safe	1.0	0.10	0.10	266
	performant	0.06	1.0	0.06	435
	non-preferred	0.0	0.61	0.0	674
	unlabeled	0.28	0.46	0.03	940
CarGoal2	safe	1.0	0.10	0.10	1709
	performant	0.22	1.0	0.22	788
	non-preferred	0.0	0.26	0.0	2396
	unlabeled	0.42	0.19	0.04	4105
CarButton2	safe	1.0	0.10	0.10	1468
	performant	0.20	1.0	0.20	745
	non-preferred	0.0	0.26	0.0	2287
	unlabeled	0.39	0.20	0.04	3755
CarPush2	safe	1.0	0.10	0.10	2004
	performant	0.34	1.0	0.34	588
	non-preferred	0.0	0.16	0.0	2403
	unlabeled	0.45	0.13	0.05	4407
HopperVelocity	safe	1.0	0.10	0.10	461
	performant	0.09	1.0	0.09	501
	non-preferred	0.0	0.26	0.0	1779
	unlabeled	0.21	0.22	0.02	2240
Walker2dVelocity	safe	1.0	0.10	0.10	1020
	performant	0.17	1.0	0.17	584
	non-preferred	0.0	0.28	0.0	1709
	unlabeled	0.37	0.21	0.04	2729
HalfCheetahVelocity	safe	1.0	0.10	0.10	664
	performant	0.13	1.0	0.13	505
	non-preferred	0.0	0.24	0.0	1831
	unlabeled	0.27	0.20	0.03	2495
AntVelocity	safe	1.0	0.10	0.10	802
	performant	0.10	1.0	0.10	782
	non-preferred	0.0	0.48	0.0	1447
	unlabeled	0.36	0.35	0.04	2249
SwimmerVelocity	safe	1.0	0.10	0.10	422
	performant	0.15	1.0	0.15	290
	non-preferred	0.0	0.20	0.0	1264
	unlabeled	0.25	0.17	0.03	1686

Table D: Statistics for 4 types of datasets for remaining tasks of Safety Gymnasium environment in DSRL Datasets.

Task	Dataset	Safe Ratio	Performant Ratio	Safe-Performant Ratio	# traj.
Bullet Safety Gym					
BallRun	`safe`	1.0	0.10	0.10	315
	`performant`	0.07	1.0	0.07	488
	`non-preferred`	0.0	0.73	0.0	625
	`unlabeled`	0.34	0.52	0.03	940
BallCircle	`safe`	1.0	0.10	0.10	385
	`performant`	0.12	1.0	0.12	338
	`non-preferred`	0.0	0.60	0.0	501
	`unlabeled`	0.43	0.38	0.04	886
CarRun	`safe`	1.0	0.10	0.10	470
	`performant`	0.52	1.0	0.52	91
	`non-preferred`	0.0	0.24	0.0	181
	`unlabeled`	0.72	0.14	0.07	651
CarCircle	`safe`	1.0	0.10	0.10	634
	`performant`	0.10	1.0	0.10	633
	`non-preferred`	0.0	0.70	0.0	816
	`unlabeled`	0.44	0.44	0.04	1450
DroneRun	`safe`	1.0	0.10	0.10	933
	`performant`	0.14	1.0	0.14	682
	`non-preferred`	0.0	0.56	0.0	1057
	`unlabeled`	0.47	0.34	0.05	1990
DroneCircle	`safe`	1.0	0.10	0.10	728
	`performant`	0.06	1.0	0.06	1219
	`non-preferred`	0.0	0.96	0.0	1195
	`unlabeled`	0.38	0.63	0.04	1923
AntRun	`safe`	1.0	0.10	0.10	654
	`performant`	0.09	1.0	0.09	727
	`non-preferred`	0.0	0.57	0.0	1162
	`unlabeled`	0.36	0.40	0.04	1816
AntCircle	`safe`	1.0	0.10	0.10	1611
	`performant`	0.05	1.0	0.05	3244
	`non-preferred`	0.0	0.75	0.0	4117
	`unlabeled`	0.28	0.57	0.03	5728

Table E: Statistics for 4 types of datasets for Bullet Safety Gym environment in DSRL Datasets.

Task	Dataset	Safe Ratio	Performant Ratio	Safe-Performant Ratio	# traj.
MetaDrive					
easydense	safe	1.0	0.10	0.10	675
	performant	0.21	1.0	0.21	331
	non-preferred	0.0	0.81	0.0	325
	unlabeled	0.68	0.33	0.07	1000
easymean	safe	1.0	0.10	0.10	634
	performant	0.16	1.0	0.16	388
	non-preferred	0.0	0.89	0.0	366
	unlabeled	0.63	0.39	0.06	1000
easysparse	safe	1.0	0.10	0.10	626
	performant	0.16	1.0	0.16	388
	non-preferred	0.0	0.87	0.0	374
	unlabeled	0.63	0.39	0.06	1000
mediumdense	safe	1.0	0.10	0.10	576
	performant	0.25	1.0	0.25	230
	non-preferred	0.0	0.41	0.0	424
	unlabeled	0.58	0.23	0.06	1000
mediummean	safe	1.0	0.10	0.10	870
	performant	0.52	1.0	0.52	166
	non-preferred	0.0	0.61	0.0	130
	unlabeled	0.87	0.17	0.09	1000
mediumsparse	safe	1.0	0.10	0.10	601
	performant	0.34	1.0	0.34	178
	non-preferred	0.0	0.29	0.0	399
	unlabeled	0.60	0.18	0.06	1000
harddense	safe	1.0	0.10	0.10	582
	performant	0.23	1.0	0.23	260
	non-preferred	0.0	0.48	0.0	418
	unlabeled	0.58	0.26	0.06	1000
hardmean	safe	1.0	0.10	0.10	527
	performant	0.22	1.0	0.22	236
	non-preferred	0.0	0.39	0.0	473
	unlabeled	0.53	0.24	0.05	1000
hardsparse	safe	1.0	0.10	0.10	523
	performant	0.18	1.0	0.18	287
	non-preferred	0.0	0.49	0.0	477
	unlabeled	0.52	0.29	0.05	1000

Table F: Statistics for 4 types of datasets for MetaDrive environment in DSRL Datasets.

G.2 Controlled Datasets

To systematically study the influence of (1) the number of trajectories and (2) the quality of datasets, we construct controlled datasets with varying levels of safety and performance; the Safety Gymnasium tasks used for this purpose are described in Table G. This involves collecting demonstrations from policies exhibiting different performance (e.g., near-optimal, medium) and safety (e.g., safe, unsafe) characteristics. We utilize the OmniSafe framework [29], which provides a suite of safe RL algorithms for the Safety Gymnasium benchmark [28], to generate policies for data collection.

Specifically, we employ PPO [56] for unconstrained MDPs and PPO-Lag [52] for constrained MDPs (CMDPs) with explicit cost limits. Using these algorithms, we train near-optimal policies for both settings: PPO-Lag yields the `safe-high` dataset (cost-constrained, high return) and PPO yields the `unsafe-high` dataset (unconstrained, high return).

To acquire the `safe-medium` and `unsafe-medium` datasets, we identify policies whose performance is approximately half that of the corresponding near-optimal policy. We ensure that the `safe-medium` policies continue to satisfy cost constraints, whereas the `unsafe-medium` policies violate these constraints by design.

For the data efficiency (**Q2**) and data quality (**Q3**) experiments (see Section 5.2), we conduct evaluations on six representative tasks: SafetyPointCircle1, SafetyPointCircle2, SafetyHopperVelocity, SafetyHalfCheetahVelocity, SafetyWalker2dVelocity, and SafetyAntVelocity.

Task	Obs. Dim	Act. Dim	Cost limit
Safe Navigation Tasks			
SafetyPointCircle1	28	2	25
SafetyPointCircle2	28	2	25
Safe Velocity Tasks			
SafetyHopperVelocity	11	3	25
SafetyWalker2dVelocity	17	6	25
SafetyHalfCheetahVelocity	17	6	25
SafetyAntVelocity	27	8	25

Table G: Descriptions of Tasks in Safety Gymnasium.

G.3 Empirical Comparison with Offline Safe Methods in Different Settings

In this section we compare DexDICE with a broader set of baselines from two families that operate under different supervision settings: DICE-based methods (SafeDICE [26], COptiDICE [39], ICSDICE [51]) and offline safe RL methods (CDT [43], FISOR [70], TraC [18]). Despite their differing assumptions, all are evaluated under our safety and return metrics for conceptual and practical comparison.

Setup. SafeDICE treats unsafe demonstrations as non-preferred. COptiDICE, CDT, FISOR, and TraC use ground-truth reward $r(s, a)$ and cost $c(s, a)$ from the dataset. ICSDICE uses true rewards and safe demonstrations $\mathcal{D}_{\mathrm{S}}$. DexDICE is trained using only $\mathcal{D}_{\mathrm{S}}$, $\mathcal{D}_{\mathrm{P}}$, and $\mathcal{D}_{\mathrm{U}}$. Its per-task best κ in Table I is selected post hoc using evaluation return and cost, separately from the fixed-κ main result.

Results. With the fixed $\kappa = 0.01$ used in the main result, DexDICE satisfies the cost constraint on 31 of the 38 tasks (Table H). Table I separately reports the exploratory comparison using DexDICE's post hoc per-task best κ. Within the DICE family, SafeDICE violates the cost constraint on 20 of the 38 tasks since its non-preferred set does not adequately cover the unsafe support, leaving exploitable gaps. COptiDICE, despite ground-truth reward and cost access, violates the constraint on 19 tasks, consistent with the observation in the DSRL paper [44] that behavior-distribution constraints (e.g., $D_f(d\|d_{\mathrm{U}})$) induce conservatism and brittle safety under lower-quality data. ICSDICE attains the highest returns on average but violates the constraint on 22 tasks, concentrated in the velocity, Bullet Safety Gym, and MetaDrive tasks. Among offline safe RL methods, FISOR, FISOR-IL, and TraC satisfy the constraint on 34, 35, and 30 tasks, respectively, under full ground-truth supervision but pay a notable return cost; their constrained-RL formulations bias the policy toward conservative behavior even when the dataset already contains high-return safe trajectories. CDT attains higher returns but violates the constraint on 18 tasks.

Environment	Metric	BC	FBC	JFBC	DemoDICE-S	DemoDICE-U	DWBC-S	FISOR-IL ($\hat{c}$)	CDT ($\bar{r}, \hat{c}$)	TraC ($\bar{r}, \hat{c}$)	DexDICE	DexDICE (best κ)
PointButton1	Episode return	0.49 ± 0.01	0.05 ± 0.04	0.59 ± 0.01	**0.05 ± 0.02**	**0.11 ± 0.02**	0.48 ± 0.01	0.57 ± 0.01	0.40 ± 0.10	0.10 ± 0.13	**0.15 ± 0.02**	**0.15 ± 0.02**
	Episode cost	2.93 ± 0.09	1.81 ± 0.28	3.23 ± 0.10	**0.51 ± 0.07**	**0.95 ± 0.14**	3.02 ± 0.10	3.10 ± 0.11	2.23 ± 0.38	1.81 ± 1.31	**0.95 ± 0.09**	**0.95 ± 0.09**
PointButton2	Episode return	0.41 ± 0.03	0.17 ± 0.03	0.56 ± 0.01	**0.21 ± 0.02**	**0.22 ± 0.02**	0.42 ± 0.02	0.56 ± 0.01	0.37 ± 0.09	0.23 ± 0.14	**0.27 ± 0.02**	**0.27 ± 0.02**
	Episode cost	1.58 ± 0.04	1.90 ± 0.09	1.70 ± 0.06	**0.69 ± 0.06**	**0.73 ± 0.06**	1.60 ± 0.08	1.82 ± 0.02	1.14 ± 0.25	1.33 ± 0.27	**0.92 ± 0.08**	**0.92 ± 0.08**
PointCircle1	Episode return	0.75 ± 0.01	−0.27 ± 0.15	0.87 ± 0.01	**0.29 ± 0.09**	0.81 ± 0.01	0.73 ± 0.04	0.84 ± 0.01	0.30 ± 0.08	**0.09 ± 0.02**	**0.27 ± 0.03**	**0.27 ± 0.03**
	Episode cost	4.37 ± 0.15	5.10 ± 0.84	4.73 ± 0.06	**0.93 ± 0.38**	4.74 ± 0.20	3.49 ± 0.46	4.46 ± 0.03	1.41 ± 0.45	**0.01 ± 0.03**	**0.27 ± 0.07**	**0.27 ± 0.07**
PointCircle2	Episode return	0.58 ± 0.13	−0.36 ± 0.21	0.82 ± 0.01	**0.51 ± 0.02**	0.82 ± 0.01	0.55 ± 0.06	0.63 ± 0.04	0.65 ± 0.05	**−0.04 ± 0.22**	0.61 ± 0.01	**0.60 ± 0.02**
	Episode cost	2.77 ± 0.11	1.93 ± 0.58	2.94 ± 0.05	**0.85 ± 0.09**	2.94 ± 0.05	2.64 ± 0.27	3.03 ± 0.08	1.26 ± 0.58	**0.73 ± 0.29**	1.30 ± 0.07	**0.93 ± 0.19**
PointGoal1	Episode return	**0.57 ± 0.01**	0.18 ± 0.03	0.71 ± 0.01	**0.25 ± 0.03**	**0.67 ± 0.01**	**0.58 ± 0.02**	**0.72 ± 0.00**	**0.57 ± 0.01**	**0.15 ± 0.11**	**0.35 ± 0.01**	**0.41 ± 0.02**
	Episode cost	**0.92 ± 0.03**	1.15 ± 0.16	1.00 ± 0.04	**0.35 ± 0.04**	**0.88 ± 0.04**	**0.94 ± 0.04**	**0.99 ± 0.02**	**0.75 ± 0.07**	**0.63 ± 0.34**	**0.40 ± 0.04**	**0.44 ± 0.05**
PointGoal2	Episode return	0.52 ± 0.02	0.15 ± 0.02	0.66 ± 0.01	**0.18 ± 0.01**	0.55 ± 0.01	0.49 ± 0.01	0.63 ± 0.01	**0.41 ± 0.06**	**0.18 ± 0.08**	**0.35 ± 0.02**	**0.38 ± 0.02**
	Episode cost	1.16 ± 0.09	1.30 ± 0.10	1.35 ± 0.01	**0.40 ± 0.04**	1.14 ± 0.05	1.19 ± 0.05	1.34 ± 0.01	**0.81 ± 0.10**	**0.93 ± 0.58**	**0.61 ± 0.05**	**0.68 ± 0.02**
PointPush1	Episode return	**0.20 ± 0.01**	**0.03 ± 0.01**	0.26 ± 0.01	**0.07 ± 0.01**	**0.23 ± 0.00**	**0.15 ± 0.01**	**0.24 ± 0.01**	**0.17 ± 0.03**	**0.01 ± 0.03**	**0.16 ± 0.01**	**0.19 ± 0.01**
	Episode cost	**0.89 ± 0.03**	**0.98 ± 0.19**	1.13 ± 0.09	**0.19 ± 0.01**	**0.78 ± 0.04**	**0.76 ± 0.03**	**0.80 ± 0.05**	**0.73 ± 0.11**	**0.72 ± 0.24**	**0.50 ± 0.03**	**0.57 ± 0.08**
PointPush2	Episode return	0.17 ± 0.01	−0.01 ± 0.01	0.20 ± 0.01	**0.09 ± 0.01**	**0.17 ± 0.01**	0.16 ± 0.01	0.16 ± 0.01	**0.09 ± 0.03**	**−0.21 ± 0.27**	0.12 ± 0.04	**0.15 ± 0.03**
	Episode cost	1.49 ± 0.14	1.29 ± 0.27	1.35 ± 0.12	**0.50 ± 0.07**	**0.98 ± 0.09**	1.09 ± 0.04	1.34 ± 0.04	**0.80 ± 0.29**	**0.60 ± 0.33**	1.08 ± 0.19	**0.96 ± 0.10**
CarButton1	Episode return	0.17 ± 0.01	0.01 ± 0.02	0.30 ± 0.02	**0.03 ± 0.05**	**−0.06 ± 0.03**	0.15 ± 0.01	0.32 ± 0.01	0.18 ± 0.08	−0.02 ± 0.17	**0.04 ± 0.05**	**0.13 ± 0.02**
	Episode cost	1.91 ± 0.08	1.62 ± 0.17	1.88 ± 0.06	**0.91 ± 0.14**	**0.63 ± 0.05**	1.77 ± 0.07	2.33 ± 0.06	1.31 ± 0.36	1.37 ± 0.85	**0.63 ± 0.05**	**0.81 ± 0.09**
CarButton2	Episode return	0.25 ± 0.02	−0.02 ± 0.01	0.38 ± 0.02	0.08 ± 0.01	**−0.09 ± 0.03**	0.23 ± 0.01	0.43 ± 0.00	0.26 ± 0.05	−0.05 ± 0.10	**0.08 ± 0.02**	**0.08 ± 0.02**
	Episode cost	2.15 ± 0.05	1.64 ± 0.17	2.85 ± 0.11	1.13 ± 0.12	**0.50 ± 0.03**	2.31 ± 0.08	3.05 ± 0.05	2.10 ± 0.27	1.56 ± 0.24	**0.86 ± 0.13**	**0.86 ± 0.13**
CarCircle1	Episode return	0.25 ± 0.04	−0.06 ± 0.05	0.72 ± 0.01	0.21 ± 0.05	0.70 ± 0.01	0.29 ± 0.06	0.67 ± 0.00	0.64 ± 0.04	−0.09 ± 0.12	0.53 ± 0.03	0.56 ± 0.03
	Episode cost	1.46 ± 0.20	1.69 ± 0.45	2.20 ± 0.12	1.13 ± 0.26	2.52 ± 0.06	1.56 ± 0.14	2.62 ± 0.04	2.44 ± 0.22	1.05 ± 0.42	1.27 ± 0.22	1.37 ± 0.17
CarCircle2	Episode return	0.36 ± 0.02	0.19 ± 0.04	0.63 ± 0.04	0.49 ± 0.02	0.71 ± 0.01	0.34 ± 0.02	0.75 ± 0.00	0.65 ± 0.05	−0.33 ± 0.04	0.58 ± 0.02	0.63 ± 0.01
	Episode cost	1.88 ± 0.21	2.66 ± 0.23	2.87 ± 0.15	1.55 ± 0.20	2.90 ± 0.08	1.24 ± 0.16	3.28 ± 0.05	2.38 ± 0.76	3.75 ± 0.95	2.00 ± 0.12	2.01 ± 0.13
CarGoal1	Episode return	0.48 ± 0.01	0.25 ± 0.02	0.68 ± 0.01	**0.14 ± 0.01**	**0.45 ± 0.01**	**0.43 ± 0.00**	0.71 ± 0.01	**0.48 ± 0.09**	**−0.02 ± 0.06**	**0.31 ± 0.01**	**0.31 ± 0.01**
	Episode cost	1.22 ± 0.06	1.39 ± 0.08	1.23 ± 0.05	**0.24 ± 0.03**	**0.62 ± 0.02**	**0.96 ± 0.05**	1.38 ± 0.06	**0.77 ± 0.22**	**0.65 ± 0.47**	**0.32 ± 0.02**	**0.32 ± 0.02**
CarGoal2	Episode return	0.36 ± 0.01	0.02 ± 0.02	0.51 ± 0.01	**0.18 ± 0.01**	**0.29 ± 0.01**	0.36 ± 0.01	0.50 ± 0.01	0.28 ± 0.04	**−0.07 ± 0.14**	**0.24 ± 0.02**	**0.26 ± 0.01**
	Episode cost	2.71 ± 0.13	1.80 ± 0.22	2.68 ± 0.09	**0.97 ± 0.10**	**0.99 ± 0.08**	2.58 ± 0.12	2.42 ± 0.06	1.24 ± 0.35	**0.58 ± 0.60**	**0.81 ± 0.08**	**0.90 ± 0.09**
CarPush1	Episode return	**0.20 ± 0.01**	**0.03 ± 0.02**	**0.28 ± 0.01**	**0.14 ± 0.01**	**0.22 ± 0.01**	**0.18 ± 0.01**	**0.27 ± 0.01**	**0.21 ± 0.02**	**−0.01 ± 0.02**	**0.19 ± 0.01**	**0.21 ± 0.00**
	Episode cost	**0.95 ± 0.05**	**0.35 ± 0.06**	**0.92 ± 0.09**	**0.30 ± 0.03**	**0.43 ± 0.05**	**0.85 ± 0.07**	**0.92 ± 0.06**	**0.52 ± 0.10**	**0.22 ± 0.23**	**0.33 ± 0.04**	**0.30 ± 0.02**
CarPush2	Episode return	0.07 ± 0.00	0.02 ± 0.02	0.17 ± 0.01	**0.03 ± 0.02**	0.11 ± 0.00	0.09 ± 0.00	0.17 ± 0.00	0.10 ± 0.02	0.04 ± 0.03	**0.10 ± 0.00**	**0.12 ± 0.01**
	Episode cost	2.49 ± 0.21	2.55 ± 0.28	2.05 ± 0.13	**0.61 ± 0.03**	1.15 ± 0.10	1.96 ± 0.11	2.08 ± 0.08	1.20 ± 0.44	1.34 ± 1.05	**0.91 ± 0.10**	**0.93 ± 0.10**
SwimmerVelocity	Episode return	0.73 ± 0.01	**0.68 ± 0.00**	0.73 ± 0.01	**0.28 ± 0.04**	0.81 ± 0.01	**0.69 ± 0.00**	0.70 ± 0.02	0.66 ± 0.01	**0.67 ± 0.00**	**0.62 ± 0.01**	**0.62 ± 0.01**
	Episode cost	2.71 ± 0.23	**0.35 ± 0.03**	3.55 ± 0.46	**0.58 ± 0.07**	4.66 ± 0.36	**0.60 ± 0.13**	2.35 ± 0.12	3.11 ± 0.50	**0.14 ± 0.23**	**0.65 ± 0.28**	**0.65 ± 0.28**
HopperVelocity	Episode return	0.35 ± 0.11	**0.84 ± 0.00**	0.59 ± 0.12	**0.08 ± 0.03**	0.29 ± 0.09	0.38 ± 0.03	0.70 ± 0.02	**0.15 ± 0.02**	**0.73 ± 0.13**	**0.36 ± 0.09**	**0.47 ± 0.07**
	Episode cost	1.71 ± 0.32	**0.00 ± 0.00**	2.97 ± 0.60	**0.52 ± 0.23**	2.39 ± 0.96	2.28 ± 0.21	3.30 ± 0.12	**0.64 ± 0.12**	**0.51 ± 0.41**	**0.40 ± 0.16**	**0.60 ± 0.12**
HalfCheetahVelocity	Episode return	0.98 ± 0.00	**0.66 ± 0.07**	0.99 ± 0.00	**0.76 ± 0.01**	0.99 ± 0.00	0.98 ± 0.00	0.97 ± 0.00	0.92 ± 0.01	**0.96 ± 0.01**	**0.76 ± 0.01**	**0.76 ± 0.01**
	Episode cost	1.92 ± 0.41	**0.13 ± 0.06**	2.82 ± 0.73	**0.74 ± 0.14**	4.30 ± 0.80	1.44 ± 0.28	1.61 ± 0.14	1.74 ± 0.78	**0.01 ± 0.02**	**0.45 ± 0.12**	**0.45 ± 0.12**
Walker2dVelocity	Episode return	0.83 ± 0.02	**0.25 ± 0.04**	0.80 ± 0.02	**0.73 ± 0.02**	0.77 ± 0.01	0.78 ± 0.02	0.75 ± 0.03	0.60 ± 0.02	**0.78 ± 0.00**	**0.79 ± 0.01**	**0.79 ± 0.01**
	Episode cost	3.27 ± 0.07	**0.49 ± 0.06**	3.35 ± 0.04	**0.94 ± 0.26**	2.25 ± 0.21	2.96 ± 0.09	3.05 ± 0.09	2.40 ± 0.34	**0.00 ± 0.01**	**0.07 ± 0.02**	**0.07 ± 0.02**
AntVelocity	Episode return	0.99 ± 0.00	**−0.02 ± 0.01**	0.99 ± 0.00	**0.96 ± 0.00**	1.00 ± 0.00	0.97 ± 0.01	0.99 ± 0.00	0.91 ± 0.02	**0.90 ± 0.01**	**0.96 ± 0.00**	**0.96 ± 0.00**
	Episode cost	1.34 ± 0.22	**0.04 ± 0.01**	1.31 ± 0.15	**0.30 ± 0.05**	2.18 ± 0.38	1.33 ± 0.13	2.03 ± 0.04	1.58 ± 0.35	**0.08 ± 0.01**	**0.15 ± 0.03**	**0.15 ± 0.03**
BallRun	Episode return	0.44 ± 0.08	0.32 ± 0.10	0.70 ± 0.07	0.64 ± 0.19	**−0.01 ± 0.00**	0.52 ± 0.03	0.57 ± 0.01	**0.43 ± 0.21**	**0.31 ± 0.00**	**0.38 ± 0.05**	**0.41 ± 0.04**
	Episode cost	1.18 ± 0.16	1.01 ± 0.05	1.61 ± 0.15	1.36 ± 0.38	**0.00 ± 0.00**	1.14 ± 0.11	1.41 ± 0.03	**0.79 ± 0.78**	**0.48 ± 0.09**	**0.89 ± 0.18**	**0.86 ± 0.19**
CarRun	Episode return	**0.97 ± 0.01**	**0.99 ± 0.00**	**0.99 ± 0.00**	0.64 ± 0.05	**0.36 ± 0.06**	**0.99 ± 0.00**	**0.99 ± 0.00**	0.97 ± 0.00	**0.99 ± 0.00**	**0.96 ± 0.00**	**0.97 ± 0.01**
	Episode cost	**0.45 ± 0.15**	**0.51 ± 0.04**	**0.37 ± 0.05**	1.79 ± 0.58	**0.42 ± 0.32**	**0.16 ± 0.02**	**0.37 ± 0.02**	1.19 ± 0.35	**0.48 ± 0.38**	**0.07 ± 0.03**	**0.79 ± 0.30**
DroneRun	Episode return	0.56 ± 0.09	0.59 ± 0.03	0.66 ± 0.07	0.48 ± 0.03	0.77 ± 0.01	0.64 ± 0.05	0.74 ± 0.01	**0.53 ± 0.06**	**0.55 ± 0.01**	0.60 ± 0.01	**0.60 ± 0.01**
	Episode cost	2.34 ± 0.16	2.57 ± 0.21	2.41 ± 0.36	2.55 ± 0.14	2.15 ± 0.06	2.67 ± 0.13	2.48 ± 0.04	**0.67 ± 0.88**	**0.00 ± 0.00**	1.07 ± 0.29	**0.68 ± 0.20**
AntRun	Episode return	0.54 ± 0.02	**0.10 ± 0.01**	0.68 ± 0.03	**0.42 ± 0.03**	0.77 ± 0.01	0.56 ± 0.02	0.56 ± 0.03	**0.26 ± 0.01**	**0.33 ± 0.05**	**0.55 ± 0.03**	**0.55 ± 0.03**
	Episode cost	1.60 ± 0.12	**0.02 ± 0.00**	1.99 ± 0.17	**0.19 ± 0.04**	1.86 ± 0.18	1.68 ± 0.14	1.47 ± 0.13	**0.09 ± 0.03**	**0.02 ± 0.01**	**0.46 ± 0.13**	**0.46 ± 0.13**
BallCircle	Episode return	0.92 ± 0.01	**0.82 ± 0.00**	0.94 ± 0.01	**0.56 ± 0.02**	0.90 ± 0.00	0.84 ± 0.02	0.90 ± 0.00	**0.64 ± 0.02**	**0.41 ± 0.04**	**0.60 ± 0.02**	**0.65 ± 0.02**
	Episode cost	1.44 ± 0.02	**0.94 ± 0.04**	1.44 ± 0.03	**0.48 ± 0.05**	1.34 ± 0.02	1.23 ± 0.04	1.42 ± 0.01	**0.90 ± 0.09**	**0.32 ± 0.12**	**0.43 ± 0.04**	**0.54 ± 0.05**
CarCircle	Episode return	0.86 ± 0.04	0.77 ± 0.01	0.92 ± 0.01	**0.56 ± 0.02**	0.92 ± 0.00	0.84 ± 0.01	0.31 ± 0.05	0.50 ± 0.03	0.13 ± 0.02	**0.70 ± 0.01**	**0.70 ± 0.01**
	Episode cost	2.27 ± 0.06	1.20 ± 0.08	2.19 ± 0.04	**0.52 ± 0.16**	2.15 ± 0.02	1.85 ± 0.08	1.41 ± 0.12	1.14 ± 0.08	1.52 ± 0.78	**0.82 ± 0.03**	**0.82 ± 0.03**
DroneCircle	Episode return	0.06 ± 0.03	−0.12 ± 0.01	0.52 ± 0.05	**0.18 ± 0.01**	0.78 ± 0.01	0.36 ± 0.06	0.38 ± 0.03	0.04 ± 0.03	−0.07 ± 0.08	**0.54 ± 0.01**	**0.54 ± 0.01**
	Episode cost	1.25 ± 0.13	1.12 ± 0.06	1.77 ± 0.12	**0.81 ± 0.06**	1.61 ± 0.04	1.61 ± 0.09	1.57 ± 0.06	1.91 ± 0.31	1.26 ± 1.05	**0.80 ± 0.04**	**0.80 ± 0.04**
AntCircle	Episode return	0.20 ± 0.01	**0.09 ± 0.01**	0.45 ± 0.01	**0.19 ± 0.01**	0.57 ± 0.02	**0.20 ± 0.01**	0.35 ± 0.01	0.16 ± 0.04	**0.13 ± 0.02**	**0.35 ± 0.01**	**0.35 ± 0.01**
	Episode cost	1.27 ± 0.10	**0.47 ± 0.10**	1.22 ± 0.08	**0.49 ± 0.05**	1.77 ± 0.07	**0.94 ± 0.05**	1.17 ± 0.09	1.27 ± 0.38	**0.13 ± 0.05**	**0.48 ± 0.06**	**0.48 ± 0.06**
MetaDrive-easysparse	Episode return	0.92 ± 0.07	**0.38 ± 0.05**	0.99 ± 0.00	**0.01 ± 0.02**	0.99 ± 0.01	0.98 ± 0.02	0.99 ± 0.00	**0.31 ± 0.25**	**−0.01 ± 0.02**	**0.23 ± 0.07**	**0.56 ± 0.09**
	Episode cost	2.06 ± 0.13	**0.50 ± 0.07**	2.11 ± 0.06	**0.15 ± 0.03**	1.87 ± 0.11	2.13 ± 0.05	2.05 ± 0.01	**0.02 ± 0.00**	**0.03 ± 0.00**	**0.33 ± 0.05**	**0.40 ± 0.20**
MetaDrive-easymean	Episode return	0.97 ± 0.01	**0.25 ± 0.06**	0.86 ± 0.07	**0.53 ± 0.03**	0.86 ± 0.09	0.93 ± 0.01	0.84 ± 0.01	**0.47 ± 0.27**	**0.13 ± 0.11**	**0.70 ± 0.04**	**0.73 ± 0.01**
	Episode cost	2.01 ± 0.02	**0.39 ± 0.06**	1.70 ± 0.08	**0.33 ± 0.02**	1.66 ± 0.25	1.98 ± 0.03	1.86 ± 0.03	**0.02 ± 0.00**	**0.03 ± 0.00**	**0.45 ± 0.02**	**0.51 ± 0.03**
MetaDrive-easydense	Episode return	0.87 ± 0.02	**0.48 ± 0.02**	0.61 ± 0.03	**0.46 ± 0.11**	0.69 ± 0.03	0.84 ± 0.02	0.56 ± 0.02	**0.26 ± 0.09**	**0.24 ± 0.04**	**0.56 ± 0.04**	**0.56 ± 0.04**
	Episode cost	1.60 ± 0.02	**0.49 ± 0.02**	1.35 ± 0.04	**0.31 ± 0.16**	1.45 ± 0.03	1.58 ± 0.03	1.19 ± 0.03	**0.03 ± 0.00**	**0.03 ± 0.00**	**0.25 ± 0.04**	**0.25 ± 0.04**
MetaDrive-mediumsparse	Episode return	0.86 ± 0.02	0.28 ± 0.04	0.82 ± 0.09	**0.24 ± 0.08**	0.81 ± 0.03	0.78 ± 0.09	0.87 ± 0.02	**0.18 ± 0.01**	**0.11 ± 0.11**	0.79 ± 0.05	**0.74 ± 0.09**
	Episode cost	1.57 ± 0.11	1.02 ± 0.05	1.94 ± 0.19	**0.45 ± 0.08**	1.94 ± 0.11	1.44 ± 0.21	2.16 ± 0.01	**0.05 ± 0.00**	**0.05 ± 0.00**	1.29 ± 0.08	**0.88 ± 0.15**
MetaDrive-mediummean	Episode return	0.94 ± 0.00	**0.05 ± 0.06**	0.66 ± 0.14	**0.44 ± 0.08**	0.89 ± 0.03	0.92 ± 0.01	0.80 ± 0.03	**0.21 ± 0.18**	**0.65 ± 0.27**	**0.51 ± 0.14**	**0.51 ± 0.14**
	Episode cost	1.64 ± 0.07	**0.25 ± 0.09**	1.70 ± 0.31	**0.27 ± 0.03**	1.90 ± 0.09	1.56 ± 0.11	1.99 ± 0.08	**0.05 ± 0.00**	**0.04 ± 0.01**	**0.82 ± 0.14**	**0.82 ± 0.14**
MetaDrive-mediumdense	Episode return	0.85 ± 0.02	**0.27 ± 0.04**	0.85 ± 0.03	**0.16 ± 0.04**	0.88 ± 0.01	0.89 ± 0.00	0.42 ± 0.08	**0.16 ± 0.08**	**0.06 ± 0.12**	**0.52 ± 0.07**	**0.53 ± 0.08**
	Episode cost	1.81 ± 0.05	**0.47 ± 0.08**	2.03 ± 0.08	**0.05 ± 0.01**	1.98 ± 0.04	1.77 ± 0.04	1.08 ± 0.14	**0.05 ± 0.00**	**0.05 ± 0.00**	**0.48 ± 0.15**	**0.73 ± 0.14**
MetaDrive-hardsparse	Episode return	0.48 ± 0.02	**0.24 ± 0.08**	0.56 ± 0.01	**0.38 ± 0.01**	0.46 ± 0.03	0.51 ± 0.01	0.44 ± 0.00	**0.18 ± 0.08**	**0.40 ± 0.11**	**0.35 ± 0.02**	**0.35 ± 0.02**
	Episode cost	1.79 ± 0.07	**0.44 ± 0.11**	2.31 ± 0.12	**0.03 ± 0.00**	1.91 ± 0.09	1.84 ± 0.09	1.29 ± 0.03	**0.05 ± 0.00**	**0.04 ± 0.01**	**0.13 ± 0.03**	**0.13 ± 0.03**
MetaDrive-hardmean	Episode return	0.42 ± 0.04	**0.05 ± 0.05**	0.44 ± 0.03	**0.43 ± 0.02**	0.50 ± 0.02	0.42 ± 0.03	0.36 ± 0.01	**0.17 ± 0.10**	**0.30 ± 0.07**	0.47 ± 0.02	**0.47 ± 0.01**
	Episode cost	1.66 ± 0.09	**0.39 ± 0.12**	2.08 ± 0.09	**0.45 ± 0.02**	1.97 ± 0.08	1.62 ± 0.08	1.91 ± 0.04	**0.05 ± 0.00**	**0.05 ± 0.00**	1.76 ± 0.12	**0.85 ± 0.05**
MetaDrive-harddense	Episode return	**0.52 ± 0.01**	**0.28 ± 0.02**	0.52 ± 0.01	**0.30 ± 0.05**	0.33 ± 0.05	0.40 ± 0.04	0.34 ± 0.01	**0.17 ± 0.11**	**0.18 ± 0.12**	**0.34 ± 0.04**	**0.36 ± 0.03**
	Episode cost	**0.98 ± 0.04**	**0.59 ± 0.02**	1.46 ± 0.09	**0.52 ± 0.05**	1.47 ± 0.19	1.31 ± 0.10	1.61 ± 0.05	**0.05 ± 0.00**	**0.05 ± 0.00**	**0.75 ± 0.08**	**0.65 ± 0.10**

Table H: Per-task safety-performance comparison on all 38 re-curated DSRL tasks ($N_{\mathcal{D}_S} = N_{\mathcal{D}_P} = 10$). Results are averaged over five seeds, with standard errors reported. The last two columns report DexDICE with the fixed default $\kappa = 0.01$ and with the per-task best κ selected from the sweep $\kappa \in \{0.001, 0.005, 0.01, 0.05, 0.1, 0.2\}$ (highest return among settings satisfying the cost constraint; the highest-return setting if none is feasible). **Bold** marks methods satisfying the cost constraint (mean cost ≤ 1, normalized to the cost upper bound). Among safety-feasible results, we highlight in **blue** those whose error bars overlap with the highest mean episode return; the per-task best-κ column (right of the vertical rule) is excluded from this comparison and is only marked for constraint satisfaction. This table is the per-task complement of the aggregated results in Table 1.

Environment	Metric	COptiDICE	CDT	TraC	FISOR	FISOR-IL	SafeDICE	ICSDICE	DexDICE
Training Inputs		$r(s,a), c(s,a), \mathcal{D}_\text{U}$	$r(s,a), c(s,a), \mathcal{D}_\text{U}$	$r(s,a), c(s,a), \mathcal{D}_\text{U}$	$r(s,a), c(s,a), \mathcal{D}_\text{U}$	$\tilde{c}(s,a), \mathcal{D}_\text{U}$	$\mathcal{D}_\text{N}, \mathcal{D}_\text{P}$	$r(s,a), \mathcal{D}_\text{S}, \mathcal{D}_\text{U}$	$\mathcal{D}_\text{P}, \mathcal{D}_\text{S}, \mathcal{D}_\text{U}$
PointButton1	Episode return	**0.13 ± 0.01**	0.59 ± 0.01	0.14 ± 0.03	**0.06 ± 0.01**	**0.03 ± 0.01**	0.11 ± 0.01	**0.09 ± 0.01**	**0.15 ± 0.02**
	Episode cost	**0.92 ± 0.12**	2.77 ± 0.31	1.08 ± 0.19	**0.42 ± 0.03**	**0.41 ± 0.04**	1.04 ± 0.14	**0.69 ± 0.08**	**0.95 ± 0.09**
PointButton2	Episode return	**0.22 ± 0.02**	0.55 ± 0.03	0.20 ± 0.04	**0.10 ± 0.01**	**0.09 ± 0.01**	**0.20 ± 0.01**	**0.25 ± 0.02**	**0.27 ± 0.02**
	Episode cost	**0.71 ± 0.04**	1.72 ± 0.10	1.11 ± 0.42	**0.34 ± 0.05**	**0.32 ± 0.04**	**0.64 ± 0.04**	**0.64 ± 0.07**	**0.92 ± 0.08**
PointCircle1	Episode return	0.74 ± 0.03	**0.57 ± 0.01**	**0.52 ± 0.01**	0.40 ± 0.03	0.46 ± 0.01	0.62 ± 0.03	0.67 ± 0.02	**0.27 ± 0.03**
	Episode cost	4.57 ± 0.28	**0.75 ± 0.05**	**0.07 ± 0.03**	3.94 ± 0.31	1.41 ± 0.21	4.02 ± 0.29	3.66 ± 0.12	**0.27 ± 0.07**
PointCircle2	Episode return	0.72 ± 0.02	**0.64 ± 0.01**	0.64 ± 0.01	0.74 ± 0.01	0.71 ± 0.01	0.72 ± 0.02	0.64 ± 0.02	**0.60 ± 0.02**
	Episode cost	2.31 ± 0.15	**0.93 ± 0.03**	1.19 ± 0.06	2.23 ± 0.13	2.05 ± 0.12	2.48 ± 0.08	1.97 ± 0.17	**0.93 ± 0.19**
PointGoal1	Episode return	**0.66 ± 0.01**	0.74 ± 0.01	**0.35 ± 0.08**	0.55 ± 0.01	**0.26 ± 0.01**	**0.65 ± 0.01**	**0.53 ± 0.01**	**0.41 ± 0.02**
	Episode cost	**0.88 ± 0.04**	1.23 ± 0.08	**0.27 ± 0.16**	1.06 ± 0.07	**0.32 ± 0.01**	**0.97 ± 0.05**	**0.78 ± 0.03**	**0.44 ± 0.05**
PointGoal2	Episode return	**0.51 ± 0.02**	0.73 ± 0.02	**0.39 ± 0.05**	**0.17 ± 0.02**	**0.24 ± 0.03**	0.50 ± 0.00	**0.38 ± 0.02**	**0.38 ± 0.02**
	Episode cost	**0.90 ± 0.03**	1.55 ± 0.11	**0.47 ± 0.07**	**0.16 ± 0.05**	**0.23 ± 0.03**	1.01 ± 0.03	**0.75 ± 0.05**	**0.68 ± 0.02**
PointPush1	Episode return	**0.23 ± 0.00**	0.29 ± 0.02	**0.17 ± 0.05**	**0.06 ± 0.01**	**0.16 ± 0.01**	**0.18 ± 0.01**	**0.18 ± 0.01**	**0.19 ± 0.01**
	Episode cost	**0.72 ± 0.06**	1.15 ± 0.18	**0.54 ± 0.20**	**0.09 ± 0.02**	**0.35 ± 0.03**	**0.56 ± 0.04**	**0.51 ± 0.04**	**0.57 ± 0.08**
PointPush2	Episode return	**0.13 ± 0.01**	0.23 ± 0.03	0.13 ± 0.02	**0.09 ± 0.01**	**0.13 ± 0.02**	**0.14 ± 0.01**	**0.16 ± 0.01**	**0.15 ± 0.03**
	Episode cost	**0.85 ± 0.04**	1.28 ± 0.22	1.32 ± 0.38	**0.49 ± 0.05**	**0.74 ± 0.05**	**0.91 ± 0.06**	**0.94 ± 0.03**	**0.96 ± 0.10**
CarButton1	Episode return	**−0.07 ± 0.02**	0.24 ± 0.03	**0.11 ± 0.03**	**−0.16 ± 0.07**	**−0.15 ± 0.05**	**−0.05 ± 0.03**	**0.06 ± 0.02**	**0.13 ± 0.02**
	Episode cost	**0.53 ± 0.04**	1.83 ± 0.15	**0.78 ± 0.17**	**0.02 ± 0.00**	**0.05 ± 0.01**	**0.55 ± 0.07**	**0.59 ± 0.02**	**0.81 ± 0.09**
CarButton2	Episode return	**−0.09 ± 0.02**	0.25 ± 0.04	**−0.11 ± 0.06**	**−0.13 ± 0.04**	**−0.08 ± 0.01**	**−0.16 ± 0.04**	**0.01 ± 0.02**	**0.08 ± 0.02**
	Episode cost	**0.56 ± 0.08**	2.17 ± 0.16	**0.55 ± 0.09**	**0.06 ± 0.01**	**0.10 ± 0.03**	**0.39 ± 0.06**	**0.60 ± 0.05**	**0.86 ± 0.13**
CarCircle1	Episode return	0.71 ± 0.00	0.74 ± 0.02	0.56 ± 0.01	**0.49 ± 0.01**	**0.56 ± 0.01**	0.71 ± 0.01	0.62 ± 0.01	0.56 ± 0.03
	Episode cost	2.43 ± 0.06	1.94 ± 0.12	1.31 ± 0.15	**0.34 ± 0.03**	**0.97 ± 0.04**	2.33 ± 0.06	2.31 ± 0.13	1.37 ± 0.17
CarCircle2	Episode return	0.73 ± 0.01	0.68 ± 0.01	0.47 ± 0.05	**0.43 ± 0.01**	0.56 ± 0.01	0.71 ± 0.02	0.70 ± 0.01	0.63 ± 0.01
	Episode cost	3.03 ± 0.16	2.04 ± 0.24	1.25 ± 0.13	**0.38 ± 0.04**	1.39 ± 0.07	2.99 ± 0.15	2.74 ± 0.13	2.01 ± 0.13
CarGoal1	Episode return	**0.43 ± 0.01**	0.71 ± 0.01	**0.19 ± 0.08**	**0.12 ± 0.01**	**0.25 ± 0.01**	**0.36 ± 0.02**	**0.34 ± 0.02**	**0.31 ± 0.01**
	Episode cost	**0.60 ± 0.04**	1.44 ± 0.07	**0.23 ± 0.09**	**0.01 ± 0.00**	**0.12 ± 0.02**	**0.46 ± 0.02**	**0.41 ± 0.05**	**0.32 ± 0.02**
CarGoal2	Episode return	**0.25 ± 0.01**	0.65 ± 0.01	**0.21 ± 0.06**	**−0.02 ± 0.01**	**0.06 ± 0.01**	**0.24 ± 0.01**	**0.23 ± 0.01**	**0.26 ± 0.01**
	Episode cost	**0.85 ± 0.05**	2.80 ± 0.30	**0.76 ± 0.21**	**0.04 ± 0.01**	**0.15 ± 0.02**	**0.85 ± 0.05**	**0.83 ± 0.08**	**0.90 ± 0.09**
CarPush1	Episode return	**0.20 ± 0.01**	**0.32 ± 0.02**	**0.12 ± 0.04**	**0.12 ± 0.01**	**0.15 ± 0.00**	**0.19 ± 0.01**	**0.20 ± 0.01**	**0.21 ± 0.00**
	Episode cost	**0.41 ± 0.04**	**0.79 ± 0.04**	**0.24 ± 0.19**	**0.14 ± 0.03**	**0.18 ± 0.03**	**0.35 ± 0.04**	**0.47 ± 0.06**	**0.30 ± 0.02**
CarPush2	Episode return	**0.09 ± 0.01**	0.19 ± 0.01	**0.03 ± 0.03**	**0.00 ± 0.01**	**0.05 ± 0.01**	**0.10 ± 0.01**	**0.10 ± 0.01**	**0.12 ± 0.01**
	Episode cost	**0.99 ± 0.10**	1.96 ± 0.17	**0.74 ± 0.19**	**0.08 ± 0.01**	**0.17 ± 0.03**	**0.97 ± 0.05**	**0.89 ± 0.06**	**0.93 ± 0.10**
SwimmerVelocity	Episode return	0.36 ± 0.03	**0.69 ± 0.00**	**0.58 ± 0.01**	−0.05 ± 0.01	**0.22 ± 0.04**	0.38 ± 0.06	0.44 ± 0.04	**0.62 ± 0.01**
	Episode cost	2.95 ± 0.49	**0.88 ± 0.00**	**0.01 ± 0.01**	1.34 ± 0.73	**0.16 ± 0.09**	2.80 ± 0.80	3.26 ± 0.46	**0.65 ± 0.28**
HopperVelocity	Episode return	0.25 ± 0.12	0.72 ± 0.25	**0.63 ± 0.07**	**0.13 ± 0.01**	**0.47 ± 0.03**	**0.26 ± 0.09**	0.15 ± 0.02	**0.47 ± 0.07**
	Episode cost	1.16 ± 0.45	1.11 ± 0.10	**0.70 ± 0.36**	**0.05 ± 0.03**	**0.85 ± 0.09**	**0.98 ± 0.31**	2.55 ± 0.33	**0.60 ± 0.12**
HalfCheetahVelocity	Episode return	**0.83 ± 0.04**	**0.98 ± 0.01**	0.97 ± 0.01	**0.87 ± 0.00**	**0.76 ± 0.02**	**0.77 ± 0.08**	0.94 ± 0.01	**0.76 ± 0.01**
	Episode cost	**0.78 ± 0.20**	**0.38 ± 0.15**	2.93 ± 1.36	**0.00 ± 0.00**	**0.00 ± 0.00**	**0.82 ± 0.12**	4.80 ± 0.69	**0.45 ± 0.12**
Walker2dVelocity	Episode return	**0.78 ± 0.00**	**0.68 ± 0.07**	**0.79 ± 0.00**	**0.34 ± 0.05**	**0.75 ± 0.01**	**0.77 ± 0.01**	**0.77 ± 0.01**	**0.79 ± 0.01**
	Episode cost	**0.69 ± 0.30**	**0.75 ± 0.24**	**0.00 ± 0.00**	**0.35 ± 0.05**	**0.33 ± 0.10**	**0.40 ± 0.21**	**0.57 ± 0.22**	**0.07 ± 0.02**
AntVelocity	Episode return	0.98 ± 0.00	0.99 ± 0.01	**0.98 ± 0.00**	**0.87 ± 0.00**	**0.93 ± 0.00**	0.98 ± 0.00	0.98 ± 0.01	**0.96 ± 0.00**
	Episode cost	3.93 ± 0.35	1.11 ± 0.08	**0.17 ± 0.03**	**0.00 ± 0.00**	**0.03 ± 0.01**	3.53 ± 0.66	1.91 ± 0.35	**0.15 ± 0.03**
BallRun	Episode return	0.65 ± 0.02	**0.53 ± 0.01**	**0.31 ± 0.02**	**0.07 ± 0.07**	**0.08 ± 0.02**	0.54 ± 0.02	1.11 ± 0.03	**0.41 ± 0.04**
	Episode cost	1.40 ± 0.04	**0.99 ± 0.01**	**0.39 ± 0.08**	**0.13 ± 0.09**	**0.08 ± 0.03**	1.17 ± 0.05	2.18 ± 0.03	**0.86 ± 0.19**
CarRun	Episode return	**0.97 ± 0.00**	**0.99 ± 0.00**	**0.97 ± 0.00**	**0.54 ± 0.11**	**−0.07 ± 0.09**	**0.97 ± 0.00**	**0.96 ± 0.00**	**0.97 ± 0.01**
	Episode cost	**0.02 ± 0.01**	**0.92 ± 0.04**	**0.00 ± 0.00**	**0.19 ± 0.12**	**0.02 ± 0.01**	**0.01 ± 0.01**	**0.09 ± 0.04**	**0.79 ± 0.30**
DroneRun	Episode return	0.67 ± 0.01	0.64 ± 0.02	**0.55 ± 0.01**	**0.27 ± 0.03**	**0.34 ± 0.04**	0.58 ± 0.04	0.64 ± 0.02	**0.60 ± 0.01**
	Episode cost	1.82 ± 0.07	1.12 ± 0.16	**0.00 ± 0.00**	**0.22 ± 0.06**	**0.62 ± 0.09**	1.49 ± 0.12	1.13 ± 0.34	**0.68 ± 0.20**
AntRun	Episode return	0.75 ± 0.01	**0.71 ± 0.02**	**0.63 ± 0.04**	**0.33 ± 0.01**	**0.68 ± 0.01**	0.72 ± 0.01	0.86 ± 0.01	**0.55 ± 0.03**
	Episode cost	1.88 ± 0.04	**0.93 ± 0.10**	**0.45 ± 0.03**	**0.02 ± 0.00**	**0.38 ± 0.05**	1.20 ± 0.11	2.99 ± 0.10	**0.46 ± 0.13**
BallCircle	Episode return	0.76 ± 0.01	**0.85 ± 0.00**	**0.74 ± 0.02**	**0.26 ± 0.01**	**0.16 ± 0.01**	**0.73 ± 0.02**	**0.76 ± 0.01**	**0.65 ± 0.02**
	Episode cost	1.00 ± 0.05	**0.99 ± 0.02**	**0.63 ± 0.05**	**0.00 ± 0.00**	**0.00 ± 0.00**	**0.94 ± 0.05**	**0.96 ± 0.04**	**0.54 ± 0.05**
CarCircle	Episode return	0.65 ± 0.02	0.82 ± 0.01	0.69 ± 0.05	**0.21 ± 0.02**	**0.02 ± 0.01**	0.67 ± 0.01	0.85 ± 0.01	**0.70 ± 0.01**
	Episode cost	1.57 ± 0.06	1.22 ± 0.06	1.26 ± 0.45	**0.02 ± 0.01**	**0.03 ± 0.02**	1.40 ± 0.04	2.02 ± 0.07	**0.82 ± 0.03**
DroneCircle	Episode return	0.70 ± 0.01	0.68 ± 0.00	**0.62 ± 0.03**	**0.35 ± 0.01**	**0.40 ± 0.01**	0.69 ± 0.02	0.68 ± 0.01	**0.54 ± 0.01**
	Episode cost	1.06 ± 0.04	1.00 ± 0.02	**0.86 ± 0.06**	**0.00 ± 0.00**	**0.05 ± 0.01**	1.09 ± 0.08	1.07 ± 0.05	**0.80 ± 0.04**
AntCircle	Episode return	0.60 ± 0.02	0.60 ± 0.02	**0.41 ± 0.02**	**0.11 ± 0.01**	**0.11 ± 0.01**	0.60 ± 0.02	0.49 ± 0.01	**0.35 ± 0.01**
	Episode cost	2.05 ± 0.08	1.94 ± 0.17	**0.53 ± 0.13**	**0.00 ± 0.00**	**0.00 ± 0.00**	2.05 ± 0.12	1.33 ± 0.06	**0.48 ± 0.06**
MetaDrive-easysparse	Episode return	**0.65 ± 0.09**	**0.99 ± 0.01**	**0.18 ± 0.15**	**0.35 ± 0.02**	**0.51 ± 0.01**	**0.44 ± 0.07**	0.88 ± 0.06	**0.56 ± 0.09**
	Episode cost	**0.94 ± 0.26**	**0.02 ± 0.01**	**0.03 ± 0.00**	**0.09 ± 0.04**	**0.29 ± 0.05**	**0.43 ± 0.13**	1.77 ± 0.22	**0.40 ± 0.20**
MetaDrive-easymean	Episode return	**0.53 ± 0.10**	**0.72 ± 0.05**	**0.19 ± 0.06**	**0.34 ± 0.02**	**0.47 ± 0.02**	0.71 ± 0.08	0.91 ± 0.02	**0.73 ± 0.01**
	Episode cost	**0.69 ± 0.24**	**0.05 ± 0.01**	**0.03 ± 0.00**	**0.06 ± 0.01**	**0.44 ± 0.09**	1.09 ± 0.18	1.80 ± 0.07	**0.51 ± 0.03**
MetaDrive-easydense	Episode return	**0.51 ± 0.05**	**0.48 ± 0.03**	**0.39 ± 0.23**	**0.39 ± 0.02**	**0.50 ± 0.03**	**0.60 ± 0.05**	**0.56 ± 0.08**	**0.56 ± 0.04**
	Episode cost	**0.69 ± 0.16**	**0.04 ± 0.00**	**0.02 ± 0.00**	**0.04 ± 0.01**	**0.41 ± 0.14**	**0.79 ± 0.16**	**0.43 ± 0.14**	**0.25 ± 0.04**
MetaDrive-mediumsparse	Episode return	0.82 ± 0.04	**0.21 ± 0.06**	**0.32 ± 0.16**	**0.43 ± 0.03**	**0.65 ± 0.06**	**0.62 ± 0.06**	0.78 ± 0.04	**0.74 ± 0.09**
	Episode cost	1.11 ± 0.17	**0.05 ± 0.00**	**0.05 ± 0.00**	**0.01 ± 0.00**	**0.54 ± 0.16**	**0.68 ± 0.20**	2.02 ± 0.14	**0.88 ± 0.15**
MetaDrive-mediummean	Episode return	0.78 ± 0.05	**0.23 ± 0.10**	**0.66 ± 0.18**	**0.28 ± 0.01**	**0.71 ± 0.05**	0.78 ± 0.04	0.82 ± 0.02	**0.51 ± 0.14**
	Episode cost	1.68 ± 0.11	**0.05 ± 0.00**	**0.04 ± 0.01**	**0.00 ± 0.00**	**0.62 ± 0.08**	1.43 ± 0.23	2.11 ± 0.03	**0.82 ± 0.14**
MetaDrive-mediumdense	Episode return	0.81 ± 0.04	**0.40 ± 0.30**	**0.77 ± 0.06**	**0.36 ± 0.04**	**0.87 ± 0.01**	0.78 ± 0.04	0.81 ± 0.08	**0.53 ± 0.08**
	Episode cost	1.45 ± 0.11	**0.04 ± 0.02**	**0.04 ± 0.01**	**0.02 ± 0.01**	**0.96 ± 0.17**	1.47 ± 0.24	1.14 ± 0.24	**0.73 ± 0.14**
MetaDrive-hardsparse	Episode return	0.47 ± 0.03	**0.45 ± 0.08**	**0.15 ± 0.05**	**0.31 ± 0.01**	**0.38 ± 0.01**	0.49 ± 0.02	0.48 ± 0.03	**0.35 ± 0.02**
	Episode cost	1.85 ± 0.12	**0.03 ± 0.01**	**0.05 ± 0.00**	**0.01 ± 0.01**	**0.65 ± 0.07**	1.95 ± 0.04	1.97 ± 0.09	**0.13 ± 0.03**
MetaDrive-hardmean	Episode return	0.49 ± 0.02	**0.34 ± 0.06**	**0.13 ± 0.07**	**0.25 ± 0.01**	**0.41 ± 0.02**	0.46 ± 0.03	0.41 ± 0.03	**0.47 ± 0.01**
	Episode cost	1.90 ± 0.12	**0.05 ± 0.01**	**0.05 ± 0.00**	**0.01 ± 0.01**	**0.91 ± 0.14**	1.67 ± 0.23	1.99 ± 0.10	**0.85 ± 0.05**
MetaDrive-harddense	Episode return	0.33 ± 0.03	**0.19 ± 0.01**	**0.28 ± 0.07**	**0.31 ± 0.02**	**0.36 ± 0.02**	0.38 ± 0.02	0.38 ± 0.04	**0.36 ± 0.03**
	Episode cost	1.30 ± 0.10	**0.06 ± 0.00**	**0.05 ± 0.00**	**0.05 ± 0.03**	**0.54 ± 0.10**	1.26 ± 0.20	1.60 ± 0.13	**0.65 ± 0.10**

Table I: Performance comparison with related-but-distinct offline settings (COptiDICE, SafeDICE, ICSDICE) and offline safe RL baselines (CDT, TraC, FISOR, FISOR-IL) on all 38 re-curated DSRL tasks. DexDICE is reported with its post hoc per-task best κ selected using evaluation return and cost, identical to the last column of Table H; this differs from the fixed-κ main result. Results are averaged over five seeds, with standard errors reported. **Bold** marks methods satisfying the cost constraint (mean cost ≤ 1, normalized to the cost upper bound). Among safety-feasible results, we highlight in **blue** those whose error bars overlap with the highest mean episode return.

G.4 Hyperparameters

Hyperparameters	BC	FBC	JFBC	DemoDICE-S	DemoDICE-U	DWBC-S	DexDICE (Ours)
Policy distribution	Gaussian						
Batch size	512						
Discount factor γ	0.99						
Network sizes (π, ν, C_{P}, C_{S})	$[256, 256]$ (Safe Navigation/Velocity, re-curated DSRL tasks) $[1024, 1024, 512]$ (Safe Manipulation tasks)						
Training iteration	1,000,000						
Evaluation frequency	10,000						
Algorithm specific configurations							
π learning rate	3×10^{-5}						
α (regularization coefficient)	-	-	-	0.1	0.1	7.5	-
η (class prior)	-	-	-	-	-	0.5	-
κ update frequency	-	100	100	-	-	-	100
ν learning rate	-	-	-	3×10^{-5}	3×10^{-5}	-	3×10^{-5}
ν gradient penalty coefficient	-	-	-	10	10	-	10
C_{P} gradient penalty coefficient	-	-	0	10	10	-	10
C_{P} learning rate	-	-	3×10^{-5}	3×10^{-5}	3×10^{-5}	3×10^{-5}	3×10^{-5}
C_{S} gradient penalty coefficient	-	0	0	-	-	-	0
C_{S} learning rate	-	3×10^{-5}	3×10^{-5}	-	-	-	3×10^{-5}

Table J: Summary of hyperparameters used for each method. A fixed value of $\kappa = 0.01$ is used for all tasks in both the re-curated DSRL and the controlled datasets without further tuning; the per-task best κ from the sweep $\kappa \in \{0.001, 0.005, 0.01, 0.05, 0.1, 0.2\}$ is reported only in the last column of Table H and in Table I.

G.5 Data Efficiency Analysis

The main paper's Figure 2 reports six-task averages of data efficiency and quality robustness with respect to both the safe demonstration set $\mathcal{D}_{\mathrm{S}}$ and the performant demonstration set $\mathcal{D}_{\mathrm{P}}$. The remainder of this section (data efficiency) and Section G.6 (data quality) expand these aggregated curves into per-task breakdowns; Section G.5.1 additionally reports a joint $\mathcal{D}_{\mathrm{S}}$/$\mathcal{D}_{\mathrm{P}}$ scaling analysis.

G.5.1 Data Efficiency with Joint Scaling

In this section, we analyze data efficiency by varying the dataset size parameter k, where $N_{\mathcal{D}_{\mathrm{S}}} = N_{\mathcal{D}_{\mathrm{P}}} = 2k$. Specifically, we vary the number of demonstrations as $k \in \{1, 3, 5, 7, 10\}$ while keeping $N_{\mathcal{D}_{\mathrm{U}}} = 2000$. Figure B presents the aggregated results across six benchmark tasks, providing a high-level comparison among the methods. Figure C reports the per-task results, illustrating how data efficiency differs across individual environments.

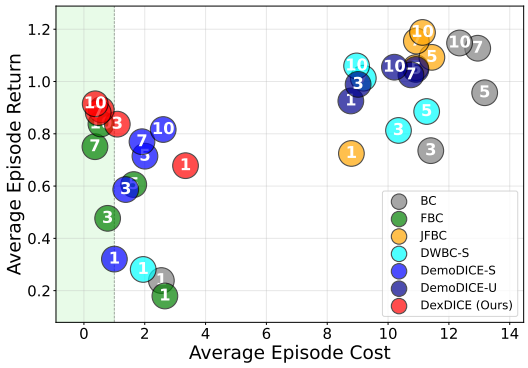


Figure B: Overall data efficiency averaged over six tasks of each method by jointly varying k, where $N_{\mathcal{D}_{\mathrm{P}}} = N_{\mathcal{D}_{\mathrm{S}}} = 2k$.

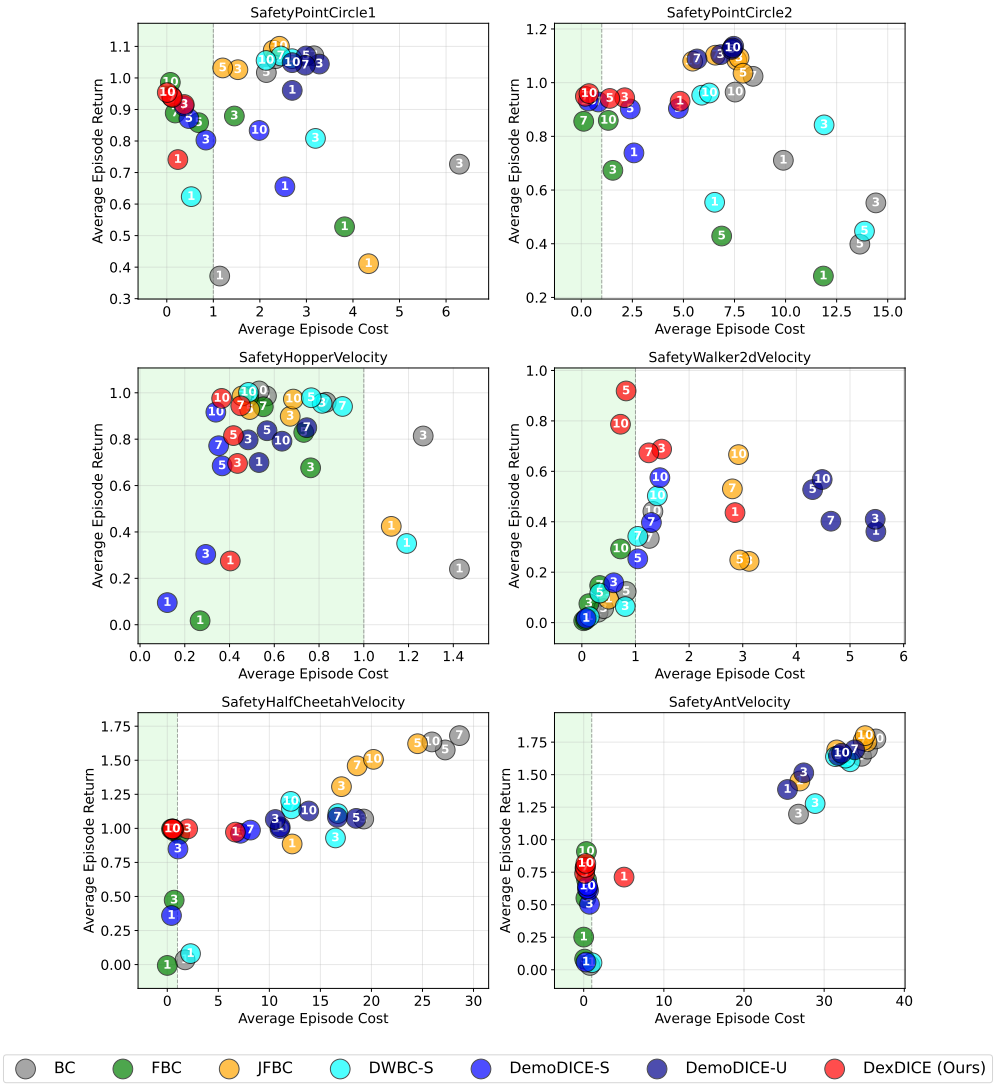


Figure C: Per-task analysis on data efficiency with jointly varying k where $N_{\mathcal{D}_\mathrm{P}} = N_{\mathcal{D}_\mathrm{S}} = 2k$.

G.5.2 Data Efficiency on $\mathcal{D}_{\mathrm{S}}$

We sweep $N_{\mathcal{D}_{\mathrm{S}}}$ while keeping $N_{\mathcal{D}_{\mathrm{P}}} = 10$ and $N_{\mathcal{D}_{\mathrm{U}}} = 2000$ fixed. Figure 2a reports the average across six tasks; Figure D provides the per-task breakdown.

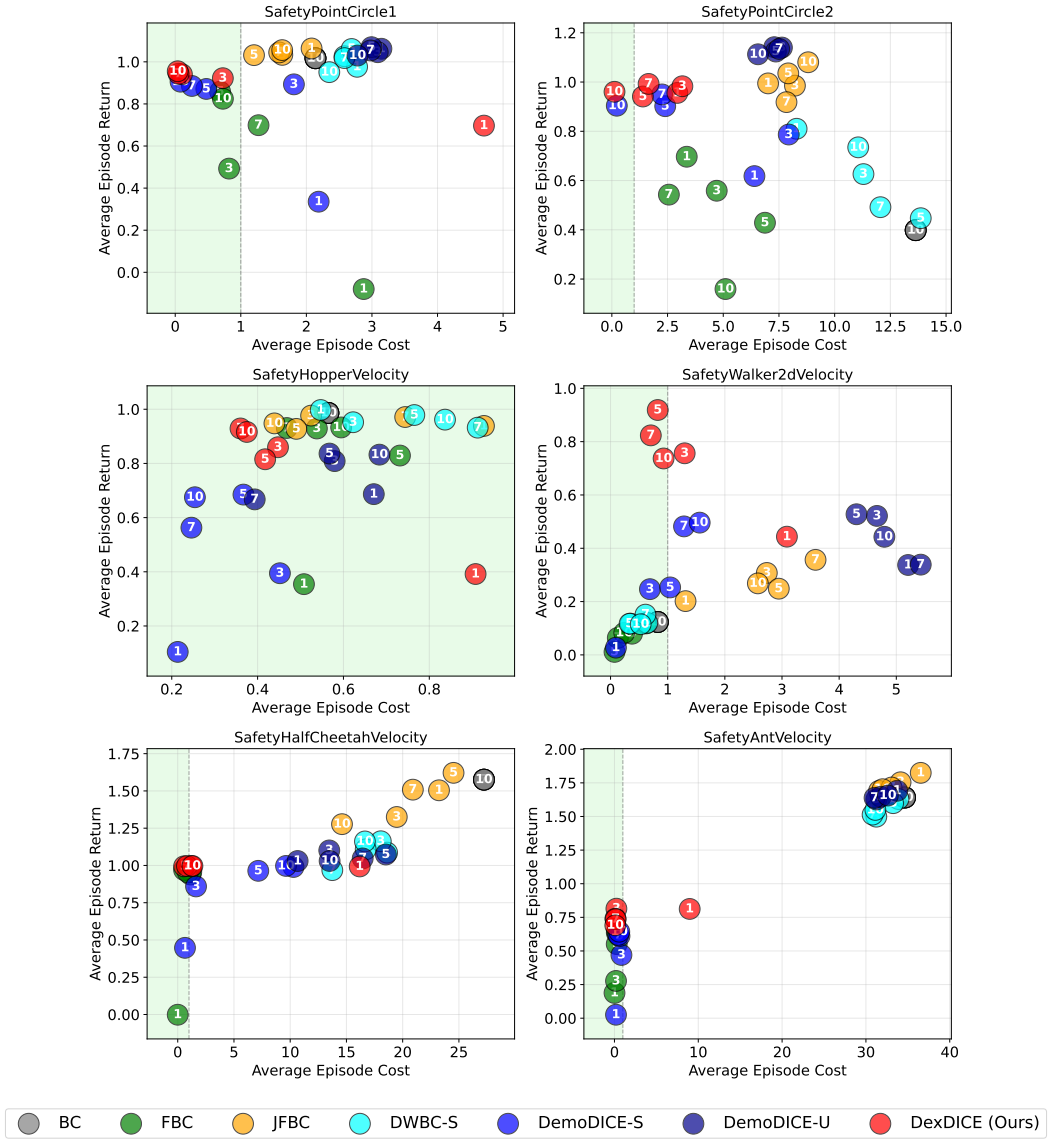


Figure D: Per-task analysis on data efficiency of safe demonstrations $\mathcal{D}_{\mathrm{S}}$.

G.5.3 Data Efficiency on $\mathcal{D}_{\mathrm{P}}$

We sweep $N_{\mathcal{D}_{\mathrm{P}}}$ while keeping $N_{\mathcal{D}_{\mathrm{S}}} = 10$ and $N_{\mathcal{D}_{\mathrm{U}}} = 2000$ fixed. Figure 2c reports the average across six tasks; Figure E provides the per-task breakdown. Trends are consistent with those reported for DemoDICE-style methods: DexDICE and DemoDICE-U scale gracefully with $N_{\mathcal{D}_{\mathrm{P}}}$, while filtering-based baselines (FBC, JFBC) saturate or regress at small $\mathcal{D}_{\mathrm{P}}$.

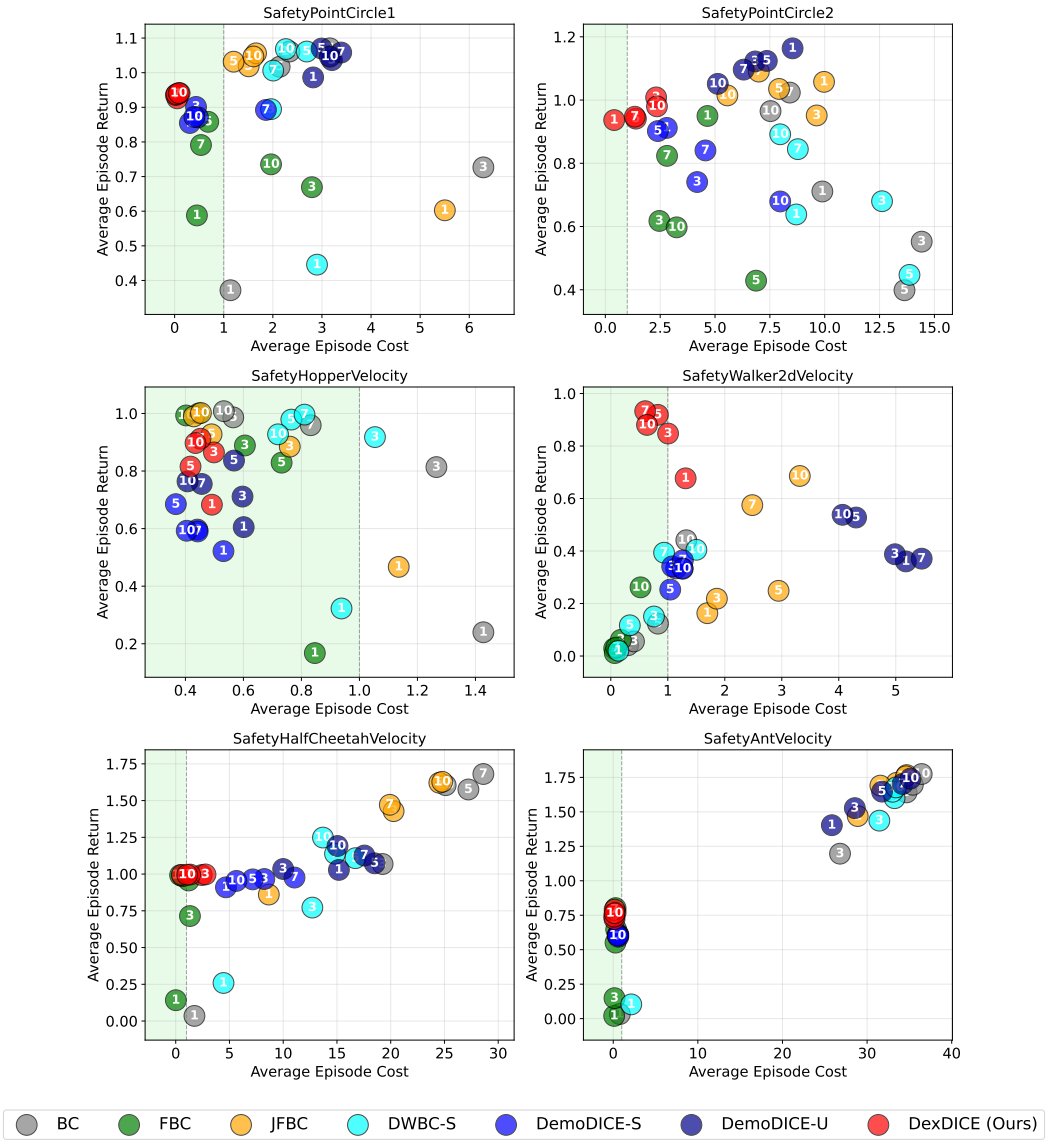


Figure E: Per-task analysis on data efficiency of performant demonstrations $\mathcal{D}_{\mathrm{P}}$.

G.6 Data Quality Analysis

This section provides per-task breakdowns of the six-task data-quality averages reported in Section 5.2 (Figures 2b and 2d). All sweeps fix $N_{\mathcal{D}_\mathrm{P}} = N_{\mathcal{D}_\mathrm{S}} = 20$ and $N_{\mathcal{D}_\mathrm{U}} = 2000$.

G.6.1 Data Quality on $\mathcal{D}_\mathrm{S}$

We vary **quality**$(\mathcal{D}_\mathrm{S}) \in \{5\%, 25\%, 50\%, 75\%, 95\%\}$ while holding **quality**$(\mathcal{D}_\mathrm{P}) = 50\%$ fixed. Figure 2b shows the average across six tasks; Figure F provides the per-task breakdown.

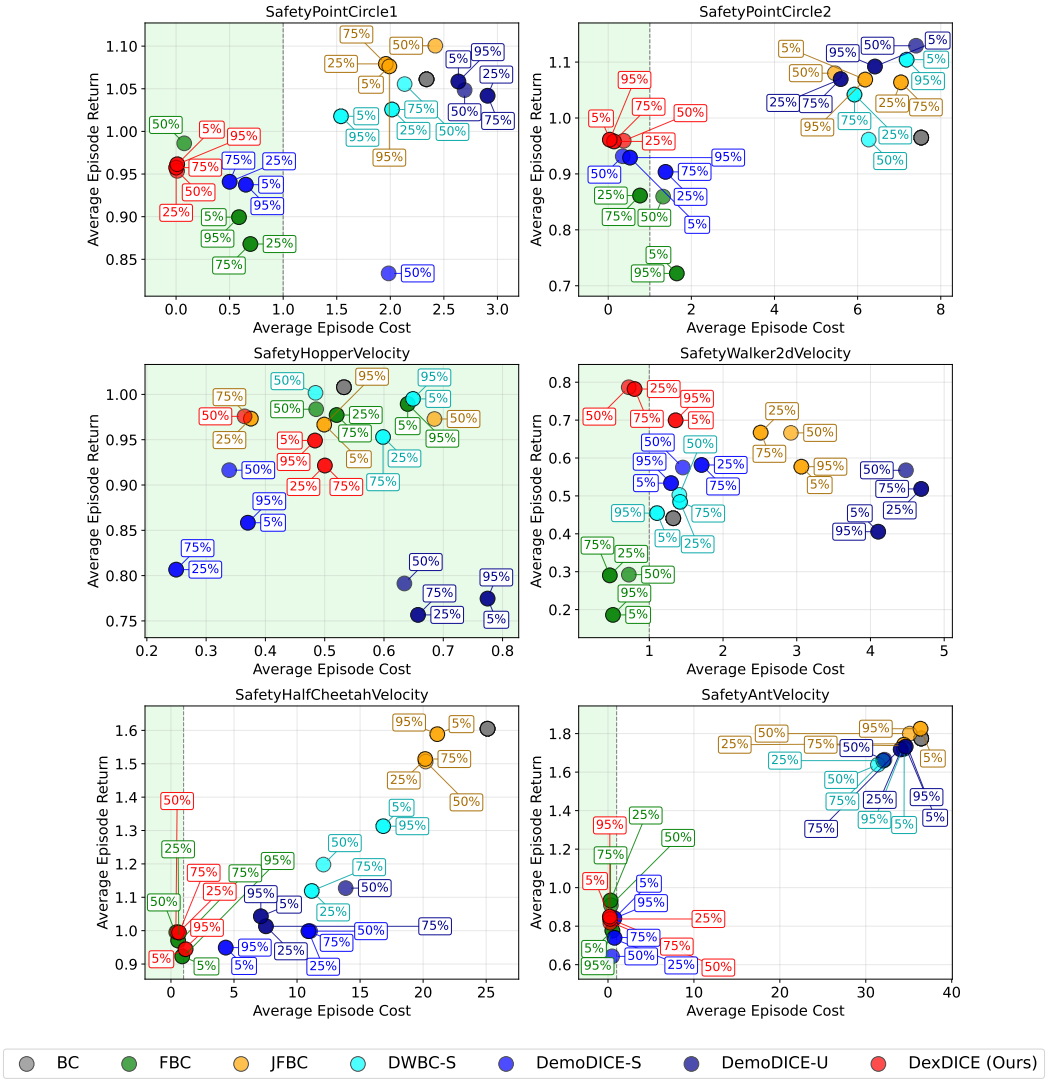


Figure F: Per-task performance for dependencies on **quality**$(\mathcal{D}_\mathrm{S})$.

G.6.2 Data Quality on $\mathcal{D}_{\mathrm{P}}$

We vary **quality**$(\mathcal{D}_{\mathrm{P}}) \in \{5\%, 25\%, 50\%, 75\%, 95\%\}$ while holding **quality**$(\mathcal{D}_{\mathrm{S}}) = 50\%$ fixed. Figure 2d shows the average across six tasks; Figure G provides the per-task breakdown. DexDICE and DemoDICE-U track each other closely, with DexDICE additionally maintaining the cost constraint across all configurations except the most extreme low-quality regime.

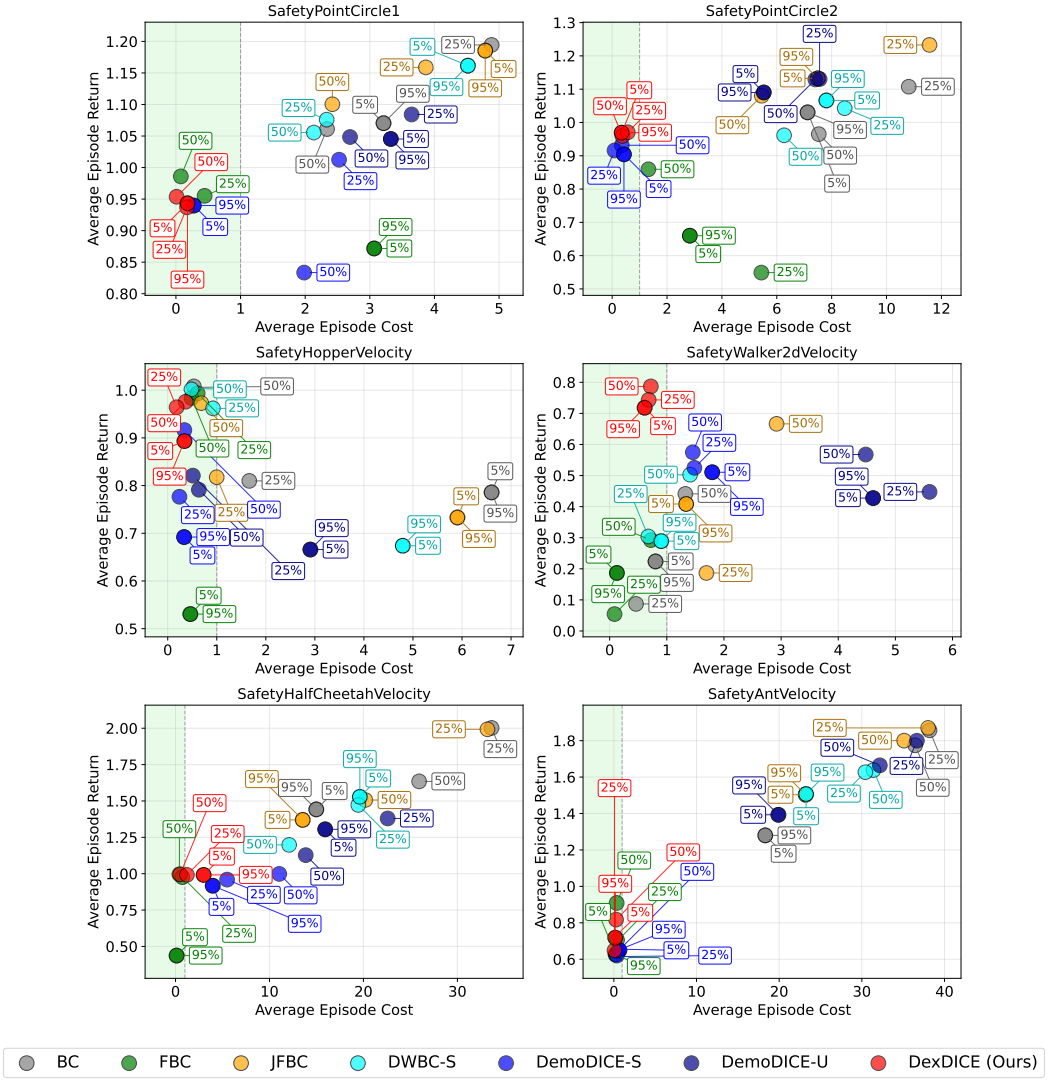


Figure G: Per-task performance analysis for dependencies on **quality**$(\mathcal{D}_{\mathrm{P}})$.

G.7 Comparison on Support Estimators

One of our core components is the support estimator constructed from unlabeled demonstrations proposed in Section 4.2. To assess effectiveness of decision functions, we compare our approach against the following established baselines:

- **GMM**: a density-estimation approach based on fitting Gaussian mixture models [6] to approximate d_{S}.
- **LOF (Local Outlier Factor)** [8]: a non-parametric method that detects outliers by measuring the local deviation of a sample's density relative to that of its neighbors.
- **OCSVM (One-class Support Vector Machine)** [55]: a kernel-based method that learns a hypersurface enclosing the majority of data, treating points outside the boundary as novelties.
- **Deep SVDD (Deep Support Vector Data Description)** [54]: a deep representation–based method that maps data into a latent space and minimizes the volume of a hypersphere enclosing the nominal samples.

Task	GMM	LOF	OCSVM	Deep SVDD	Ours
PointCircle1	0.62 ± 0.01	0.46 ± 0.00	0.43 ± 0.00	0.56 ± 0.00	**0.87 ± 0.00**
PointCircle2	0.71 ± 0.00	0.60 ± 0.00	0.52 ± 0.00	0.67 ± 0.00	**0.80 ± 0.00**
HopperVelocity	**0.83 ± 0.00**	0.73 ± 0.00	0.65 ± 0.00	0.76 ± 0.01	0.74 ± 0.02
Walker2dVelocity	0.94 ± 0.00	0.88 ± 0.00	0.82 ± 0.00	0.85 ± 0.00	**0.95 ± 0.00**
HalfCheetahVelocity	0.56 ± 0.00	0.58 ± 0.00	0.49 ± 0.00	0.45 ± 0.00	**0.65 ± 0.00**
CarCircle	0.50 ± 0.00	0.47 ± 0.00	0.49 ± 0.00	0.49 ± 0.00	**0.72 ± 0.00**
CarRun	0.54 ± 0.01	0.35 ± 0.00	0.48 ± 0.00	0.45 ± 0.02	**0.64 ± 0.00**
DroneCircle	0.67 ± 0.00	0.49 ± 0.00	0.64 ± 0.00	0.56 ± 0.01	**0.90 ± 0.00**
DroneRun	0.77 ± 0.00	0.81 ± 0.00	0.35 ± 0.00	**0.88 ± 0.00**	0.83 ± 0.00
AntCircle	0.55 ± 0.01	0.56 ± 0.00	0.46 ± 0.00	0.54 ± 0.00	**0.69 ± 0.00**
AntRun	0.62 ± 0.00	0.75 ± 0.00	0.69 ± 0.00	0.64 ± 0.01	**0.76 ± 0.01**

Table K: AUROC comparison of different novelty detection methods on re-curated DSRL tasks.

As shown in Table K, our method achieves the highest AUROC on 8 of 11 tasks, indicating consistently superior novelty-detection performance. This highlights the benefit of leveraging density-ratio estimation for reliable support estimation. We further examine how the choice of support estimator impacts safety and performance on DexDICE; the results are reported in Table L.

Task	Metric	GMM	LOF	OCSVM	Deep SVDD	Ours
PointCircle1	Episode return	0.64 ± 0.03	0.80 ± 0.01	0.79 ± 0.01	0.87 ± 0.01	**0.27 ± 0.03**
	Episode cost	2.81 ± 0.19	4.76 ± 0.10	4.81 ± 0.12	4.64 ± 0.09	**0.27 ± 0.07**
PointCircle2	Episode return	0.68 ± 0.01	**0.03 ± 0.12**	0.79 ± 0.01	0.73 ± 0.03	**0.60 ± 0.02**
	Episode cost	2.56 ± 0.07	**0.65 ± 0.37**	3.10 ± 0.06	2.67 ± 0.19	**0.93 ± 0.19**
HopperVelocity	Episode return	**0.13 ± 0.06**	**0.10 ± 0.05**	**0.12 ± 0.04**	0.20 ± 0.06	**0.47 ± 0.07**
	Episode cost	**0.17 ± 0.04**	**0.00 ± 0.00**	**0.63 ± 0.20**	2.10 ± 0.60	**0.60 ± 0.12**
HalfCheetahVelocity	Episode return	0.94 ± 0.01	0.96 ± 0.01	0.92 ± 0.04	0.76 ± 0.19	**0.76 ± 0.01**
	Episode cost	1.89 ± 0.49	3.09 ± 0.91	1.48 ± 0.32	2.02 ± 0.60	**0.45 ± 0.12**
Walker2dVelocity	Episode return	**0.80 ± 0.01**	0.88 ± 0.01	0.79 ± 0.01	0.81 ± 0.01	**0.79 ± 0.01**
	Episode cost	**0.52 ± 0.10**	3.38 ± 0.16	1.33 ± 0.32	1.72 ± 0.44	**0.07 ± 0.02**
CarRun	Episode return	**0.97 ± 0.00**	**0.97 ± 0.00**	**0.98 ± 0.00**	**0.97 ± 0.00**	**0.97 ± 0.01**
	Episode cost	**0.37 ± 0.22**	**0.18 ± 0.07**	**0.41 ± 0.27**	**0.15 ± 0.06**	**0.79 ± 0.30**
CarCircle	Episode return	0.58 ± 0.05	0.83 ± 0.01	0.82 ± 0.01	0.87 ± 0.01	**0.70 ± 0.01**
	Episode cost	1.20 ± 0.09	2.00 ± 0.03	1.92 ± 0.05	2.03 ± 0.02	**0.82 ± 0.03**
DroneRun	Episode return	**0.58 ± 0.02**	0.74 ± 0.02	0.75 ± 0.01	0.74 ± 0.01	**0.60 ± 0.01**
	Episode cost	**0.76 ± 0.19**	2.28 ± 0.03	2.31 ± 0.08	2.26 ± 0.07	**0.68 ± 0.20**
DroneCircle	Episode return	**0.54 ± 0.02**	0.69 ± 0.01	0.76 ± 0.01	0.77 ± 0.01	**0.54 ± 0.01**
	Episode cost	**0.89 ± 0.03**	1.62 ± 0.06	1.53 ± 0.06	1.53 ± 0.05	**0.80 ± 0.04**
AntRun	Episode return	**0.60 ± 0.01**	0.75 ± 0.01	0.78 ± 0.01	0.76 ± 0.02	**0.55 ± 0.03**
	Episode cost	**0.23 ± 0.05**	1.81 ± 0.09	2.03 ± 0.14	1.77 ± 0.20	**0.46 ± 0.13**
AntCircle	Episode return	**0.32 ± 0.01**	0.51 ± 0.01	0.58 ± 0.01	0.58 ± 0.01	**0.35 ± 0.01**
	Episode cost	**0.76 ± 0.08**	1.14 ± 0.06	1.82 ± 0.05	1.80 ± 0.09	**0.48 ± 0.06**

Table L: Safety-Performance comparison with different support estimators on re-curated DSRL tasks. Results are averaged over five seeds with standard errors. We follow the same evaluation protocol as in Table H.

G.8 Sensitivity Analysis on Target FNR κ

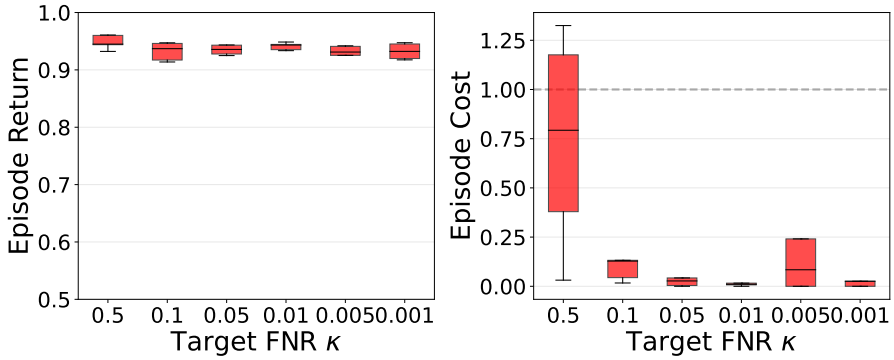


Figure H: Sensitivity analysis of DexDICE with respect to the target FNR κ used as the threshold for $\mathbb{1}_{d_S}$. Experiments are conducted on controlled datasets of the PointCircle1 task with $N_{\mathcal{D}_P} = N_{\mathcal{D}_S} = 10$, over 5 random seeds.

The target FNR κ determines the threshold for the estimated safe support: as κ increases, the quantile threshold τ_κ rises and the set of accepted state-action pairs becomes smaller. Figure H shows that episode return and cost remain stable across much of the tested range, with performance degrading at larger κ values as the estimated safe support narrows.

G.9 Robustness on Noisy Safe Demonstrations

Figure I evaluates DexDICE when a fraction of the safe demonstrations in $\mathcal{D}_S$ is replaced with unsafe ones; DexDICE sustains high return and low cost under moderate corruption, while most baselines degrade sharply.

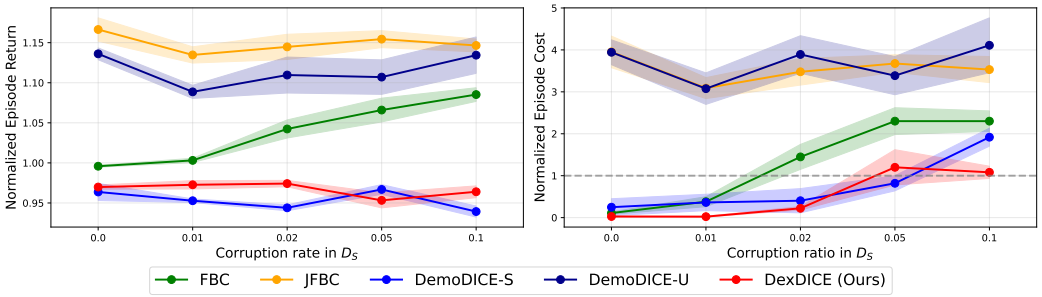


Figure I: Stability under corruption of safe demonstrations. We evaluate on the controlled dataset of the PointCircle1 task, using a sufficient number of demonstrations ($N_{\mathcal{D}_P} = 200$, $N_{\mathcal{D}_S} = 200$). The x-axis indicates the corruption rate of $\mathcal{D}_S$, i.e., the proportion of safe demonstrations replaced with unsafe demonstrations that exceed the episode cost upper bound. Both DexDICE and DemoDICE-S maintain relatively high episode return and low episode cost even under moderate corruption, while other methods degrade significantly as $\mathcal{D}_S$ becomes increasingly corrupted.

G.10 Sensitivity to Dataset Construction Thresholds

The re-curated DSRL benchmark labels trajectories as safe or performant using two thresholds (Appendix G.1): a cost threshold defining $\mathcal{D}_S$ and a return threshold defining $\mathcal{D}_P$. These thresholds determine, respectively, the demonstration-defined safety support and the performance target. We therefore provide a one-at-a-time sensitivity sweep over both, on six tasks with $N_{\mathcal{D}_S} = N_{\mathcal{D}_P} = 10$ and 5 seeds, following the evaluation protocol of Table 1. (i) We vary the cost threshold defining $\mathcal{D}_S$ from $0.5\times$ to $1.5\times$ the DSRL cost limit while holding the $\mathcal{D}_P$ threshold at its default (Table M). (ii) We vary the return threshold defining $\mathcal{D}_P$ over the 50th, 75th, 90th, and 95th percentiles of safe

trajectory returns, plus a threshold above the maximum safe trajectory return, while holding the $\mathcal{D}_{\mathrm{S}}$ threshold at its default (Table N). These thresholds change only the dataset labels; every policy is evaluated against the same fixed normalized cost bar of 1.0 defined by the original DSRL cost limit.

At the default settings ($1.0\times$ cost and 90th-percentile return), all six tasks remain below the cost bar. Across the full sweep, all 30 $\mathcal{D}_{\mathrm{S}}$-threshold cells and 27 of 30 $\mathcal{D}_{\mathrm{P}}$-threshold cells remain below it; the three gray entries occur only at non-default $\mathcal{D}_{\mathrm{P}}$ thresholds. Returns vary by task, but the default thresholds show no systematic degradation relative to neighboring settings.

Cost threshold	Metric	PointCircle1	DroneCircle	Walker2dVelocity	HopperVelocity	AntVelocity	HalfCheetahVelocity
$0.5\times$	Episode return	0.26 ± 0.03	0.41 ± 0.04	0.78 ± 0.00	0.20 ± 0.07	0.93 ± 0.02	0.80 ± 0.02
	Episode cost	0.12 ± 0.10	0.48 ± 0.04	0.02 ± 0.01	0.93 ± 0.39	0.10 ± 0.04	0.01 ± 0.00
$0.75\times$	Episode return	0.46 ± 0.02	0.42 ± 0.04	0.71 ± 0.06	0.14 ± 0.06	0.96 ± 0.01	0.73 ± 0.05
	Episode cost	0.10 ± 0.04	0.77 ± 0.02	0.27 ± 0.22	0.22 ± 0.08	0.17 ± 0.05	0.17 ± 0.16
$1.0\times$ (default)	Episode return	0.27 ± 0.03	0.54 ± 0.01	0.79 ± 0.01	0.47 ± 0.07	0.96 ± 0.00	0.76 ± 0.01
	Episode cost	0.27 ± 0.07	0.80 ± 0.04	0.07 ± 0.02	0.60 ± 0.12	0.15 ± 0.03	0.45 ± 0.12
$1.25\times$	Episode return	0.31 ± 0.02	0.52 ± 0.02	0.79 ± 0.01	0.19 ± 0.02	0.96 ± 0.01	0.87 ± 0.01
	Episode cost	0.18 ± 0.14	0.75 ± 0.06	0.06 ± 0.03	0.53 ± 0.16	0.32 ± 0.08	0.01 ± 0.01
$1.5\times$	Episode return	0.34 ± 0.04	0.45 ± 0.04	0.77 ± 0.01	0.11 ± 0.03	0.98 ± 0.00	0.86 ± 0.00
	Episode cost	0.43 ± 0.25	0.66 ± 0.04	0.00 ± 0.00	0.49 ± 0.16	0.24 ± 0.06	0.19 ± 0.06

Table M: Sensitivity analysis to the cost threshold used to label $\mathcal{D}_{\mathrm{S}}$, expressed as a multiple of the DSRL cost limit, with the $\mathcal{D}_{\mathrm{P}}$ threshold fixed at its default. Normalized episode return and cost are reported as 5-seed means $\pm$ standard errors. All settings satisfy the cost bar of 1.0.

Return threshold	Metric	PointCircle1	DroneCircle	Walker2dVelocity	HopperVelocity	AntVelocity	HalfCheetahVelocity
50th	Episode return	0.44 ± 0.02	0.28 ± 0.02	0.78 ± 0.01	0.18 ± 0.07	0.86 ± 0.01	0.89 ± 0.01
	Episode cost	0.22 ± 0.09	0.46 ± 0.10	0.34 ± 0.25	0.35 ± 0.16	0.21 ± 0.10	1.80 ± 0.35
75th	Episode return	0.47 ± 0.02	0.46 ± 0.05	0.80 ± 0.00	0.66 ± 0.03	0.88 ± 0.01	0.91 ± 0.01
	Episode cost	0.13 ± 0.05	0.61 ± 0.07	0.03 ± 0.01	0.40 ± 0.16	0.07 ± 0.03	0.42 ± 0.11
90th (default)	Episode return	0.27 ± 0.03	0.54 ± 0.01	0.79 ± 0.01	0.47 ± 0.07	0.96 ± 0.00	0.76 ± 0.01
	Episode cost	0.27 ± 0.07	0.80 ± 0.04	0.07 ± 0.02	0.60 ± 0.12	0.15 ± 0.03	0.45 ± 0.12
95th	Episode return	0.17 ± 0.04	0.55 ± 0.04	0.78 ± 0.00	0.34 ± 0.11	0.97 ± 0.00	0.92 ± 0.00
	Episode cost	0.07 ± 0.02	0.91 ± 0.05	0.07 ± 0.06	0.70 ± 0.12	0.15 ± 0.04	0.06 ± 0.02
$>$ max	Episode return	0.39 ± 0.03	0.56 ± 0.01	0.77 ± 0.01	0.26 ± 0.06	0.98 ± 0.01	0.69 ± 0.02
	Episode cost	0.07 ± 0.03	0.78 ± 0.05	0.08 ± 0.05	1.81 ± 0.26	0.47 ± 0.10	1.56 ± 0.39

Table N: Sensitivity analysis to the return threshold used to label $\mathcal{D}_{\mathrm{P}}$, expressed as a percentile of safe-trajectory returns ($>$ max: above the maximum safe-trajectory return), with the $\mathcal{D}_{\mathrm{S}}$ threshold fixed at its default. Normalized episode return and cost are reported as 5-seed means $\pm$ standard errors. Gray values indicate costs above the bar of 1.0; all three such cases occur at non-default thresholds.

G.11 Deployment-Time Projection onto the Estimated Safe Support

Theorem 3 applies to the ideal support-constrained solution π_κ with $\mathrm{supp}(d^{\pi_\kappa}) \subseteq \hat{\mathbf{S}}_{\tau_\kappa}$. The policy extracted in practice by weighted behavioral cloning (Appendix F.3) is a Gaussian with full support over the action space and hence cannot be hard-supported on $\hat{\mathbf{S}}_{\tau_\kappa}$. We quantify this gap by the *support-mismatch ratio*, the fraction of executed online steps whose action lies outside $\hat{\mathbf{S}}_{\tau_\kappa}$, i.e., $\hat{r}_{\mathrm{S}}(s,a) < \tau_\kappa$.

To mitigate the gap, we implement a finite-budget, gradient-based action projection at deployment time. Starting from the proposed action, the filter ascends the safe-support discriminator's logit with respect to the action, clips every update to the action bounds, and terminates once $\hat{r}_{\mathrm{S}}(s,a) \geq \tau_\kappa$; steps at which the search budget is exhausted are logged. To obtain smoother support-score gradients for the projection, the safe-support discriminator's gradient-penalty coefficient C_{S} is raised from its default 0 (Table J) to 10, matching the performance discriminator's setting $C_{\mathrm{P}} = 10$; both rows of Table O use this setting. Exact support inclusion under such filtering would require every reached state s to admit at least one action a with $(s,a) \in \hat{\mathbf{S}}_{\tau_\kappa}$, which need not hold in general, and even then a finite-budget search may fail to find it. The residual mismatch after projection is therefore reported separately and is not covered by the exact guarantee. To obtain smoother support-score gradients

for the projection, the safe-support discriminator's gradient-penalty coefficient C_{S} is set to 10 in this experiment (default 0 in Table J), matching the performance discriminator's setting $C_{\mathrm{P}} = 10$.

Table O reports results on six tasks with $N_{\mathcal{D}_{\mathrm{S}}} = N_{\mathcal{D}_{\mathrm{P}}} = 10$ over 5 seeds. On the same checkpoints, projection reduces the task-average mismatch ratio from 14.88% to 0.88%, with a modest reduction in mean return ($0.650 \to 0.623$) and a decrease in mean cost ($0.450 \to 0.357$).

Method	Metric	PointCircle1	DroneCircle	Walker2dVelocity	HopperVelocity	AntVelocity	HalfCheetahVelocity
DexDICE ($C_{\mathrm{S}} = 10$)	Episode return	0.52 ± 0.02	0.61 ± 0.01	0.78 ± 0.01	0.22 ± 0.08	0.98 ± 0.00	0.79 ± 0.01
	Episode cost	0.70 ± 0.22	0.72 ± 0.05	0.07 ± 0.02	0.67 ± 0.28	0.28 ± 0.06	0.26 ± 0.09
	Mismatch ratio	9.69%	9.57%	3.26%	19.85%	8.68%	38.23%
+ projection filter	Episode return	0.46 ± 0.03	0.58 ± 0.00	0.77 ± 0.01	0.19 ± 0.06	0.97 ± 0.00	0.77 ± 0.01
	Episode cost	0.55 ± 0.20	0.70 ± 0.03	0.10 ± 0.04	0.30 ± 0.07	0.24 ± 0.05	0.25 ± 0.10
	Mismatch ratio	2.35%	0.28%	0.27%	2.31%	0.03%	0.07%

Table O: DexDICE with deployment-time projection onto the estimated safe support ($N_{\mathcal{D}_{\mathrm{S}}} = N_{\mathcal{D}_{\mathrm{P}}} = 10$, 5 seeds). Normalized episode return and cost are reported as means $\pm$ standard errors; the mismatch ratio is the fraction of executed online steps whose action lies outside $\hat{\mathbf{S}}_{\tau_\kappa}$.

H Details of Real-World Robot Experiment

H.1 Hardware Specification

Mobile platform. We use the Robotnik SUMMIT-XL-Steel, a four-wheel indoor mobile operated in omni-directional mode. Specifications relevant to our experiments are summarized in Table P.

Dimensions ($L \times W \times H$)	$750 \times 665 \times 510$ mm
Drive	Skid-steer / omni-directional (experiments: omni-directional)
Max speed	3 m/s (experiments: 1.6 m/s, 3 rad/s)
LiDAR	Two SICK TIM571 2D scanners

Table P: Specifications of the SUMMIT-XL-Steel.

Safety violation threshold. We define a safety violation as any timestep in which the closest LiDAR-measured obstacle distance falls below $d_{\mathrm{SV}} = 0.65$ m corresponding to the robot's bounding radius of 0.624 m plus a margin for localization noise. The *violation rate* reported in Table 3b is the per-trial fraction of timesteps below d_{SV}.

H.2 MDP Specification

State. The state $s_t \in \mathbb{R}^7$ is expressed entirely in the robot body frame:

$$s_t \;=\; \big[\, v_x,\; v_y,\; \omega_z,\; \Delta x^b,\; \Delta y^b,\; \cos(\Delta\psi),\; \sin(\Delta\psi) \,\big]^\top, \tag{21}$$

where (v_x, v_y, ω_z) is the body-frame linear/angular velocity and $(\Delta x^b, \Delta y^b, \Delta\psi)$ is the goal pose relative to the current robot pose. Letting (x, y, ψ) and (x_g, y_g, ψ_g) denote the robot and goal poses in the world frame,

$$\begin{bmatrix} \Delta x^b \\ \Delta y^b \end{bmatrix} = R(-\psi) \begin{bmatrix} x_g - x \\ y_g - y \end{bmatrix}, \qquad \Delta\psi = \mathrm{wrap}_{[-\pi,\pi]}(\psi_g - \psi), \tag{22}$$

with $R(\theta) \in SO(2)$ the planar rotation. The yaw error $\Delta\psi$ is encoded as $(\cos\Delta\psi, \sin\Delta\psi)$ to avoid the $\pm\pi$ wrap-around discontinuity.

Observation (LiDAR). In addition to s_t, the policy receives a 2D LiDAR scan with 370 beams over a 360° field of view (SICK TIM571), with each ray clipped at $\ell_{\max} = 30$ m. Following Tai et al. [61], who used 10 sub-sampled range readings for mapless navigation, we apply a sparse-K encoding that retains only the $K = 10$ beams closest to the robot. Each retained beam k is represented as $(\ell_k/\ell_{\max},\; \cos\phi_k,\; \sin\phi_k)$, where ℓ_k is the measured distance and ϕ_k is the beam angle in the body frame, yielding a permutation-invariant $3K$-dimensional set representation in the spirit of PointNet [50]. This LiDAR feature is concatenated with s_t as the policy input, giving an overall input dimension of $7 + 3K$.

Action. The action $a_t \in \mathbb{R}^3$ is a body-frame velocity command:

$$a_t \;=\; \left[\, v_x^{\text{cmd}},\; v_y^{\text{cmd}},\; \omega_z^{\text{cmd}} \,\right]^\top, \qquad |v_x^{\text{cmd}}|,\; |v_y^{\text{cmd}}| \le v_{\max}, \quad |\omega_z^{\text{cmd}}| \le \omega_{\max}, \tag{23}$$

with $v_{\max} = 1.6\,\text{m/s}$ and $\omega_{\max} = 3\,\text{rad/s}$.

Goal and termination. An episode terminates upon goal reaching, collision, or after $T_{\max} = 500$ steps. An episode is counted as a success when the robot reaches the goal pose within a position tolerance $\sqrt{\Delta x^2 + \Delta y^2} < 0.5\,\text{m}$ before the timeout.

H.3 Training Setup

Datasets, observation, action. We use the three datasets defined in Section 5.3 ($N_{\mathcal{D}_\text{P}}$=10, $N_{\mathcal{D}_\text{S}}$=10, $N_{\mathcal{D}_\text{U}}$=114 including $\mathcal{D}_\text{P}$ and $\mathcal{D}_\text{S}$); observation and action spaces follow the MDP Specification above.

Baselines. DexDICE is compared against four baselines trained on the same datasets and spaces: **BC-Safe** (Gaussian BC on $\mathcal{D}_\text{S}$), **FBC** (filtered BC with a discriminator-quantile filter), **DemoDICE-U** (DemoDICE with $\mathcal{D}_\text{U}$ as the extraction set), and **FISOR-IL($\hat{c}$)**, the offline-IL variant of FISOR [70] fed with a pseudo cost from a $\mathcal{D}_\text{S}$-vs-$(\mathcal{D}_\text{U} \cup \mathcal{D}_\text{P})$ discriminator.

Architecture and optimization. Gaussian-policy methods (BC-Safe, FBC, DemoDICE-U, DexDICE) use an MLP actor with hidden sizes $[256, 256]$, whose input is the concatenation of the sparse-K LiDAR features and a 32-dimensional encoding of s_t produced by a two-layer MLP (64 and 32 units); FISOR-IL($\hat{c}$) instead uses a DDPM actor that selects among 32 candidate length-8 action chunks at inference. Each method is trained for 10^6 Adam steps with batch size 512 over 5 seeds; DexDICE uses target FNR κ=0.2 and initial Lagrange multiplier λ_0=5.

H.4 Evaluation Protocol

Calibrated 2D simulator. Before real-robot deployment, candidate policies must be validated for safe and sensible behavior to avoid hardware damage or unsafe trials. We perform this validation in a 2D simulator whose first-order velocity dynamics are calibrated to the real platform from teleoperation logs and whose arena/obstacle layout matches the physical setup.

Trials. Simulator: E=50 episodes per (method, seed) under $\pm 0.20\,\text{m}$ / $\pm 0.20\,\text{rad}$ init-pose perturbations, terminating on goal reach, collision, or $T_{\max}$=500 steps (50 s at 10 Hz); $5 \times 50 = 250$ trials per method are aggregated for pre-deployment validation. Real robot: 10 trials per method's validated checkpoint via the same 10 Hz inference loop, providing the main-paper results in Section 5.3.

Metrics.

- **Success rate**: fraction of trials reaching the goal ($\|(\Delta x, \Delta y)\| < 0.50\,\text{m}$) without a collision (closest LiDAR beam $\ge d_{\text{coll}}$=0.15 m throughout).
- **Time-to-goal**: mean episode duration over successful trials.
- **Safety-violation rate**: fraction of timesteps with closest LiDAR beam below d_{SV} (defined in Appendix H.1), regardless of whether a collision occurs.

H.5 Per-Trial Visualization

To complement the aggregate metrics in Table 3b (Section 5.3), Figure J overlays the trajectories of all 10 deployment trials per method on the navigation arena floorplan. The visualization makes the qualitative failure modes that drive the per-method differences directly inspectable.

Qualitative analysis. **BC-Safe** is the safest cloning baseline, but its conservatism slows trajectories to $\sim$40 s and stalls two trials before reaching the goal. **FBC** matches expert speed but its support filter is too permissive on hardware, leading to obstacle clipping and collisions. **FISOR-IL** with pseudo-labeled cost shows a similar fast-but-unsafe profile to FBC, with violations spread more

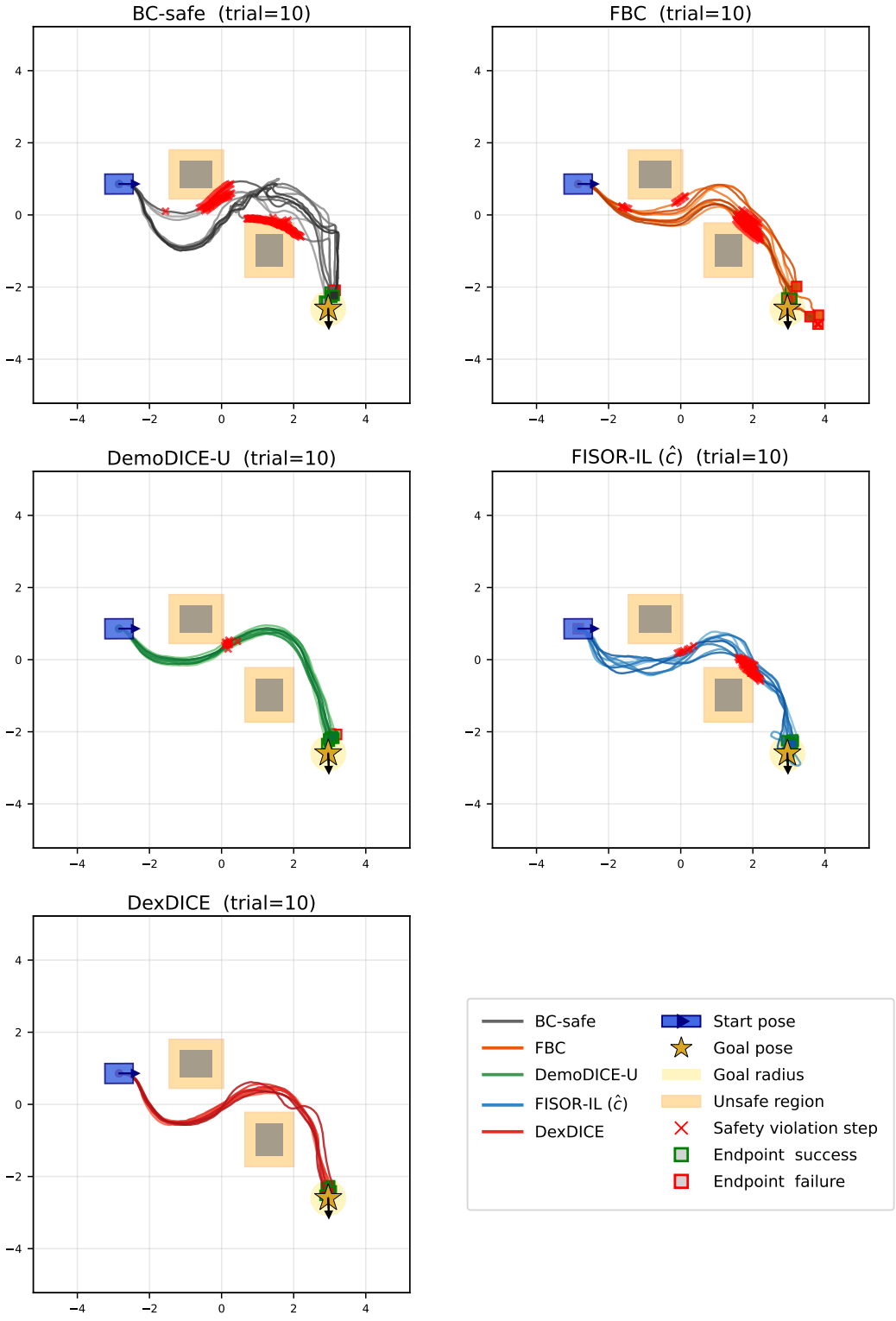


Figure J: Visualization of per-trial trajectories for the five evaluated methods across 10 deployment trials each. Each colored polyline represents one rollout; obstacle boxes are shown in gray.

uniformly across the trajectory. **DemoDICE-U** is closest to DexDICE in both safety and speed; its two failures are timeouts driven by overly cautious detours, consistent with the simulation finding that distribution matching to d_{S} biases the policy toward suboptimal safe routes. **DexDICE (ours)** reaches the goal in all ten trials along nearly indistinguishable trajectories that combine d_{P}'s directness with d_{S}'s wide obstacle margins, with zero boundary violations and tight intra-method variance.

I Compute Resources

Experiments were run on (i) two workstation nodes (NVIDIA RTX 5090 32 GiB, Intel Core Ultra 9 285K, 62 GiB RAM), (ii) four NVIDIA DGX Spark units, and (iii) Google Cloud `n4` instances with Tesla T4 accelerators provisioned on demand for sweep parallelism. A single DexDICE training run on one re-curated DSRL task takes approximately 4–6 hours on an RTX 5090. Real-world experiments (Section 5.3) ran inference on the SUMMIT-XL on-board compute and trained offline on the workstation nodes.

J Broader Impacts

This paper aims to advance safe offline imitation learning for safety-critical applications. While our framework is designed to ensure safety, it could in principle be misused to circumvent safety requirements. Deployment in domains such as autonomous driving or healthcare without rigorous validation may cause harm; responsible use lies with practitioners and deploying organizations.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The abstract and introduction claim a dual-expert offline IL formulation with a support constraint, a high-probability safety guarantee scoped to the ideal support-constrained policy (Theorem 3), and empirical gains on 38 re-curated DSRL tasks, controlled datasets, and a real robot; each claim is supported by Section 4, Section 5, and the appendices.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: The conclusion states the two main limitations: the feasibility (overlap) condition of the problem and the gap between the ideal policy covered by the guarantee and the weighted-BC policy (Appendix G.11). It also notes that the theoretical conditions are monitored only through empirical diagnostics (Appendix G.7) and that results depend on the dataset-construction thresholds (Appendix G.10).

   Guidelines:

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

3. **Theory assumptions and proofs**

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

   Answer: [Yes]

   Justification: Theorem 3 is stated informally in Section 4.3 and restated with all assumptions (pointwise soundness, i.i.d. calibration, uniform discriminator error, bounded DICE weights, safe margin) and full proofs in Appendix E. The strong-duality proposition is proven in Appendix D under an explicit domain and constraint qualification.

   Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: Section 4 specifies the algorithm, Section 5 the dataset construction and evaluation protocol, and Appendix G.4 all hyperparameters for DexDICE and every baseline; dataset and MDP details are in Appendices G.1 and H.

   Guidelines:

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

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

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

   Answer: [Yes]

   Justification: Code for DexDICE and all baselines, the DSRL re-curation script, and the real-world teleoperation dataset are publicly released (see the footnote in the abstract) with a README describing dependencies and reproduction commands. Public benchmarks (DSRL, Safety Gymnasium) are accessed through their official releases.

   Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: Section 5 reports dataset sizes, the evaluation protocol, and the averaging convention; Appendix G.4 lists per-method hyperparameters, including the fixed $\kappa = 0.01$ and the κ sweep reported in Appendix G.8.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [Yes]

Justification: Benchmark results are averaged over 50 rollouts at the last checkpoint across 5 random seeds, with standard errors in tables and shaded bands in figures; the tabular study uses 50 seeds; real-robot results report mean $\pm$ standard deviation over 10 trials.

Guidelines:

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

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: Appendix I reports the hardware used and the approximate wall-clock time of a single training run.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: The work conforms to the Code of Ethics: it uses public benchmarks under their licenses and does not collect or release personally identifying or sensitive data.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.

- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: Appendix J discusses the positive impact on safety-critical applications and the potential misuse of bypassing safety requirements without rigorous validation.

    Guidelines:

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

11. **Safeguards**

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

    Answer: [N/A]

    Justification: The released code and the small in-house teleoperation dataset involve no pre-trained generative model, scraped data, or personally identifiable information.

    Guidelines:

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

12. **Licenses for existing assets**

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

Answer: [Yes]

Justification: DSRL [44] and Safety Gymnasium [28] are cited; the DSRL datasets are under CC BY 4.0, and the DSRL code and Safety Gymnasium are under Apache License 2.0.

Guidelines:

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

13. **New assets**

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

Answer: [Yes]

Justification: The public repository (see the footnote in the abstract) documents the codebase and the teleoperation dataset (data format and usage) in its README; both are released under the MIT License.

Guidelines:

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

14. **Crowdsourcing and research with human subjects**

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

Answer: [N/A]

Justification: The teleoperation dataset was collected by one of the authors, with no external participants or compensation.

Guidelines:

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

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

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

    Answer: [N/A]

    Justification: The data was collected by one of the authors and contains only robot states, sensor readings, and joystick commands, so no human-subjects research is involved.

    Guidelines:

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

16. **Declaration of LLM usage**

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

    Answer: [N/A]

    Justification: LLMs were not used in the core method development; their use was limited to writing and editing assistance.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.