OSIL: Learning Offline Safe Imitation Policies with Safety Inferred from Non-preferred Trajectories

Returaj Burnwal
Indian Institute of Technology Madras
Chennai, India
cs21d406@smail.iitm.ac.in

Nirav Pravinbhai Bhatt
Indian Institute of Technology Madras
Chennai, India
niravbhatt@dsai.iitm.ac.in

Balaraman Ravindran
Indian Institute of Technology Madras
Chennai, India
ravi@dsai.iitm.ac.in

ABSTRACT

This work addresses the problem of offline safe imitation learning (IL), where the goal is to learn safe and reward-maximizing policies from demonstrations that do not have per-timestep safety cost or reward information. In many real-world domains, online learning in the environment can be risky, and specifying accurate safety costs can be difficult. However, it is often feasible to collect trajectories that reflect undesirable or unsafe behavior, implicitly conveying what the agent should avoid. We refer to these as non-preferred trajectories. We propose a novel offline safe IL algorithm, OSIL, that infers safety from non-preferred demonstrations. We formulate safe policy learning as a Constrained Markov Decision Process (CMDP). Instead of relying on explicit safety cost and reward annotations, OSIL reformulates the CMDP problem by deriving a lower bound on reward maximizing objective and learning a cost model that estimates the likelihood of non-preferred behavior. Our approach allows agents to learn safe and reward-maximizing behavior entirely from offline demonstrations. We empirically demonstrate that our approach can learn safer policies that satisfy cost constraints without degrading the reward performance, thus outperforming several baselines.

KEYWORDS

Imitation Learning, Offline RL, Safe RL

1 INTRODUCTION

Reinforcement Learning (RL) provides a framework for enabling agents to autonomously learn intelligent behavior through interaction with their environment. While RL has recently demonstrated remarkable success in various simulated tasks [5, 25, 31], its real-world deployment remains limited [1, 15, 24] due to the following reasons. First, RL typically requires many online interactions with the environment to learn a good policy. This reliance on data collection poses a significant barrier in domains where such interactions

are costly or risky, such as robotics or autonomous driving. Second, designing a suitable reward function that accurately reflects the desired behavior is often difficult in practice. Since the agent learns by maximizing the reward, poorly defined reward functions can lead to unintended and potentially dangerous behavior [4, 11]. Third, many real-world applications necessitate adherence to safety constraints alongside reward maximization. Safety in RL is usually modeled as a Constrained Markov Decision Process (CMDP) [2]. In CMDP, the agent aims to maximize rewards over time while satisfying safety cost constraints. However, specifying appropriate per-timestep constraint cost information can be equally difficult. For instance, defining constraint cost information for assessing toxicity in conversational agents or complex domains like surgical robotics can be challenging.

Consider a real-world scenario of autonomous driving where we have access to a large collection of human-driven trajectories that are high-return, but exhibit varying degrees of safety cost. Standard imitation learning (IL) algorithms [3, 28] can learn such high-return behavior. However, in most real-world applications, it is often necessary to ensure that learned behaviors also satisfy safety constraints. In this paper, we address the problem of offline safe IL, when given a large set of high-return trajectories with varying degrees of safety cost, *union trajectory dataset*, and a small set of *non-preferred trajectory dataset* that are high-return and high-cost. We aim to learn a policy that infers safety from these non-preferred demonstrations. The non-preferred trajectories that violate safety constraints are often naturally collected. For example, instances of illegal driving, like running red lights, can be used as non-preferred trajectories. This setting has mostly been unexplored and cannot be effectively addressed using the existing methods. The recent work on offline safe IL, SafeDICE [15], learns a safe policy under this setting. However, the SafeDICE makes a limiting assumption: that the *union dataset*, high-return trajectories, contains either low-cost or high-cost behavior. In practice, high-return trajectories may not fit neatly into this binary categorization. Another relevant approach is based on preference-based policy learning (PPL) [7, 20, 21, 30], which learns a policy based on preferences over pairs of demonstrations. PPL can be used to train a policy that favors *union trajectories* over non-preferred ones. However, this approach assumes that *union trajectories* are safe, which is not guaranteed. A more realistic assumption would be to allow these high-return *union trajectories* to have varying degrees of safety cost.

We propose a novel offline safe IL algorithm, OSIL, that infers safety from these non-preferred demonstrations. Our key contributions are as follows: 1) We formulate the safe policy learning as a CMDP. Since we do not have access to reward and cost information, we reformulate the CMDP problem by deriving a lower bound

Union Dataset Composition

	Low Cost	Intermediate Cost	High Cost
SafeDICE	O	✕	O
PPL	O	✕	✕
OSIL (our)	O	O	O

(a)
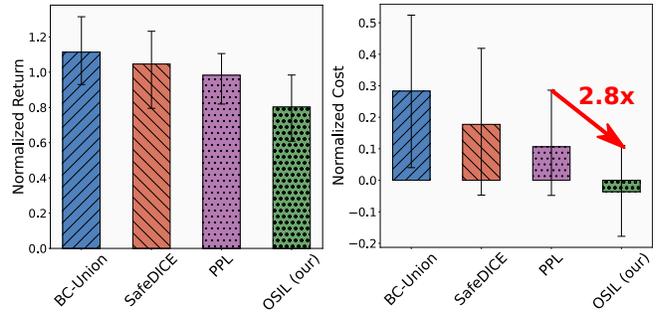
(b)

Figure 1: (a) PPL works well when the union dataset consists of low-cost trajectories and SafeDICE requires that the union dataset contains low-cost or high-cost trajectories. This assumption is often unrealistic in practice since trajectories collected from real-world typically span a spectrum of safety costs. OSIL do not make any assumptions about the safety cost of a trajectory in the union dataset, as it may contain trajectories with varying levels of safety cost. (b) Under this setting, we observe that OSIL learns a high-return, safer (i.e., low-cost) policy, outperforming the best baseline by nearly 2.8x. We report the mean performance of the algorithm after 1 million training steps, aggregated across both velocity-constrained and navigation tasks. Mean and 95% CIs over 5 seeds.

of the reward maximizing objective and approximating the safety constraints by learning a parameterized cost model that estimates the likelihood of a state-action pair being non-preferred. 2) We then solve the reformulated CMDP problem using a Lagrangian relaxation with an adaptive penalty coefficient to obtain a policy that effectively balances safety and performance. 3) We empirically demonstrate that OSIL learns safer policies (Figure 1) than existing state-of-the-art offline safe IL methods, while maintaining competitive task success.

2 RELATED WORKS

Offline Imitation Learning [17, 18, 26] primarily focuses on replicating actions of the expert demonstrations without explicitly considering safety. It implicitly assumes that expert demonstrations are safe, but if they contain non-preferred trajectories, simply imitating them can lead to learning policies that are non-preferred. We address the problem of learning safe imitation policy by seeking limited access to non-preferred trajectories and an abundance of high-return union trajectories that contain trajectories with varying costs. This specific scenario has received limited attention in the literature.

Learning Imitation Policy from Non-Preferred Trajectories: SafeDICE [15] directly estimating the stationary distribution of the low-cost policy using non-preferred and union trajectory dataset. However, it assumes that the union dataset contain either low-cost or high-cost trajectories, a simplification that often does not hold in practice. The estimated stationary distribution is then used to learn a safe policy that avoids high-cost behaviors.

Learning Imitation Policy from Suboptimal Trajectories: Preference based policy learning (PPL) methods, such as T-REX [7], B-Pref [20], PEBBLE [21], OPRL [30], aim to learn a reward function from ranked trajectories. These methods prioritize learning reward functions that assign higher total reward to higher ranked trajectories in the dataset. T-REX, B-Pref, PEBBLE then uses online-RL [14, 29] algorithm, and OPRL uses offline-RL [19, 27] algorithm to

learn the policy. Since ranked trajectories are difficult to obtain, D-REX [8] and SSRR [10] both learn reward function by generating trajectories of varying optimality by injecting noise to the policy learned from suboptimal demonstrations. However, the trajectory generation requires online interaction with the environment, which is not feasible in an offline setting. Another method, DWBC [32] uses positive unlabeled learning [12, 33] to train a discriminator network. This discriminator network is subsequently used as a weight in the BC loss function.

3 PRELIMINARIES

3.1 Constrained Markov Decision Process

Consider a finite Constrained Markov Decision Process (CMDP) [2] represented by the tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{T}, r, \rho_0, \gamma, C)$, where $\mathcal{S}$ and $\mathcal{A}$ represent state and action spaces, respectively. $\mathcal{T} : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ is the transition probability, $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ denotes the immediate reward function, $\rho_0 \in \Delta(\mathcal{S})$ is the initial state distribution, and $\gamma \in (0, 1)$ is the discount factor. $C = \{(c_i, b_i)\}_{i=1}^{m}$ is a constraint set, where $c_i : \mathcal{S} \times \mathcal{A} \to \mathbb{R}_{\geq 0}$ is the i-th cost function and $b_i \in \mathbb{R}_{\geq 0}$ is the corresponding threshold. A stochastic policy $\pi : \mathcal{S} \to \Delta(\mathcal{A})$ corresponds to a map from state to a probability distribution over actions. The reward value function and action-value function under policy π is defined as $V_r^{\pi}(s) = \mathbb{E}_{\tau \sim \pi}[\sum_{t=0}^{T-1} \gamma^t r(s_t, a_t)|s_0 = s]$ and $Q_r^{\pi}(s) = \mathbb{E}_{\tau \sim \pi}[\sum_{t=0}^{T-1} \gamma^t r(s_t, a_t)|s_0 = s, a_0 = a]$, where $\tau = (s_0, a_0, \ldots, s_{T-1}, a_{T-1}, s_T)$ is a T length trajectory sampled under policy π. Similarly, the i-th cost action-value function is defined as $Q_{c_i}^{\pi}(s, a) = \mathbb{E}_{\tau \sim \pi}[\sum_{t=0}^{T-1} \gamma^t c_i(s_t, a_t)|s_0 = s, a_0 = a]$. Therefore, the reinforcement learning problem in CMDP is

$$\pi^{\star} = arg \max_{\pi} J_r(\pi) := \mathbb{E}_{\tau \sim \pi} \left[\sum_{t=0}^{T-1} \gamma^t r(s_t, a_t) \right]$$

$$\text{s.t. } \mathbb{E}_{s \sim \rho_0} \left[Q_{c_i}^{\pi}(s, \pi(s)) \right] \leq b_i, \quad \forall i \tag{1}$$

where $J_r(\pi)$ denotes the the performance of the policy π and $\mathbb{E}_{s \sim \rho_0} \left[Q_{c_i}^{\pi}(s, \pi(s)) \right] \leq b_i$ is the i-th constraint. Imitation learning
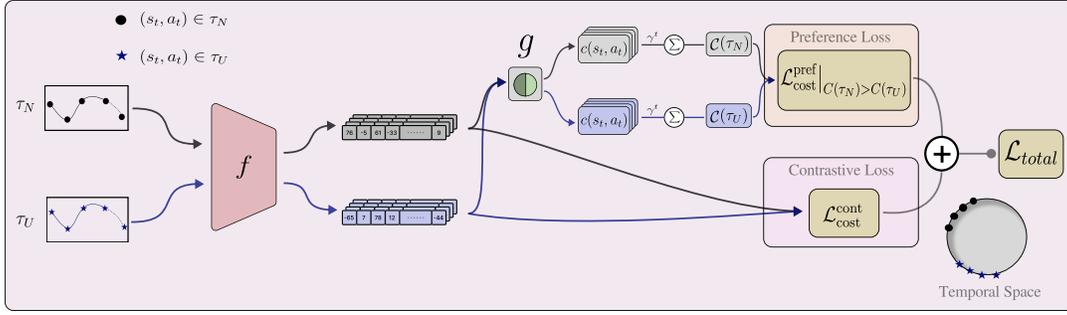
Figure 2: Overview of the cost learning model. f and g are a learnable encoder and a linear model, respectively. The cost model is trained by minimizing the two loss function: $\mathcal{L}_{\text{cost}}^{\text{const}}$ encourages temporally adjacent state-action pairs within a trajectory to remain close in the learned representation space, and $\mathcal{L}_{\text{cost}}^{\text{pref}}$ ensures that the discounted cost of the trajectory τ_N is greater than trajectory τ_U.

does not rely on environmental rewards but rather on demonstrations (trajectories). This work focuses on an offline setting, where the policy is learned solely from pre-collected trajectories that do not contain reward and cost information explicitly, i.e., $\tau = (s_0, a_0, \ldots, s_{T-1}, a_{T-1}, s_T)$. In this work, non-preferred and union trajectories are defined as follows:

Definition 1 (Non-preferred Trajectory Dataset). *A non-preferred trajectory dataset contains trajectories that achieve high return but violate multiple constraints by a large margin. In other words, it is a high-return, high-cost trajectory dataset.*

Definition 2 (Union Trajectory Dataset). *A union trajectory dataset contains trajectories that achieve high return with varying degrees of constraint violation. In other words, it is a high-return, varying-cost trajectory dataset.*

We assume we have access to a limited number of non-preferred trajectories, $\mathcal{D}_N$, and a large number of union trajectories, $\mathcal{D}_U$. Additionally, we represent the policy that generated the union trajectory dataset as π_U. This means the actions in the trajectory, $\tau \in \mathcal{D}_U$, are sampled from π_U.

3.2 Contrastive Learning

The procedure starts with a set of N randomly sampled labeled examples, $\{(x_i, y_i)\}_{i=1}^{N}$. For each sample, we generate two random augmentations, resulting in a batch of $2N$ augmented pairs: $\mathcal{B} = \{(\tilde{x}_i, \tilde{y}_i)\}_{i=1}^{2N}$, where for each $k \in \{1, 2, \ldots, N\}$, the augmented inputs $\tilde{x}_{2k-1}$ and $\tilde{x}_{2k}$ are derived from x_k, and both share the same label: $\tilde{y}_{2k-1} = \tilde{y}_{2k} = y_k$. This batch of $2N$ pairs is then used to train an encoder $f(\cdot)$ that maps input x to a d-dimensional unit-norm representation vectors $z = f(x) \in \mathbb{R}^d$.

Supervised Contrastive Training [16]: We train the encoder model f using a batch $\mathcal{B}$ of $2N$ augmented labeled sample pairs. The training objective of the supervised contrastive loss is defined as:

$$\mathcal{L}^{sup} = \sum_{i \in \mathcal{B}} \frac{-1}{|P(i)|} \sum_{p \in P(i)} \log \frac{\exp(z_i^\top z_p / \eta)}{\sum_{b \in \mathcal{B} \setminus i} \exp(z_i^\top z_b / \eta)} \quad (2)$$

Here, $z_i = f(\tilde{x}_i)$ is the encoded representation of the augmented input $\tilde{x}_i$, $P(i) = \{p \in \mathcal{B} \setminus i \mid \tilde{y}_p = \tilde{y}_i\}$ is the set of all positive samples in the batch that share the same label as $\tilde{x}_i$, $|P(i)|$ denotes its cardinality, and $\eta \in \mathbb{R}^+$ is a temperature scaling parameter. This contrastive loss enables the model to learn label-aware representations by pulling together samples from the same class in the embedding space, thereby promoting class-consistent representations.

4 METHODOLOGY

This section introduces OSIL method which formulates safe policy learning as a CMDP problem, defined in Equation 1. In general, both the reward and safety cost functions are difficult to define, and in our work, we do not assume access to them. Instead, we assume access to a limited number of non-preferred trajectories $\mathcal{D}_N$ and union trajectories $\mathcal{D}_U$. In Section 4.1, we first approximate the safety cost functions using a parameterized cost model that estimates the likelihood of a state-action pair being non-preferred and use it to learn the cost action-value function. Finally, in Section 4.2 we provide a lower bound on policy's performance, $J_r(\pi)$, that does not depend on reward labels and solves the resulting optimization problem to obtain a policy that effectively balances safety and performance.

4.1 Learning Cost Action-Value Function

To learn a cost action-value function, we first need to learn a cost model that effectively captures the safety cost information. We define our parameterized cost model as a composition $\tilde{c} := g \circ f$, where $f : \mathcal{S} \times \mathcal{A} \to \mathbb{R}^d$ is an encoder model that maps state-action pairs to a d-dimensional unit-norm latent representation, and $g : \mathbb{R}^d \to [0, 1]$ is a linear model that maps the encoded representation to a scalar value (See Figure 2).

Loss function for cost model: The encoder model f is trained such that the state-action pairs that appear in the same trajectory have similar representations, thereby encouraging the model to capture temporal dependencies. We use partial trajectory of length $H << T$, rather than full trajectory of length T. This has two benefits: (1) short partial trajectories better capture local temporal correlations, as nearby state-action pairs tend to be more similar; and (2) shorter sequences reduce computational overhead during

training. Given a batch of trajectories from the union dataset $\mathcal{B}_U \sim \mathcal{D}_U$, and the non-preferred dataset $\mathcal{B}_N \sim \mathcal{D}_N$, we train the encoder using a contrastive loss:

$$\mathcal{L}_{\text{cost}}^{\text{cont}} = \sum_{\tau \in \mathcal{B}_U \cup \mathcal{B}_N} \sum_{t \in \tau} \frac{-1}{|\tau| - 1} \sum_{t' \in \tau \setminus t} \log \frac{\exp(z_t^\top z_{t'}/\eta)}{\sum_{\tau \in \mathcal{B}_U \cup \mathcal{B}_N} \sum_{k \in \tau} \exp(z_t^\top z_k/\eta)} \tag{3}$$

Here, $t \in \tau$ denotes a state-action pair (s_t, a_t) in trajectory τ, $z_t = f(s_t, a_t)$ is its encoded representation, and $|\cdot|$ represents the cardinality. Minimizing $\mathcal{L}_{\text{cost}}^{\text{cont}}$ ensures that the state-action pairs in the same trajectory are temporally close.

The output of the encoder model is then passed to the linear model g that estimates the likelihood of the state-action pair being non-preferred. Given a trajectory τ we define the total discounted cost of a trajectory as $C(\tau) = \sum_{t=0}^{T-1} \gamma^t \tilde{c}(s_t, a_t)$ where $\tilde{c} := g \circ f$ is our cost model. Since the non-preferred trajectories typically incur higher costs than those from the union dataset, we introduce a preference-based training objective to encourage the cost-model to assign higher costs to non-preferred trajectory τ_N than to union trajectory τ_U, i.e., $C(\tau_N) > C(\tau_U)$. To formalize this preference, we define the loss function based on the Bradley–Terry model [6] as:

$$p_{\text{non}} = p(\tau_N \succ \tau_U) = \frac{\exp(C(\tau_N))}{\exp(C(\tau_N)) + \exp(C(\tau_U))} \tag{4}$$

$$\mathcal{L}_{\text{cost}}^{\text{pref}} = \mathbb{E}_{\tau_N, \tau_U} \left[-\log p_{\text{non}} \right] = \mathbb{E}_{\tau_N, \tau_U} \left[\text{BCE}(p_{\text{non}}; \mathbb{1}) \right] \tag{5}$$

Here, p_{non} represents the prediction probability that τ_N has higher cost over τ_U, and BCE$(\cdot, \cdot)$ denotes binary cross-entropy loss function with a hard target label as $\mathbb{1}$, indicating a strong supervision signal that τ_N should have higher cost than τ_U. While this hard labeling assumption may not always strictly hold, since some of the trajectories in the union dataset are high-cost, we observe in our experiments that using this hard labeling does not degrade the performance. This is likely because, on average, trajectories sampled from the union dataset tend to have lower cost than those from the non-preferred dataset, making the pairwise ranking signal reliable in practice.

Finally, we jointly train our cost model by minimizing both the contrastive and the preference-based loss function:

$$\mathcal{L}_{\text{cost}} = \mathcal{L}_{\text{cost}}^{\text{pref}} + \mathcal{L}_{\text{cost}}^{\text{cont}} \tag{6}$$

Loss function for cost action-value: We train our cost value function $Q_c^\pi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ for a given policy π, using a learned cost model over the union dataset. The cost action-value function $Q_{\tilde{c}}^\pi(s, a)$ represents the expected long-term cost incurred by taking action a in state s and subsequently following policy π. To estimate Q_c^π, we iteratively minimize the following squared temporal-difference (TD) loss:

$$\mathcal{L}_{\text{value}} = \mathbb{E}_{(s,a,s') \sim \mathcal{D}_U} \left[(y_{\text{target}}(s, a, s') - Q_{\tilde{c}}^\pi(s, a))^2 \right] \tag{7}$$

where $y_{\text{target}}(s, a, s') = \tilde{c}(s, a) + \gamma Q_{\tilde{c}}^{\pi^-}(s', \pi(s'))$ is the TD target, $\tilde{c}(s, a)$ is the immediate cost estimated from the learned cost model, and $Q_{\tilde{c}}^{\pi^-}$ is the target Q network which is slowly updated using polyak averaging scheme $Q_{\tilde{c}}^{\pi^-} \leftarrow (1 - \zeta) Q_{\tilde{c}}^{\pi^-} + \zeta Q_{\tilde{c}}^\pi$ with a fixed smoothing coefficient $\zeta \in (0, 1)$.

4.2 Learning Safe Policy

For a given cost action-value function we can rewrite the reinforcement learning problem for CMDP as:

$$arg \max_\pi J_r(\pi) := \mathbb{E}_{\tau \sim \pi} \left[\sum_{t=0}^{T-1} \gamma^t r(s_t, a_t) \right]$$

$$\text{s.t.} \quad \mathbb{E}_{s \sim \rho_0} \left[Q_c^\pi(s, \pi(s)) \right] \leq b \tag{8}$$

Solving this constrained optimization is challenging, as we do not have access to the reward information. However, we know that the union dataset $\mathcal{D}_U$ contains high-return trajectories. We derive a lower bound on the performance of the policy π in terms of the performance of the policy π_U that generated $\mathcal{D}_U$.

Theorem 1. *Let π_U represents the policy that generated union dataset. Define $\epsilon = \max_{s,a} |Q_r^\pi(s, a) - V_r^\pi(s)|$, and $D_{KL}^{max}(\pi_U, \pi) = \max_s D_{KL}(\pi_U(.|s) || \pi(.|s))$. Then the following bound holds:*

$$J_r(\pi) \geq J_r(\pi_U) - \frac{2\epsilon}{1 - \gamma} \sqrt{D_{KL}^{max}(\pi_U, \pi)} \tag{9}$$

The proof of the theorem can be found in [9, Appendix A]. Since π_U is assumed to produce high-return trajectories, $J_r(\pi_U)$ serves as a proxy for the maximum achievable performance.

The objective in Equation 8 can be optimized by maximizing the lower bound on $J_r(\pi)$. This leads to the following reformulation of the objective equation 8 as:

$$arg \max_\pi J_r(\pi) \succcurlyeq arg \max_\pi J_r(\pi_U) - C\sqrt{D_{\text{KL}}^{\max}(\pi_U, \pi)}$$

$$\stackrel{(a)}{=} J_r(\pi_U) - arg \max_\pi C\sqrt{D_{\text{KL}}^{\max}(\pi_U, \pi)}$$

$$= arg \min_\pi D_{\text{KL}}^{\max}(\pi_U, \pi) \tag{10}$$

Here, $C = \frac{2\epsilon}{1 - \gamma}$, and step (a) follows from the fact that $J_r(\pi_U)$ is the maximum achievable performance. The equation 10 imposes constraint that the KL divergence is bounded at every point in the state space. While this formulation is theoretically grounded, it is often impractical to solve due to the large number of constraints. To make the problem tractable, we approximate the constraint using the average KL divergence.

$$\mathbb{E}_{s \sim \mathcal{D}_U} \left[D_{\text{KL}}(\pi_U || \pi) \right] = \mathbb{E}_{s \sim \mathcal{D}_U} \left[\mathbb{E}_{a \sim \pi_U(.|s)} \left[\log \frac{\pi_U(a|s)}{\pi(a|s)} \right] \right]$$

$$= \mathbb{E}_{(s,a) \sim \mathcal{D}_U} \left[\log \frac{\pi_U(a|s)}{\pi(a|s)} \right] \tag{11}$$

The implication of this approximation is discussed in [9, Appendix H.1]. The CMDP problem, Equation 1, is reformulated as:

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

$$\text{s.t.} \quad \mathbb{E}_{s \sim \rho_0, a \sim \pi} \left[Q_{\tilde{c}}^\pi(s, a) \right] \leq b \tag{12}$$

We relax the constrained optimization problem Equation 12 using the Lagrangian formulation. We introduce a penalty coefficient α and solve the following unconstrained optimization problem:

$$\mathcal{L}_{\text{policy}} = -\mathbb{E}_{(s,a) \sim \mathcal{D}_U} \left[\log \pi(a|s) \right] + \mathbb{E}_{s \sim \rho_0} \left[\alpha \, Q_{\tilde{c}}^\pi(s, \pi(s)) \right]$$

$$= \underbrace{-\mathbb{E}_{(s,a) \sim \mathcal{D}_U} \left[\log \pi(a|s) \right]}_{\text{Behavior Cloning}} + \underbrace{\mathbb{E}_{(s_0, a_0) \sim \mathcal{D}_U} \left[\alpha \, Q_{\tilde{c}}^\pi(s_0, \pi(s_0)) \right]}_{\text{Cost Critic}}$$
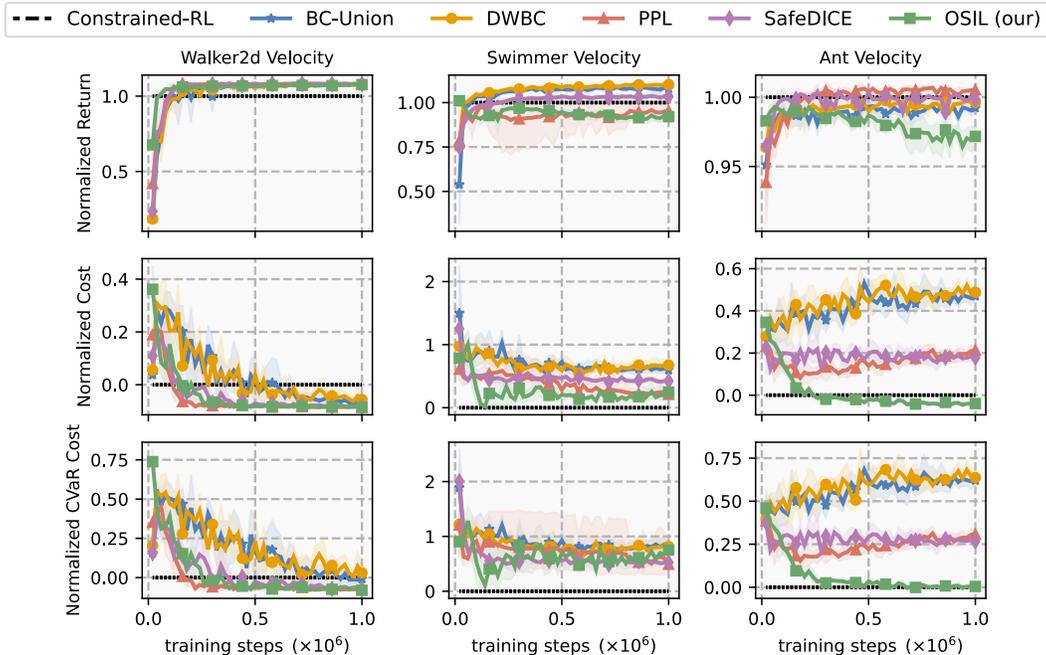
$$\tag{13}$$

Figure 3: Performance Comparison. Experimental results on Walker2d-Velocity, Swimmer-Velocity, Ant-Velocity task. The shaded area represents the standard error. In velocity-constrained tasks, our method is able to recover safer policies without compromising reward performance.

In Equation 13, the loss function consists of two components: a behavior cloning term that encourages the policy π to imitate actions from the high-return dataset $\mathcal{D}_U$, and a cost-critic term that penalizes risky behavior by minimizing the cost-action value $Q_{\hat{c}}^{\pi}$. This formulation resembles TD3+BC [13], an offline RL algorithm that also combines behavior cloning with value function for policy learning. However, a key distinction lies in the critic: TD3+BC uses a reward action-value function to guide the policy toward high-return actions, while our approach employs a cost-action value function to promote safe actions.

The hyperparameter α, in equation 13, balances the trade-off between safety (minimizing cost-action value) and performance (maximizing reward through imitation). We choose an adaptive scalar α that balances this trade-off:

$$\alpha = \frac{\bar{\alpha}}{\frac{1}{N} \sum_{(s_0,a_0) \sim \mathcal{D}_U} \exp\left(Q_{\hat{c}}^{\pi}(s_0,a_0) - Q_{\hat{c}}^{\pi}(s_0,\pi(s))\right)} \quad (14)$$

This adaptive formulation ensures that α increases when the learned policy selects actions that are, on average, more costly than those in the dataset, i.e., $Q_{\hat{c}}^{\pi}(s_0,a_0) < Q_{\hat{c}}^{\pi}(s_0,\pi(s))$. In such cases, the policy learns safe behavior by placing a higher weight on the cost-critic term. Conversely, when the policy consistently chooses safer actions, i.e., $Q_{\hat{c}}^{\pi}(s_0,a_0) \geq Q_{\hat{c}}^{\pi}(s_0,\pi(s))$, the value of α decreases, reducing the regularization and allowing the policy to focus more on high-return behavior.

5 EXPERIMENTS

We evaluate our algorithm against the state-of-the-art offline safe IL method using the Datasets for Offline Safe RL (DSRL) across a suite of benchmark tasks [22]. Specifically, we run experiments on: (i) MuJoCo-based velocity-constrained tasks (Walker-Velocity, Swimmer-Velocity, and Ant-Velocity), where agents are required to move as fast as possible while respecting velocity limits; and (ii) navigation tasks (Point-Circle2, Point-Goal1, and Point-Button1), where agents aim to maximize performance while avoiding collisions and hazardous areas (See [9, Appendix B] for further details). Together, we conduct experiments on six tasks, providing a comprehensive scenario to assess the safety and effectiveness of offline safe IL algorithms.

We select a limited number of non-preferred (i.e., high-return-high-cost behavior) trajectories, $\mathcal{D}_N$, and a large number of high-return varying cost union trajectories, $\mathcal{D}_U$ from the DSRL dataset. We removed all the reward and cost annotations from the non-preferred ($\mathcal{D}_N$) and the union ($\mathcal{D}_U$) trajectory dataset. Our goal is to recover low-cost behavior while preserving high-reward performance. See [9, Appendix C] for additional details on the $\mathcal{D}_N$ and $\mathcal{D}_U$ datasets for each environment.

Baselines: We compare our algorithm against three baselines. **(1) BC-Union**, a BC policy on the union dataset $\mathcal{D}_U$. This serves as a baseline to evaluate the performance of BC when the dataset contains high-return and varying cost trajectories. **(2) DWBC**, a variant of the discriminator-weighted behavior cloning algorithm [32] that uses *negative-unlabeled* learning to train the discriminator model. The trained discriminator is then used as weight in the
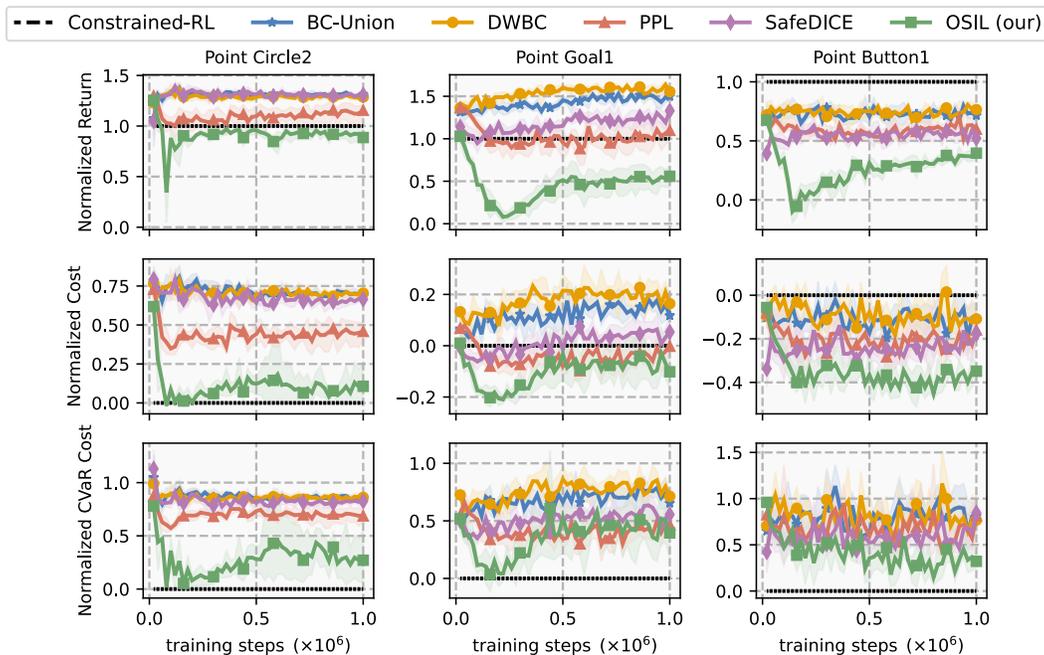
Figure 4: Performance Comparison. Experimental results on Point-Circle2, Point-Goal1, Point-Button1 tasks. The shaded area represents the standard error. Similar to the results in Figure 3, our method is able to recover safer policy compared to other baselines.

weighted BC loss function. **(3) PPL** [7, 20, 21, 30], a preference-based method that learns a reward function to prefer union trajectories over non-preferred ones. The learned reward function is then used as a weight in the weighted BC loss function. **(4) SafeDICE** [15], method directly estimates the stationary distribution corrections for the low-cost behavior and then trains a weighted BC policy. See [9, Appendix E], for the implementation details for all the baselines.

Additionally, we train an offline constrained RL policy, Constrained Decision Transformer [23], using the union dataset augmented with ground-truth reward and cost annotations. In our work, we use Constrained-RL method, to estimate the safe performance under complete information. We compare OSIL, which operates under stricter information constraints, to assess how close its performance is to the Constrained-RL approach. Constrained-RL method is not a baseline. No other methods in our experiments have access to reward and cost information.

We evaluate and compare ours and other baselines algorithms in terms of task performance (measured by expected episodic return) and safety (measured by expected episodic cost). We report the results with the following metrics: **(1) Normalized Return**: scales the mean episodic return of a given policy, such that "0" represents the episodic return from a random policy, while "1" represents the episodic return achieved by the Constrained-RL policy. **(2) Normalized Cost**: scales the given policy's mean episodic cost, such that "0" represents the cost achieved by the Constrained-RL policy, and "1" represents the maximum episodic cost. **(3) Normalized**

Conditional Value at Risk performance (CVaR) 20% Cost: similar to *Normalized Cost* it scales the policy's mean episodic cost of the worst 20% runs.

Our primary focus in the evaluation is on safety, measured by *Normalized Cost* and *Normalized CVaR Cost*. We use *Normalized Return* to examine whether the agents can achieve high-return behaviors. We consider a policy to have successfully recovered the low-cost behavior if its performance closely matches the Constrained-RL policy. This is evidenced by the *Normalized Cost* and *Normalized CVaR Cost* less than or close to "0", and the *Normalized Return* being greater than or close to "1". All plots are generated by averaging the performance of 50 trajectories generated from the learned policy. To assess statistical significance, we generate 1000 bootstrap samples from the data, using results from 5 different random seeds, and we plot the resulting 95% confidence intervals.

We answer the following questions through our experiments:

(1) *Performance Comparison*: How does our algorithm perform relative to other baselines in learning safe policies that satisfy cost constraints while preserving high reward?

(2) *Impact of Non-Preferred Trajectory Dataset Size*: How does performance vary with the size of the non-preferred trajectory dataset $\mathcal{D}_N$?

(3) *Impact of Contrastive Loss*: What is the effect of removing contrastive loss from the cost model training?

(4) *Sensitivity to Trajectory Length*: How does the choice of sub trajectory length affect the algorithm's performance?
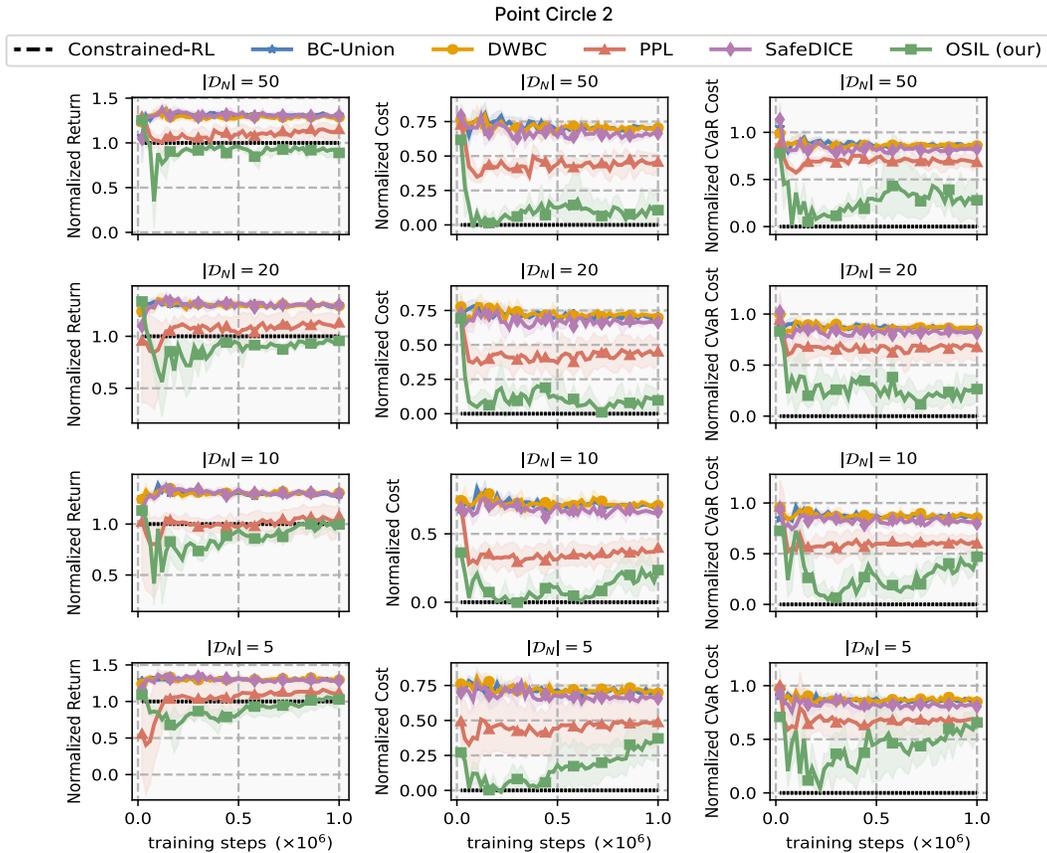
Figure 5: Impact of Non-Preferred Trajectory Dataset Size. Experimental result on Point-Circle2 with varying non-preferred trajectory dataset size $\mathcal{D}_N = \{5, 10, 20, 50\}$. **We observe the performance gradually decreases with smaller $|\mathcal{D}_N|$. However, our approach consistently outperforms all baselines across all different dataset size.**

5.1 Performance Comparison

In both the velocity-constrained tasks (Figure 3) and navigation tasks (Figure 4), we observe that the standard BC policy trained on the union dataset can recover high-reward behavior. However, the resulting policy also incurs a high cost, which is expected, as the union dataset contains trajectories with varying cost levels. Our algorithm, OSIL, effectively recovers low-cost behavior from the union dataset in both velocity-constrained and navigation tasks, achieving safety performance comparable to that of the Constrained RL policy. In the Walker2d velocity task, most baselines succeed in recovering low-cost behavior. We find that SafeDICE performs similarly to the BC-Union policy across most domains. This can be due to its strong assumption that the union dataset should contain only low-cost and high-cost trajectories, which is a stringent requirement. On the other hand, PPL utilizes the fact that the union trajectories are generally less costly than the non-preferred ones. It learns a reward model that prefers union over non-preferred trajectories, thereby capturing some notion of safety in the state-action space. As a result, PPL outperforms SafeDICE and BC-Union but still underperforms compared to our method.

5.2 Ablation and Sensitivity Analysis

Impact of Non-Preferred Trajectory Dataset Size: We studied the impact of non-preferred dataset size on our algorithm by varying $|\mathcal{D}_N| \in \{5, 10, 20, 50\}$ in the Point-Circle2 environment. As shown in Figure 5, the performance gradually decreases with smaller $|\mathcal{D}_N|$, indicating the importance of non-preferred examples. We observe that when $|\mathcal{D}_N|$ is large, the non-preferred behaviors are well represented, we are able to recover low-cost behavior. However, as $|\mathcal{D}_N|$ decreases, the coverage of these non-preferred behaviors also decreases, making it harder for OSIL and other baselines to learn safe behavior. Nonetheless, our approach consistently outperforms all baselines across all different dataset sizes.

Impact of Contrastive Loss: To evaluate the impact of the contrastive loss term in our OSIL algorithm, we performed experiments on all navigational and velocity-constrained tasks. All other design choices were held constant, and we removed the contrastive loss term, $\mathcal{L}_{cost}^{cont}$, for comparison. We trained both variants of OSIL with and without the contrastive loss for 1 million training steps. The final performance results were aggregated across all six tasks, as
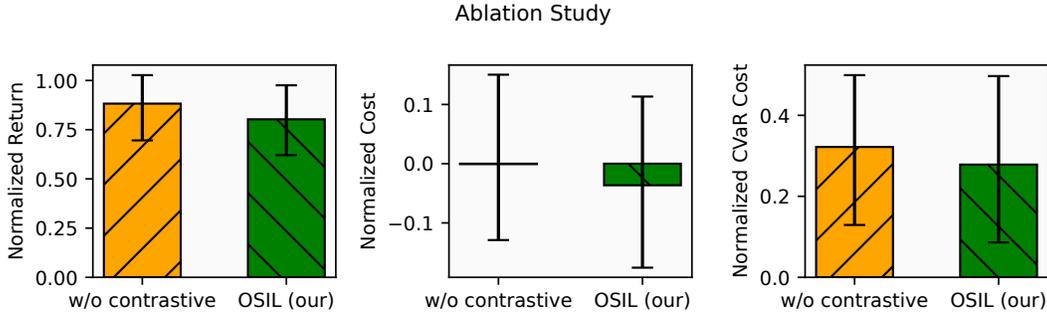
Figure 6: Impact of Contrastive loss. We report the mean performance of the algorithm after 1 million training steps, aggregated across both velocity-constrained and navigation tasks. Mean and 95% CIs over 5 seeds. Our ablation highlights the significance of the contrastive loss in overall performance. See [9, Appendix G.1] for per-task performance.
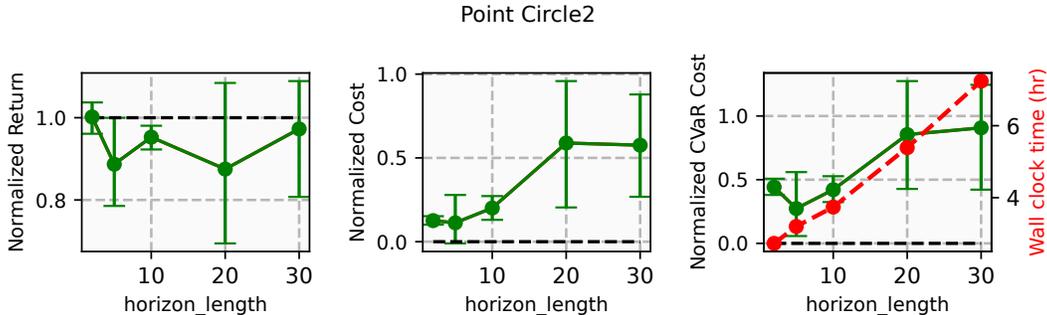

Figure 7: Sensitivity to Trajectory Length. We report the final mean performance of the algorithm on Point Circle2 environment for different partial trajectory length $H = \{2, 5, 10, 20, 30\}$, after training for 1 million steps. Mean and 95% CIs over 5 seeds. We observe that the shorter trajectory length tends to better capture temporal correlations and are computationally more efficient (see rightmost plot, the red curve indicates wall-clock training time in hours).

shown in Figure 6. The results show that the contrastive loss significantly aids in learning lower-cost policies without compromising overall return.

Sensitivity to Trajectory Length: To analyze the effect of different partial trajectory lengths H on our algorithm, we performed experiments on the Point-Circle2 environment for different values of $H = \{2, 5, 10, 20, 30\}$. Figure 7 summarizes the final performance results. We observe that shorter trajectories can effectively capture temporal correlations, which facilitates learning a better cost function and improves overall performance. Our finding also supports using partial trajectories for training, as they offer a computationally efficient alternative to full-length trajectories without compromising overall performance.

5.3 Additional Ablations and Discussions

We observe that the performance gradually degrades as the union dataset size decreases, [9, Appendix G.2]. We also find that the learned cost model is able to successfully recover the true cost function in most environments, [9, Appendix G.4]. Moreover, OSIL is robust to label noise in the non-preferred dataset $\mathcal{D}_N$, i.e., when

some trajectories in $\mathcal{D}_N$ do not correspond to high-cost behaviors, [9, Appendix G.5]. Overall, OSIL learns safer and more robust policies compared to the baselines.

6 CONCLUSION

This work introduces an offline safe imitation learning algorithm that learns a safe policy by inferring safety from non-preferred trajectories. We formulate the safe policy learning as a CMDP. Since we do not have access to reward and cost information, we reformulate the CMDP problem by deriving a lower bound of the reward maximizing objective and approximating the safety constraints by learning a cost model that estimates the likelihood of non-preferred behavior. We empirically demonstrate that our method can learn safer policies in the constrained RL benchmarks. A key limitation of our work is the assumption that the union dataset $\mathcal{D}_U$ contains large number of high-return trajectories. Relaxing this requirement is an important direction for future work.

REFERENCES

[1] Ron Alterovitz, Sven Koenig, and Maxim Likhachev. 2016. Robot planning in the real world: Research challenges and opportunities. *Ai Magazine* 37, 2 (2016), 76–84.

[2] Eitan Altman. 1999. Constrained Markov Decision Processes. https://api.semanticscholar.org/CorpusID:14906227

[3] Brenna D Argall, Sonia Chernova, Manuela Veloso, and Brett Browning. 2009. A survey of robot learning from demonstration. *Robotics and Autonomous Systems* 57, 5 (2009), 469–483.

[4] Bowen Baker, Ingmar Kanitscheider, Todor Markov, Yi Wu, Glenn Powell, Bob McGrew, and Igor Mordatch. 2020. Emergent Tool Use From Multi-Agent Autocurricula. In *International Conference on Learning Representations*. https://openreview.net/forum?id=SkxpxJBKwS

[5] Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemysław Dębiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, et al. 2019. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680* (2019).

[6] Ralph Allan Bradley and Milton E. Terry. 1952. Rank Analysis of Incomplete Block Designs: I. The Method of Paired Comparisons. *Biometrika* 39 (1952), 324. https://api.semanticscholar.org/CorpusID:125209808

[7] Daniel Brown, Wonjoon Goo, Prabhat Nagarajan, and Scott Niekum. 2019. Extrapolating beyond suboptimal demonstrations via inverse reinforcement learning from observations. In *International conference on machine learning*. PMLR, 783–792.

[8] Daniel S Brown, Wonjoon Goo, and Scott Niekum. 2020. Better-than-demonstrator imitation learning via automatically-ranked demonstrations. In *Conference on robot learning*. PMLR, 330–359.

[9] Returaj Burnwal, Nirav Pravinbhai Bhatt, and Balaraman Ravindran. 2026. OSIL: Learning Offline Safe Imitation Policies with Safety Inferred from Non-preferred Trajectories. *arXiv preprint arXiv:2602.11018* (2026).

[10] Letian Chen, Rohan Paleja, and Matthew Gombolay. 2020. Learning from suboptimal demonstration via self-supervised reward regression. In *Conference on robot learning*. PMLR, 1262–1277.

[11] Daniel Dewey. 2014. Reinforcement learning and the reward engineering principle. In *2014 AAAI Spring Symposium Series*.

[12] Charles Elkan and Keith Noto. 2008. Learning classifiers from only positive and unlabeled data. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. Association for Computing Machinery, New York, NY, USA, 213–220. https://doi.org/10.1145/1401890.1401920

[13] Scott Fujimoto and Shixiang Shane Gu. 2021. A Minimalist Approach to Offline Reinforcement Learning. In *Thirty-Fifth Conference on Neural Information Processing Systems*.

[14] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. 2018. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*. PMLR, 1861–1870.

[15] Youngsoo Jang, Geon-Hyeong Kim, Jongmin Lee, Sungryull Sohn, Byoungjip Kim, Honglak Lee, and Moontae Lee. 2023. SafeDICE: Offline Safe Imitation Learning with Non-Preferred Demonstrations. In *Thirty-seventh Conference on Neural Information Processing Systems*. https://openreview.net/forum?id=toEGuA9Qfn

[16] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. *Advances in neural information processing systems* 33 (2020), 18661–18673.

[17] Geon-Hyeong Kim, Seokin Seo, Jongmin Lee, Wonseok Jeon, HyeongJoo Hwang, Hongseok Yang, and Kee-Eung Kim. 2022. DemoDICE: Offline Imitation Learning with Supplementary Imperfect Demonstrations. In *International Conference on Learning Representations*. https://openreview.net/forum?id=BrPdX1bDZkQ

[18] Ilya Kostrikov, Ofir Nachum, and Jonathan Tompson. 2020. Imitation Learning via Off-Policy Distribution Matching. In *International Conference on Learning Representations*. https://openreview.net/forum?id=Hyg-JC4FDr

[19] Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. 2020. Conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems* 33 (2020), 1179–1191.

[20] Kimin Lee, Laura Smith, Anca Dragan, and Pieter Abbeel. 2021. B-Pref: Benchmarking Preference-Based Reinforcement Learning. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*. https://openreview.net/forum?id=ps95-mkHF_

[21] Kimin Lee, Laura M. Smith, and P. Abbeel. 2021. PEBBLE: Feedback-Efficient Interactive Reinforcement Learning via Relabeling Experience and Unsupervised Pre-training. In *International Conference on Machine Learning*. https://api.semanticscholar.org/CorpusID:235377145

[22] Zuxin Liu, Zijian Guo, Haohong Lin, Yihang Yao, Jiacheng Zhu, Zhepeng Cen, Hanjiang Hu, Wenhao Yu, Tingnan Zhang, Jie Tan, and Ding Zhao. 2024. Datasets and Benchmarks for Offline Safe Reinforcement Learning. *Journal of Data-centric Machine Learning Research* (2024).

[23] Zuxin Liu, Zijian Guo, Yihang Yao, Zhepeng Cen, Wenhao Yu, Tingnan Zhang, and Ding Zhao. 2023. Constrained decision transformer for offline safe reinforcement learning. In *International Conference on Machine Learning*. PMLR, 21611–21630.

[24] Christoforos Mavrogiannis, Francesca Baldini, Allan Wang, Dapeng Zhao, Pete Trautman, Aaron Steinfeld, and Jean Oh. 2021. Core challenges of social robot navigation: A survey. *arXiv preprint arXiv:2103.05668* (2021).

[25] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *nature* 518, 7540 (2015), 529–533.

[26] Dean A Pomerleau. 1991. Efficient training of artificial neural networks for autonomous navigation. *Neural computation* 3, 1 (1991), 88–97.

[27] Rong-Jun Qin, Xingyuan Zhang, Songyi Gao, Xiong-Hui Chen, Zewen Li, Weinan Zhang, and Yang Yu. 2022. NeoRL: A near real-world benchmark for offline reinforcement learning. *Advances in Neural Information Processing Systems* 35 (2022), 24753–24765.

[28] Stefan Schaal. 1996. Learning from demonstration. *Advances in neural information processing systems* 9 (1996).

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

[30] Daniel Shin, Anca Dragan, and Daniel S. Brown. 2023. Benchmarks and Algorithms for Offline Preference-Based Reward Learning. *Transactions on Machine Learning Research* (2023). https://openreview.net/forum?id=TGuXXlbKsn

[31] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. 2016. Mastering the game of Go with deep neural networks and tree search. *Nature* 529, 7587 (2016), 484–489.

[32] Haoran Xu, Xianyuan Zhan, Honglei Yin, and Huiling Qin. 2022. Discriminator-weighted offline imitation learning from suboptimal demonstrations. In *International Conference on Machine Learning*. PMLR, 24725–24742.

[33] Konrad Żołna, Chitwan Saharia, Leonard Boussioux, David Yu-Tung Hui, Maxime Chevalier-Boisvert, Dzmitry Bahdanau, and Yoshua Bengio. 2021. Combating false negatives in adversarial imitation learning. In *2021 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 1–9.