When to Ask for Help: Proactive Interventions in Autonomous Reinforcement Learning

Annie Xie*, **Fahim Tajwar***, **Archit Sharma***, **Chelsea Finn**
Stanford University
{anniexie,tajwar93,architsh,cbfinn}@stanford.edu

Abstract

A long-term goal of reinforcement learning is to design agents that can autonomously interact and learn in the world. A critical challenge to such autonomy is the presence of irreversible states which require external assistance to recover from, such as when a robot arm has pushed an object off of a table. While standard agents require constant monitoring to decide when to intervene, we aim to design proactive agents that can request human intervention only when needed. To this end, we propose an algorithm that can efficiently learns to detect and avoid states that are irreversible, and proactively ask for help in case the agent does enter them. On a suite of continuous control environments with unknown irreversible states, we find that our algorithm exhibits both better sample- and intervention-efficiency compared to existing methods.

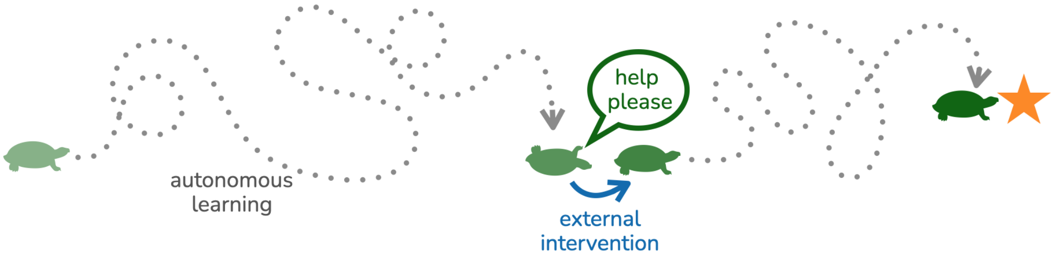

Figure 1: Autonomous agents struggle to make progress without external interventions when they are stuck in an irreversible state. Reinforcement learning agents therefore need active monitoring throughout training to detect and intervene when the agent reaches an irreversible state. Enabling the agents to detect irreversible states and proactively request for help can substantially reduce the human monitoring required for training agents.

1 Introduction

A reinforcement learning (RL) agent should be able to autonomously learn behavior by exploring in and interacting with its environment. However, in most realistic learning environments, there are irreversible states from which the agent cannot recover on its own. For example, a robot arm can inadvertently push an object off the table, such that an external supervisor must return it back to the robot's workspace to continue the learning process. Current agents demand constant monitoring to decide when the agent enters an irreversible state and therefore when to intervene. In this work, we aim to build greater autonomy into RL agents by addressing this problem. In particular, we envision proactive agents that can instead detect irreversible states, proactively request interventions when needed, and otherwise learn autonomously.

[*] Equal contribution

Decision Awareness in Reinforcement Learning Workshop at the 39th International Conference on Machine Learning (ICML), Baltimore, Maryland, USA, 2022. Copyright 2022 by the author(s).

While prior work has studied autonomy in RL with the objective of minimizing the number of human-provided resets at the end of each episode, previous setups generally assume the environment is fully reversible [19, 51, 35, 34, 17]. A related desiderata, however, arises in the safe RL setting; safe RL methods aim to learn policies that minimize visits to unsafe states, and the developed approaches are designed to avoid those particular parts of the state space [1, 10, 39, 37, 43, 41]. Prior safe RL algorithms assume that agent is given a safety label on demand for *every* state it visits. In contrast, an autonomous agent may not know when it has reached an irreversible state (such as knocking an important object off a table), and an algorithm in this setting should instead learn to both detect and avoid such states, while minimizing queries about whether a state is reversible.

We can reduce the labeling requirement with a simple observation: all states *proceeding* an irreversible state are irreversible, and all states *preceding* a reversible state will be reversible. Based on this observation, we design a deterministic scheme based on binary search to generate reversibility labels for a trajectory of length T using at most $\mathcal{O}(\log T)$ label queries, compared to the $\mathcal{O}(T)$ queries made by safe RL methods. We further reduce labeling burden by only querying labels in a large batch at the end of each extended episode, i.e. typically only after tens of thousands of steps. By combining this label efficient scheme with proactive requests for an intervention and batch of labels, we can enable agents to learn amidst irreversible states with a high degree of autonomy.

Concretely, we propose a framework for reversibility-aware autonomous RL, which we call *proactive agent interventions (PAINT)*, that aims to minimize human monitoring and supervision required throughout training. First, we train a reversibility-aware Q-value function that penalizes visits to irreversible states. Second, the reversibility labels are generated by our proposed label-efficient binary search routine, which makes at most logarithmic number of queries in the length of the interaction with the environment. Finally, the labeled states can be used to learn a classifier for predicting irreversible states, which can then be leveraged to proactively call for interventions. Our proposed framework PAINT can be used to adapt any value-based RL algorithm, in both episodic and non-episodic settings, to learn with minimal and proactive interventions. We compare PAINT to prior methods for autonomous RL and safe RL on a suite of continuous control tasks, and find that PAINT exhibits both better sample- and intervention-efficiency compared to existing methods. On challenging autonomous object manipulation tasks, PAINT only requires around 100 interventions while training for 3 million steps, which is upto $15\times$ fewer than those required by prior algorithms.

2 Related Work

Deployment of many RL algorithms in physical contexts is challenging, because they fail to avoid undesirable states in the environment and require human-provided resets between trials. Safe RL, reversibility-aware RL, and autonomous RL, which we review next, address parts of these problems.

Safe RL. The goal of our work is to learn to avoid irreversible states. Algorithms for safe RL also need to avoid regions of the state space, and achieve this by formulating a constrained optimization problem [10, 39, 37, 48] or by assigning low rewards to unsafe states [43, 41]. Another class of algorithms construct shielding-based policies that yield control to a backup policy if following the learning policy leads to an unsafe state [1, 3, 42, 40, 7, 43, 6]. Critically, however, all of these approaches assume that safety labels for each state can be queried freely at every time-step of training, whereas our objective is to minimize labeling requirements over training.

Reversibility-aware RL. Reversibility and reachability have been studied in the context of RL to avoid actions that lead to irreversible states [24, 23, 31, 15] or, conversely, to guide exploration towards difficult-to-reach states [33, 5]. Unlike prior work, our study of reversibility primarily focuses on the non-episodic setting to minimize the number of human interventions during learning. While prior methods are self-supervised, our experiments also find that our algorithm learns with significantly fewer interventions than prior methods by leveraging some binary reversibility labels.

Autonomous RL. Multiple prior works have also studied autonomy in RL, motivated by the fact that deployments of RL algorithms on real robots often require human-provided resets between episodes [13, 16, 14]. To avoid the supervision needed for episodic resets, prior work has proposed to learn controllers to return to specific state distributions, such as the initial state [19, 11], the uniform distribution over states [51] or demonstration states [36], adversarially learned distributions [47], or curriculum-based distributions [34]. However, most work in the reset-free setting assumes the agent's environment is reversible [30, 35, 34, 17], whereas we specifically tackle the setting where this is not the case. One notable exception is the Leave No Trace algorithm [11], which checks whether the agent has successfully returned back to the initial state distribution and requests an intervention otherwise. Our approach differs from Leave No trace by requesting a reset based on the estimated reversibility of the state, which we find requires significantly fewer interventions in our evaluation.

Human-in-the-loop learning. Learning from human feedback has enabled RL agents to acquire complex skills that are difficult to encode in a reward function [22, 27, 45, 12, 4]. However, interactive RL algorithms are often to difficult to scale as they rely on feedback at every time-step of training. More feedback-efficient algorithms have learned reward models from human-provided preferences [2, 38, 46, 32, 8, 26, 44], which removes the need for constant feedback. Similarly, the interactive imitation learning learning has seen more query-efficient algorithms, which query expert actions based on the estimated risk or novelty of a visited state [49, 28, 21, 20]. While these algorithms augment the agent with human-provided preferences or expert actions, our approach leverages a different mode of feedback, that is, reversibility labels for visited states.

3 Reinforcement Learning in Irreversible Environments

Consider a Markov decision process $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, r, \rho_0, \gamma)$ with state space $\mathcal{S}$, action space $\mathcal{A}$, transition dynamics $\mathcal{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \mapsto [0, 1]$, bounded reward function $r : \mathcal{S} \times \mathcal{A} \mapsto [R_{\text{min}}, R_{\text{max}}]$, initial state distribution $\rho_0 : \mathcal{S} \mapsto [0, 1]$ and discount factor $\gamma \in [0, 1)$. In this work, we build on the formalism of autonomous RL [35], but we remove the assumption that the environment is reversible, i.e., the MDP is no longer strongly connected (see example in [25, Chapter 38] and description below). The environment is initialized at $s_0 \sim \rho$ and an algorithm continually interacts with the environment till it requests the environment to be reset via an external intervention to state $s'_0 \sim \rho$. Specifically, an algorithm $\mathbb{A} : \{s_i, a_i, s_{i+1}, r_i\}_{i=0}^{t-1} \mapsto (a_t, \pi_t)$ generates a sequence $(s_0, a_0, s_1, \ldots)$ in $\mathcal{M}$, mapping the states, actions, and rewards seen till time $t-1$ to an action $a_t \in \mathcal{A} \cup \{a_{\texttt{reset}}\}$, and the current guess at the optimal policy $\pi_t : \mathcal{S} \times \mathcal{A} \mapsto [0, \infty)$. Here, $a_{\texttt{reset}}$ is a special action the agent can execute to reset the environment through extrinsic interventions, i.e. $\mathcal{P}(\cdot \mid s, a_{\texttt{reset}}) = \rho_0(\cdot)$.

A MDP is strongly connected if for all pairs of states $s_i, s_j \in \mathcal{S}$, there exists a policy π such that s_j has a non-zero probability of being visited when executing the policy π from state s_i. This assumption can easily be violated in practice, for example, when a robot arm pushes an object out of its reach. At an abstract level, the agent has transitioned into a component of MDP that is not connected with the high reward states, and thus cannot continue making progress, as visualized in Figure 2. The agent can invoke an extrinsic agent (such as human) through $a_{\texttt{reset}}$, and the extrinsic agent can reset the environment to a state from the initial state distribution. For example, the human supervisor can reset the object to the initial state, which is within the reach of the robot arm. For every state $s \in \mathcal{S}$, define $\mathcal{R}_\rho : \mathcal{S} \mapsto \{0, 1\}$ as the indicator whether the state s is in the same component as the initial state distribution. State s is defined to be reversible if $\mathcal{R}_\rho(s) = 1$, and irreversible if $\mathcal{R}_\rho(s) = 0$. We assume that the $\mathcal{R}_\rho$ is unknown, but can be queried for a state s.

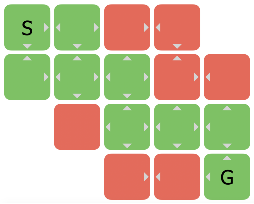

Figure 2: Example of an MDP with irreversible states (in red). The agent starts in the state 'S' and its goal is to reach the state 'G', which are connected.

While we do not assume that the MDP is strongly connected, we assume that the states visited by the optimal policy are in the same connected component as the initial state distribution. Under this assumption, we can design agents that can autonomously practice the task many times. Otherwise, the environment would need to be reset after every successful trial of the task.

The objective is to learn an optimal policy $\pi^* \in \arg\max_\pi J(\pi) = \arg\max_\pi \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)]$. Note, $J(\pi)$ is approximated by computing the return when the policy is rolled out from $s_0 \sim \rho_0$. Algorithms are typically evaluated on the sample efficiency, that is minimizing $\mathbb{D}(\mathbb{A}) = \sum_{t=0}^{\infty} J(\pi^*) - J(\pi_t)$. However, since we care about minimizing the human supervision required and resetting the environment can entail expensive human supervision, we will primarily evaluate algorithms on *intervention-efficiency*, defined as $\mathbb{I}(\mathbb{A}) = \sum_{k=0}^{\infty} J(\pi^*) - J(\pi_k)$, where π_k is the policy learned after k interventions.

4 Preliminaries

Episodic settings reset the environment to a state from the initial state distribution after every trial, typically after every few hundred steps of interaction with the environment. Such frequent resetting of the environment entails an extensive amount of external interventions, typically from a human. Prior works on autonomous RL have sought to reduce the supervision required for resetting the environments by learning a backward policy that resets the environment [11, 50, 34]. Meaningfully improving the autonomy of RL in irreversible environments requires us to curb the requirement of episodic resets first. While our proposed framework is compatible with any autonomous RL algorithm, we describe MEDAL [36], which will be used in our experiments.

MEDAL learns a forward policy π_f and a backward policy π_b, alternately executed for a fixed number of steps in the environment. The forward policy maximizes the conventional cumulative task reward, that is $\mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)\right]$, and the backward policy minimizes the Jensen-Shannon divergence $\mathcal{D}_{\text{JS}}\left(\rho^b(s) \mid\mid \hat{\rho}^*(s)\right)$ between the marginal state distribution of the backward policy ρ^b and the state distribution of the optimal forward policy $\hat{\rho}^*$, approximated by a small number of expert demonstrations. Thus, the backward policy keeps the agent close to the demonstration states, allowing the forward agent to try the task from a mix of easy and hard initial states. The proposed divergence can be minimized via the objective $\min_{\pi_b} \max_C \mathbb{E}_{s\sim\rho^*}\big[\log C(s)\big] + \mathbb{E}_{s\sim\rho^b}\big[\log(1 - C(s))\big]$, where $C : \mathcal{S} \mapsto [0, 1]$ is a classifier that maximizes the log-probability of states visited in the forward demonstrations, and minimizes the probability of the states visited by the backward policy. The optimization problem for the backward policy can be written as a RL problem:

$$\min_{\pi_b} \mathbb{E}_{s\sim\rho^b}\left[\log\left(1 - C(s)\right)\right] = \max_{\pi_b} \mathbb{E}\left[-\sum_{t=0}^{\infty} \gamma^t \log\left(1 - C(s)\right)\right] \tag{1}$$

where π_b maximizes the reward function $r(s, a) = -\log\left(1 - C(s)\right)$. Correspondingly, $C(s)$ is trained to discriminate between states visited by the backward policy and the demonstrations.

5 Proactive Agent Interventions for Autonomous Reinforcement Learning

To minimize human monitoring and supervision when an agent is learning in an environment with irreversible states, the agent needs to (a) learn to avoid irreversible states over the course of training and (b) learn to detect and *request* an intervention whenever the agent is stuck. For the former, we first describe a simple modification to the reward function to explicitly penalize visitation of irreversible states in Section 5.1. However, such a modification requires the knowledge of reversibility of the visited states, which is not known apriori. We learn a classifier to estimate reversibility, proposing a label-efficient algorithm to query reversibility labels of visited states in Section 5.2. Since both the dynamics and the set of irreversible states are unknown apriori, the agent will inevitably still visit irreversible states as a part of the exploration. To ensure that a human does not have to monitor the agent throughout training, the agent should have a mechanism to decide and request for an intervention. We discuss such a mechanism in Section 5.3. Finally, we put together all these components in Section 5.4 for our proposed framework **<u>P</u>**roactive **<u>A</u>**gent **<u>INT</u>**erventions (PAINT), an overview of which is given in Figure 3

5.1 Penalizing Visitation of Irreversible States

Our goal is to penalize visitation of irreversible states by ensuring all actions leading to irreversible states are 'worse' than those leading to reversible states. To this end, we adapt the reward-penalty framework from safe RL [41] for learning in the presence of irreversible states. Let $\mathcal{S}_{\text{rev}} = \{(s, a) \mid (\mathcal{R}_\rho(s') = 1\}$ denote the set of state-action pairs that lead to a reversible state, where $\mathcal{R}_\rho$ indicates whether a state is reversible. For a transition (s, a, s'), consider a surrogate reward function $\tilde{r}$:

$$\tilde{r}(s, a) = \begin{cases} r(s, a), & (s, a) \in \mathcal{S}_{\text{rev}} \\ R_{\text{min}} - \epsilon, & (s, a) \notin \mathcal{S}_{\text{rev}} \end{cases} \tag{2}$$

Whenever the next state s' is a reversible state, the agent gets the environment reward. Otherwise if it has entered an irreversible, it gets a constant reward $R_{\text{min}} - \epsilon$ that is worse than any reward given out by the environment. Whenever an agent enters an irreversible state, it will continue to remain in an

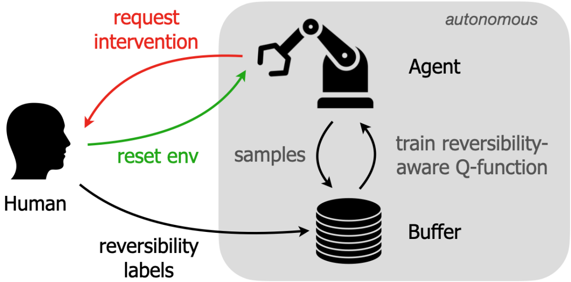

Figure 3: Overview of our framework PAINT for minimizing human monitoring and supervision when learning in the presence of irreversible states. The agent proactively requests interventions, freeing the human from active monitoring of training. When an intervention is requested, the human resets the environment and provides reversibility labels for the latest experience since the previous intervention.

Algorithm 1: Reversibility Labeling via Binary Search

```
input: $\tau = \{s_i\}_{i=0}^{T}$; // unlabeled trajectory
while $len(\tau) > 0$ do
    $m \leftarrow \lfloor \text{len}(\tau)/2 \rfloor$; // get midpoint
    // query for midpoint
    if $\mathcal{R}_\rho(s_m) = 1$ then
        // label first half reversible and query for second half
        label $\{s_i\}_{i=0}^{m}$ as 1;
        $\tau \leftarrow \{s_i\}_{i=m+1}^{\text{len}(\tau)}$;
    else
        // label second half irreversible and query for first half
        label $\{s_i\}_{i=m+1}^{\text{len}(\tau)}$ as 0;
        $\tau \leftarrow \{s_i\}_{i=0}^{m}$;
```

irreversible state and get a constant reward of $R_{\text{min}} - \epsilon$. Therefore, for any $(s, a) \notin \mathcal{S}_{\text{rev}}$, the Q-value can be computed directly as:

$$Q^\pi(s,a) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \tilde{r}(s_t, a_t) \mid s_0 = s, a_0 = a\right] = (R_{\text{min}} - \epsilon)\sum_{t=0}^{\infty}\gamma^t = \frac{R_{\text{min}} - \epsilon}{1-\gamma}$$

This observation allows us to bypass the need to perform Bellman backups on irreversible states, and instead directly regress to the Q-value. More specifically, we can rewrite the loss function for the Q-value function as, $\ell(Q) = \mathbb{E}_{(s,a,s',r)\sim\mathcal{D}}\left[Q(s,a) - \mathcal{B}^\pi Q(s,a)\right]$, where the application of Bellman backup operator $\mathcal{B}^\pi Q(s,a)$ can be expanded as:

$$\mathcal{B}^\pi Q(s,a) = \begin{cases} r(s,a) + \gamma \mathbb{E}_{a'\sim\pi(\cdot|s')}\hat{Q}(s',a'), & \mathcal{R}_\rho(s') = 1 \\ (R_{\text{min}} - \epsilon) \,/\, (1-\gamma)\,, & \mathcal{R}_\rho(s') = 0 \end{cases} \tag{3}$$

$$= \mathcal{R}_\rho(s')\left(r(s,a) + \gamma \mathbb{E}_{a'\sim\pi(\cdot|s')}\hat{Q}(s',a')\right) + (1 - \mathcal{R}_\rho(s'))\frac{R_{\text{min}} - \epsilon}{1-\gamma} \tag{4}$$

Here, $\mathcal{D}$ denotes the replay buffer, $\hat{Q}$ denotes the use of target networks commonly used in Q-learning algorithms to stabilize training when using neural networks as function approximators [29]. This surrogate reward function and the modified Bellman operator can be used for any value-based RL algorithm in both episodic and autonomous RL settings. The hyperparameter ϵ controls how aggressively the agent is penalized for visiting irreversible states. The following theorem shows that the surrogate reward function induces desirable behavior for $\epsilon > 0$:

Theorem 5.1. *Let $(s,a) \in \mathcal{S}_{rev}$ denote a state-action leading to a reversible state and let $(s\_, a\_) \notin \mathcal{S}_{rev}$ denote a state-action pair leading to an irreversible state. Then, for all such pairs*

$$Q^\pi(s,a) > Q^\pi(s\_, a\_)$$

for all $\epsilon > 0$ and for all policies π.

The proof can be found in Appendix A.1. The result guarantees that actions leading to irreversible states indeed have worse Q-values than those keeping the agent amongst the reversible states, encouraging policies to visit irreversible states fewer times over the course of training. Additionally, any $\epsilon > 0$ should suffice in theory, though practical considerations for optimization affect this choice.

5.2 Estimating Reversibility

In general, $\mathcal{R}_\rho$ is not known apriori and will have to be estimated. We define $\hat{\mathcal{R}}_\rho : \mathcal{S} \mapsto [0,1]$ as an estimator of the reversibility of a state $s \in \mathcal{S}$. We can then define an empirical Bellman backup operator $\hat{\mathcal{B}}^\pi$ from equation 4 by replacing $\mathcal{R}_\rho$ with the estimator $\hat{\mathcal{R}}_\rho$. We propose to fit a classifier by minimizing the binary cross-entropy loss

$\ell(\hat{\mathcal{R}}_\rho) = -\mathbb{E}_{s\sim\mathcal{D}}\big[\mathcal{R}_\rho(s)\log\hat{\mathcal{R}}_\rho(s) + (1-\mathcal{R}_\rho(s))\log(1-\hat{\mathcal{R}}_\rho(s))\big]$, where the states $s \sim \mathcal{D}$ represent the states visited by the agent. Minimizing the loss requires the reversibility labels $\mathcal{R}_\rho(s)$ for $s \sim \mathcal{D}$. Since labeling requires supervision, it is critical to query $\mathcal{R}_\rho$ efficiently.

Given a trajectory of states $\tau = (s_0, s_1, \dots s_T)$, a naïve approach would be to query the labels $\mathcal{R}_\rho(s_i)$ for all states s_i, leading to $\mathcal{O}(T)$ queries per trajectory. However, observe that we have the following properties: (a) all states *following* an irreversible state will be irreversible and (b) all states *preceding* a reversible state will be reversible. It follows from these properties that every trajectory can be split into a reversible segment $\tau_r = (s_0, s_1, \dots s_k)$ and an irreversible segment $\tau_{\sim r} = (s_{k+1}, \dots s_T)$, where the irreversible segment $\tau_{\sim r}$ can be empty potentially. Identifying s_{k+1}, the first irreversible state, generates the labels for the entire trajectory automatically. Fortunately, we can construct a scheme based on binary search to identify s_{k+1} in $\mathcal{O}(\log T)$ queries: s_{k+1} occurs after the midpoint of the trajectory if the midpoint is reversible, otherwise it occurs before it. The pseudocode for this binary search inspired routine is given in Algorithm 1.

The total number of labels required would be $\mathcal{O}\left(N \log |\tau|_{\text{max}}\right)$, where N is the number of trajectories in the replay buffer $\mathcal{D}$ and the $|\tau|_{\text{max}}$ denotes the maximum length of the trajectory. This represents a reduction in label requirement of $\mathcal{O}\left(N\, |\tau|_{\text{max}}\right)$ by prior safe RL methods. Since the agent trains to avoid irreversible states, there will be fewer and longer trajectories as training progresses. Thus, labeling reduces over time because the labels required is linear in N and logarithmic in $|\tau|_{\text{max}}$.

5.3 Proactive Interventions

Despite trying to avoid irreversible states via reward penalties, an agent will inevitably encounter some irreversible states due to exploratory behaviors. It is critical that the agent proactively asks for help in such situations, rather than requiring a human to constantly monitor the training process. An agent should request an intervention when it is an irreversible state. Since $\mathcal{R}_\rho(s)$ is not available, the agent again needs to estimate the reversibility of the state. It is natural to reuse the learned reversibility estimator $\hat{\mathcal{R}}_\rho$ to proactively request interventions. We propose the following rule: the agent executes $a_{\texttt{reset}}$ whenever the reversibility classifier's prediction falls below 0.5, i.e., $\hat{\mathcal{R}}_\rho(s) < 0.5$.

5.4 Putting it Together

With the key components in place, we summarize our proposed framework. High-level pseudocode is given in Alg. 2, and a more detailed pseudocode is deferred to Appendix A.2.

PAINT can modify any value-based RL algorithm, in both episodic and autonomous settings. This description and Alg. 2 focus on the latter setting, although adapting it to the episodic setting is straightforward. The agent's interaction with the environment consists of a sequence of trials that end whenever the environment is reset to a state $s \sim \rho_0$. During each trial, the agent operates autonomously, and the Bellman update for the critic is modified according to the empirical Bellman backup $\hat{\mathcal{B}}^\pi$. Whenever the reversibility classifier $\hat{\mathcal{R}}_\rho < 0.5$, the agent requests an intervention. The agent can execute a fixed number of exploration steps after requesting an intervention and before the intervention is performed. Whenever the classifier predicts an irreversible state correctly, these exploration steps can help the agent gather more information about irreversible states. At the time of the intervention, all new states visited since the previous intervention are labeled for reversibility via Algorithm 1. Finally, the reversibility classifier is trained on all the labeled data before the environment is reset to a state $s \sim \rho_0$ for the next trial.

Algorithm 2: PAINT

```
input: $\mathbb{P}$; // agent, params abstracted away
initialize $\hat{\mathcal{R}}_\rho, \mathcal{D}$; // rev classifier, replay buffer
while not done do
    $s \sim \rho_0$; // reset environment
    // continue till classifier detects irreversibility
    while $\hat{\mathcal{R}}_\rho(s) > 0.5$ do
        // step in the environment
        $a \sim \mathbb{P}(s)$, $s \sim \mathcal{P}(\cdot \mid s, a)$;
        // update replay buffer and agent
        update $\mathcal{D}, \mathbb{P}$;
    // optionally explore environment
    for explore steps do
        $a \sim \text{unif}(\mathcal{A})$, $s \sim \mathcal{P}(\cdot \mid s, a)$;
        update $\mathcal{D}$;
    // reversibility labels via binary search
    update reversibility labels in $\mathcal{D}$;
    // train classifier on all labeled data, new and old
    train $\hat{\mathcal{R}}_\rho$;
```

The agent is provided reversibility labels only when the external reset is provided. This simplifies supervision as the human can reset the environment and provide labels at the same time. This means

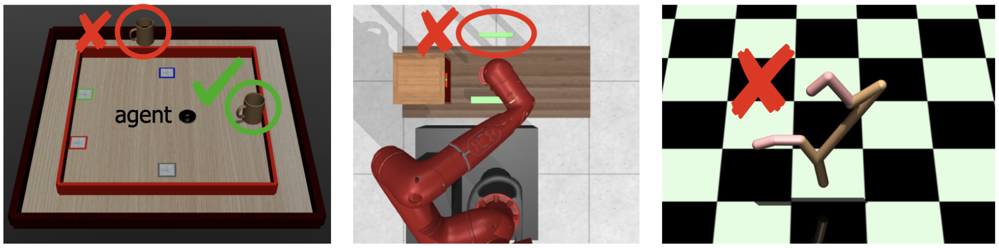

Figure 4: A subset of our evaluation tasks: Tabletop Manipulation, Peg Insertion, and Half-Cheetah Velocity. Irreversible states in the first two environments are when the agent drops the object outside the red boundary (*left*) and off of the table (*middle*). Cheetah is in an irreversible state whenever it is flipped over (*right*).

the replay buffer $\mathcal{D}$ will contain states with and without reversibility labels, since states from the current trial will not yet have labels. We use Eq. 4 for states that have reversibility labels to avoid errors from the classifier affecting the critic update and use $\hat{\mathcal{B}}^{\pi}$ for those that do not have labels.

6 Experiments

We design several experiments to study the efficiency of our algorithm in terms of the required number of reset interventions and number of queried reversibility labels. Videos of our results are at: `https://sites.google.com/view/proactive-interventions`.

6.1 Experimental Setup

Environments. To illustrate the wide applicability of our method, we design environments that represent three distinct RL setups: episodic, forward-backward, and continuing.

- **Maze** (episodic). A 2D continuous maze environment with trenches, which represent groups of connected irreversible states. The agent can fall into a trench, and once entered, it can roam freely within the trench but cannot leave it without an environment reset.
- **Tabletop Organization** [35] (forward-backward). The agent must grasp the mug and put it down on one of the four goal positions. Dropping the mug outside of the red boundary is irreversible.
- **Peg Insertion** [35] (forward-backward). The agent must insert the peg into the goal but can potentially drop it off the table, which is irreversible.
- **Half-Cheetah Vel** [9] (continuing). The agent must run at the specified target velocity, which changes every 500 steps, and can potentially flip over onto its back, which is irreversible.

We visualize and fully describe each environment in Fig. 4 and in Appendix A.3 respectively.

Comparisons. In the episodic and continuing settings, we consider safe RL baselines that rely on reversibility labels at every time-step of training.

- **Safe Model-Based Policy Optimization (SMBPO)** [41]. This comparison implements the modified Bellman operator defined in Eqn. 4 in Section 5.1, using the true reversibility labels.
- **Safety Q-functions for RL (SQRL)** [37]. A safe RL method that trains a safety critic, which estimates the future probability of entering an irreversible state for a safety-constrained policy.

In the forward-backward setting, we consider methods designed for the autonomous learning setup. These methods do not require any reversibility labels. Hence, our goal here is to compare the reset-efficiency of our method to prior work.

- **Leave No Trace (LNT)** [11]. An autonomous RL method that jointly trains a forward policy and reset policy. When the reset policy fails to return to the initial state, the agent requests a reset.
- **Matching Expert Distributions for Autonomous Learning (MEDAL)** [36]. This method trains a reset policy that returns to the distribution of demonstration states provided for the forward policy. MEDAL does not have a built-in intervention rule.

In all tasks, we compare to a recently proposed reversibility-aware RL method, **Reversibility-Aware Exploration (RAE)** [15], which does not require any reversibility labels. It instead trains a self-supervised reversibility estimator to predict whether a state transition $(s, \tilde{s})$ is more likely than the reverse $(\tilde{s}, s)$. We augment RAE with an intervention rule, similar to our method, defined in terms

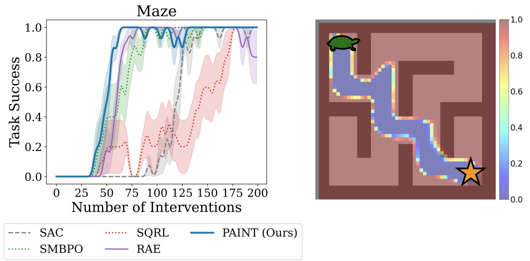

Figure 5: (*left*) Task success versus interventions. Shaded regions denote the standard error over 5 seeds. (*right*) Predictions generated by our reversibility classifier, where the purple region is predicted to be reversible.

Task	Method	Labels
Maze	SMBPO/SQRL PAINT (Ours)	200K 3260 ± 12
Tabletop	PAINT (Ours)	1021 ± 69
Peg Insertion	PAINT (Ours)	2083 ± 149
Cheetah	SMBPO w Term. PAINT (Ours)	3M 8748 ± 3762

Figure 6: Number of queried reversibility labels. For our method, we average the number of labels used across 5 seeds and report the standard error.

of predictions from its self-supervised classifier. In the forward-backward setting, we train both the forward and backward policies with RAE. Finally, we also evaluate **Episodic RL**, which represents the typical RL setup with frequent resets and thus provides an upper-bound on task success. In Appendix A.4, we provide full implementation details of each comparison.

6.2 Main Results

In Fig. 5 (*left*) and Fig. 7, we plot the task success versus the number of interventions in the 4 tasks. For methods that use reversibility labels, we report the total number of labels queried in Fig. 6.

Maze. In the Maze task, resets are provided to each agent every 500 time-steps. While the safe RL methods, SMBPO and SQRL, require reversibility labels at every time-step, our approach PAINT only requires on average 3260 queries to label all 200K states visited. In Fig. 5 (right), we visualize predictions from our reversibility classifier at the end of training, where zero predicts 'reversible' and one predicts 'irreversible'. The classifier correctly identifies the path that leads to the goal as reversible. Interestingly, it classifies all other regions as stuck states, including the states that *are* reversible. Because these states are irrelevant to the task, however, classifying them as irreversible, and therefore to be avoided, is advantageous to our policy as it reduces its area of exploration.

More complex domains. In the Tabletop Organization and Peg Insertion tasks, each agent is reset every 200K and 100K time-steps, per the EARL benchmark [35]. However, we allow agents to request earlier resets, and under this setting, we compare PAINT to other methods that implement intervention rules. Compared to Leave No Trace and Reversibility-Aware Exploration, PAINT requires significantly fewer resets—80 and 124 resets respectively, which corresponds to roughly **one intervention for every 25K steps**. Importantly, the number of interventions plateaus as training progresses, and the agent requires fewer and fewer resets over time (see Appendix A.5 for additional plots of number of interventions versus time-steps). The exception is MEDAL (green segment near the origin), which is not equipped with an early termination rule and so only uses 10 interventions total. However, it also fails to make meaningful progress on the task with few resets.

On the continuing Half-Cheetah task, agents do not receive any resets, unless specifically requested. Here, we compare PAINT to SMPBO with early termination, an **oracle** version of our method, which assumes that reversibility labels are available at *every* time-step and immediately requests an intervention if the agent is flipped over. PAINT converges to its final performance after around 750 resets, on par with the number of resets required by SMPBO with early termination. On the other hand, a standard episodic RL agent, which receives resets at every 2K steps, and RAE, which trains a self-supervised classifier, learn significantly slower with respect to the number of interventions.

6.3 Ablations and Sensitivity Analysis

Early termination. In the episodic Maze setting, our algorithm switches to a uniform-random policy for the remainder of the episode if the termination condition is met. In Fig. 8 (*left*), we plot the performance without early termination, i.e., running the agent policy for the full episode. Taking random explorations, after the agent believes it has entered an irreversible state, significantly helps our method, as it increases the number and diversity of irreversible states the agent has seen.

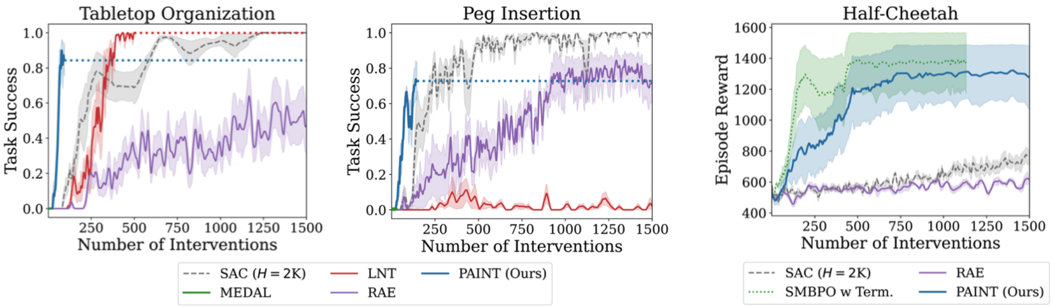

Figure 7: Task success versus interventions averaged over 5 seeds. Methods with stronger assumptions, i.e., SAC resets every H steps and SMBPO requires labels at every time-step, are dotted.

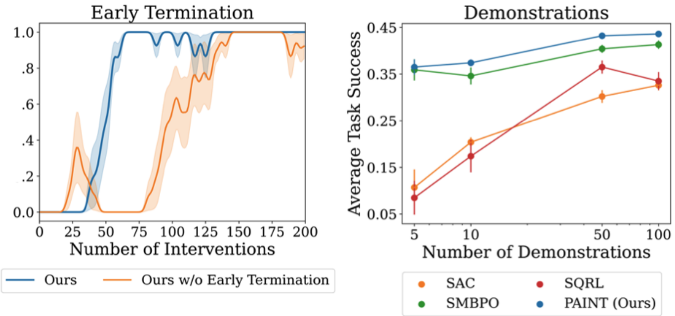

Figure 8: (*left*) After removing the early termination condition, which initiates random exploration, we find that PAINT learns less efficiently. (*right*) Varying the number of demonstrations suggests that PAINT and SMBPO are robust to the amount of available demonstrations.

Varying the number of demonstrations. Our method leverages demonstrations in a subset of environments. While we provide these demonstrations to all comparisons as well, we want to study how much our method relies on them. We plot the average task success during training versus number of demonstrations in Fig. 8 (*right*). While PAINT and SMBPO are robust to the amount, alternative methods tend to achieve significantly lower success when given fewer demonstrations.

We study additional ablations on the termination condition and pseudo-labeling in Appendix A.5.

7 Discussion

In this work, we sought to build greater autonomy into RL agents, particularly in irreversible environments. We proposed an algorithm, PAINT, that learns to detect and avoid irreversible states, and proactively requests an intervention when in an irreversible state. PAINT leverages reversibility labels to learn to identify irreversible states more quickly, and improves upon existing methods on a range of learning setups in terms of task success, reset-efficiency, and label-efficiency.

Despite these improvements, PAINT has multiple important limitations. In environments where irreversible states are not encountered until further into training, the reversibility classifier may produce false positives which would significantly delay the next intervention. Further, while PAINT is far more label-efficient than prior safe RL methods, it still requires around thousands of reversibility labels. We expect that this limitation may be mitigated with more sophisticated querying strategies, e.g. that take into account the classifier's confidence. Finally, we hope that future work can validate the ability for reversibility aware techniques to improve the autonomy of real robotic learning systems.

References

[1] Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *International conference on machine learning*, pages 22–31. PMLR, 2017.

[2] Riad Akrour, Marc Schoenauer, and Michele Sebag. Preference-based policy learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 12–27. Springer, 2011.

[3] Mohammed Alshiekh, Roderick Bloem, Rüdiger Ehlers, Bettina Könighofer, Scott Niekum, and Ufuk Topcu. Safe reinforcement learning via shielding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[4] Christian Arzate Cruz and Takeo Igarashi. A survey on interactive reinforcement learning: design principles and open challenges. In *Proceedings of the 2020 ACM designing interactive systems conference*, pages 1195–1209, 2020.

[5] Adrià Puigdomènech Badia, Pablo Sprechmann, Alex Vitvitskyi, Daniel Guo, Bilal Piot, Steven Kapturowski, Olivier Tieleman, Martín Arjovsky, Alexander Pritzel, Andew Bolt, et al. Never give up: Learning directed exploration strategies. *arXiv preprint arXiv:2002.06038*, 2020.

[6] Osbert Bastani, Shuo Li, and Anton Xu. Safe reinforcement learning via statistical model predictive shielding. In *Robotics: Science and Systems*, 2021.

[7] Homanga Bharadhwaj, Aviral Kumar, Nicholas Rhinehart, Sergey Levine, Florian Shkurti, and Animesh Garg. Conservative safety critics for exploration. *arXiv preprint arXiv:2010.14497*, 2020.

[8] Erdem Biyik and Dorsa Sadigh. Batch active preference-based learning of reward functions. In *Conference on robot learning*, pages 519–528. PMLR, 2018.

[9] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

[10] Yinlam Chow, Mohammad Ghavamzadeh, Lucas Janson, and Marco Pavone. Risk-constrained reinforcement learning with percentile risk criteria. *The Journal of Machine Learning Research*, 18(1):6070–6120, 2017.

[11] Benjamin Eysenbach, Shixiang Gu, Julian Ibarz, and Sergey Levine. Leave no trace: Learning to reset for safe and autonomous reinforcement learning. *arXiv preprint arXiv:1711.06782*, 2017.

[12] Taylor A Kessler Faulkner, Elaine Schaertl Short, and Andrea L Thomaz. Interactive reinforcement learning with inaccurate feedback. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7498–7504. IEEE, 2020.

[13] Chelsea Finn, Xin Yu Tan, Yan Duan, Trevor Darrell, Sergey Levine, and Pieter Abbeel. Deep spatial autoencoders for visuomotor learning. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 512–519. IEEE, 2016.

[14] Ali Ghadirzadeh, Atsuto Maki, Danica Kragic, and Mårten Björkman. Deep predictive policy training using reinforcement learning. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2351–2358. IEEE, 2017.

[15] Nathan Grinsztajn, Johan Ferret, Olivier Pietquin, Matthieu Geist, et al. There is no turning back: A self-supervised approach for reversibility-aware reinforcement learning. *Advances in Neural Information Processing Systems*, 34, 2021.

[16] Shixiang Gu, Ethan Holly, Timothy Lillicrap, and Sergey Levine. Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 3389–3396. IEEE, 2017.

[17] Abhishek Gupta, Justin Yu, Tony Zhao, Vikash Kumar, Aaron Rovinsky, Kelvin Xu, Thomas Devlin, and Sergey Levine. Reset-free reinforcement learning via multi-task learning: Learning dexterous manipulation behaviors without human intervention. *ArXiv*, abs/2104.11203, 2021.

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

[19] Weiqiao Han, Sergey Levine, and Pieter Abbeel. Learning compound multi-step controllers under unknown dynamics. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6435–6442. IEEE, 2015.

[20] Ryan Hoque, Ashwin Balakrishna, Ellen Novoseller, Albert Wilcox, Daniel S Brown, and Ken Goldberg. Thriftydagger: Budget-aware novelty and risk gating for interactive imitation learning. *arXiv preprint arXiv:2109.08273*, 2021.

[21] Ryan Hoque, Ashwin Balakrishna, Carl Putterman, Michael Luo, Daniel S Brown, Daniel Seita, Brijen Thananjeyan, Ellen Novoseller, and Ken Goldberg. Lazydagger: Reducing context switching in interactive imitation learning. In *2021 IEEE 17th International Conference on Automation Science and Engineering (CASE)*, pages 502–509. IEEE, 2021.

[22] W Bradley Knox and Peter Stone. Interactively shaping agents via human reinforcement: The tamer framework. In *Proceedings of the fifth international conference on Knowledge capture*, pages 9–16, 2009.

[23] Victoria Krakovna, Laurent Orseau, Ramana Kumar, Miljan Martic, and Shane Legg. Penalizing side effects using stepwise relative reachability. *arXiv preprint arXiv:1806.01186*, 2018.

[24] Maarja Kruusmaa, Yuri Gavshin, and Adam Eppendahl. Don't do things you can't undo: reversibility models for generating safe behaviours. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pages 1134–1139. IEEE, 2007.

[25] Tor Lattimore and Csaba Szepesvári. *Bandit algorithms*. Cambridge University Press, 2020.

[26] Kimin Lee, Laura Smith, and Pieter Abbeel. Pebble: Feedback-efficient interactive reinforcement learning via relabeling experience and unsupervised pre-training. *arXiv preprint arXiv:2106.05091*, 2021.

[27] James MacGlashan, Mark K Ho, Robert Loftin, Bei Peng, Guan Wang, David L Roberts, Matthew E Taylor, and Michael L Littman. Interactive learning from policy-dependent human feedback. In *International Conference on Machine Learning*, pages 2285–2294. PMLR, 2017.

[28] Kunal Menda, Katherine Driggs-Campbell, and Mykel J Kochenderfer. Ensembledagger: A bayesian approach to safe imitation learning. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5041–5048. IEEE, 2019.

[29] 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. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

[30] Teodor Mihai Moldovan and Pieter Abbeel. Safe exploration in markov decision processes. *arXiv preprint arXiv:1205.4810*, 2012.

[31] Nasim Rahaman, Steffen Wolf, Anirudh Goyal, Roman Remme, and Yoshua Bengio. Learning the arrow of time for problems in reinforcement learning. 2020.

[32] Dorsa Sadigh, Anca D Dragan, Shankar Sastry, and Sanjit A Seshia. *Active preference-based learning of reward functions*. 2017.

[33] Nikolay Savinov, Anton Raichuk, Raphaël Marinier, Damien Vincent, Marc Pollefeys, Timothy Lillicrap, and Sylvain Gelly. Episodic curiosity through reachability. *arXiv preprint arXiv:1810.02274*, 2018.

[34] Archit Sharma, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Autonomous reinforcement learning via subgoal curricula. *Advances in Neural Information Processing Systems*, 34, 2021.

[35] Archit Sharma, Kelvin Xu, Nikhil Sardana, Abhishek Gupta, Karol Hausman, Sergey Levine, and Chelsea Finn. Autonomous reinforcement learning: Formalism and benchmarking. *arXiv preprint arXiv:2112.09605*, 2021.

[36] Archit Sharma, Rehaan Ahmad, and Chelsea Finn. A state-distribution matching approach to non-episodic reinforcement learning. *arXiv preprint arXiv:2205.05212*, 2022.

[37] Krishnan Srinivasan, Benjamin Eysenbach, Sehoon Ha, Jie Tan, and Chelsea Finn. Learning to be safe: Deep rl with a safety critic. *arXiv preprint arXiv:2010.14603*, 2020.

[38] Hiroaki Sugiyama, Toyomi Meguro, and Yasuhiro Minami. Preference-learning based inverse reinforcement learning for dialog control. In *Thirteenth Annual Conference of the International Speech Communication Association*, 2012.

[39] Chen Tessler, Daniel J Mankowitz, and Shie Mannor. Reward constrained policy optimization. *arXiv preprint arXiv:1805.11074*, 2018.

[40] Brijen Thananjeyan, Ashwin Balakrishna, Suraj Nair, Michael Luo, Krishnan Srinivasan, Minho Hwang, Joseph E Gonzalez, Julian Ibarz, Chelsea Finn, and Ken Goldberg. Recovery rl: Safe reinforcement learning with learned recovery zones. *arXiv preprint arXiv:2010.15920*, 2020.

[41] Garrett Thomas, Yuping Luo, and Tengyu Ma. Safe reinforcement learning by imagining the near future. *Advances in Neural Information Processing Systems*, 34, 2021.

[42] Matteo Turchetta, Andrey Kolobov, Shital Shah, Andreas Krause, and Alekh Agarwal. Safe reinforcement learning via curriculum induction. *Advances in Neural Information Processing Systems*, 33:12151–12162, 2020.

[43] Nolan C Wagener, Byron Boots, and Ching-An Cheng. Safe reinforcement learning using advantage-based intervention. In *International Conference on Machine Learning*, pages 10630–10640. PMLR, 2021.

[44] Xiaofei Wang, Kimin Lee, Kourosh Hakhamaneshi, Pieter Abbeel, and Michael Laskin. Skill preferences: Learning to extract and execute robotic skills from human feedback. In *Conference on Robot Learning*, pages 1259–1268. PMLR, 2022.

[45] Zhaodong Wang and Matthew E Taylor. Interactive reinforcement learning with dynamic reuse of prior knowledge from human/agent's demonstration. *arXiv preprint arXiv:1805.04493*, 2018.

[46] Christian Wirth and Johannes Fürnkranz. Preference-based reinforcement learning: A preliminary survey. In *Proceedings of the ECML/PKDD-13 Workshop on Reinforcement Learning from Generalized Feedback: Beyond Numeric Rewards*. Citeseer, 2013.

[47] Kelvin Xu, Siddharth Verma, Chelsea Finn, and Sergey Levine. Continual learning of control primitives: Skill discovery via reset-games. *ArXiv*, abs/2011.05286, 2020.

[48] Moritz A Zanger, Karam Daaboul, and J Marius Zöllner. Safe continuous control with constrained model-based policy optimization. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3512–3519. IEEE, 2021.

[49] Jiakai Zhang and Kyunghyun Cho. Query-efficient imitation learning for end-to-end autonomous driving. *arXiv preprint arXiv:1605.06450*, 2016.

[50] Henry Zhu, Abhishek Gupta, Aravind Rajeswaran, Sergey Levine, and Vikash Kumar. Dexterous manipulation with deep reinforcement learning: Efficient, general, and low-cost. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 3651–3657. IEEE, 2019.

[51] Henry Zhu, Justin Yu, Abhishek Gupta, Dhruv Shah, Kristian Hartikainen, Avi Singh, Vikash Kumar, and Sergey Levine. The ingredients of real-world robotic reinforcement learning. *arXiv preprint arXiv:2004.12570*, 2020.

A Appendix

A.1 Proofs

A.1.1 Penalizing Visitation of Irreversible States

We give a simple proof for Theorem 5.1 assuming deterministic dynamics first, and then give a more restricted proof for stochastic dynamics under additional assumptions.

Theorem 5.1. *Let $(s, a) \in \mathcal{S}_{rev}$ denote a state-action leading to a reversible state and let $(s\_, a\_) \notin \mathcal{S}_{rev}$ denote a state-action pair leading to an irreversible state. Then, for all such pairs*

$$Q^\pi(s, a) > Q^\pi(s\_, a\_)$$

for all $\epsilon > 0$ and for all policies π.

Proof. By definition, the reward function is bounded, i.e., $r(s', a') \in [R_{\text{min}}, R_{\text{max}}]$ for any $(s', a') \in \mathcal{S} \times \mathcal{A}$. For any $(s', a') \in \mathcal{S}_{\text{rev}}$, we have the following (using equation 2):

$$\tilde{r}(s', a') = r(s', a') \geq R_{\text{min}} > R_{\text{min}} - \epsilon$$

and for any $(s', a') \notin \mathcal{S}_{\text{rev}}$, we have:

$$\tilde{r}(s', a') = R_{\text{min}} - \epsilon$$

This simplifies to $\tilde{r}(s', a') \geq R_{\text{min}} - \epsilon$ for any $(s', a') \in \mathcal{S} \times \mathcal{A}$. From the definition of Q^π, we have

$$Q^\pi(s', a') = \mathbb{E}\big[\sum_{t=0}^{\infty} \gamma^t \tilde{r}(s_t, a_t) \big| s_0 = s', a_0 = a'\big] \geq \sum_{t=0}^{\infty} \gamma^t (R_{\text{min}} - \epsilon) = \frac{R_{\text{min}} - \epsilon}{1 - \gamma} \tag{5}$$

where the lower bound of $Q^\pi(s', a')$ is achieved if $(s', a') \notin \mathcal{S}_{\text{rev}}$. For $(s, a) \in \mathcal{S}_{\text{rev}}$,

$$\begin{aligned} Q^\pi(s, a) &= \tilde{r}(s, a) + \gamma \mathbb{E}_{a' \sim \pi(\cdot|s')} \left[Q^\pi(s', a')\right] \\ &\geq \tilde{r}(s, a) + \gamma \frac{R_{\text{min}} - \epsilon}{1 - \gamma} \\ &\geq R_{\text{min}} + \gamma \frac{R_{\text{min}} - \epsilon}{1 - \gamma} \qquad \left[\tilde{r}(s, a) = r(s, a) \geq R_{\text{min}}, \text{ since } (s, a) \in \mathcal{S}_{\text{rev}}\right] \\ &= \epsilon + (R_{\text{min}} - \epsilon) + \gamma \frac{R_{\text{min}} - \epsilon}{1 - \gamma} = \epsilon + \frac{R_{\text{min}} - \epsilon}{1 - \gamma} = \epsilon + Q^\pi(s\_, a\_) \end{aligned}$$

where $(s\_, a\_) \notin \mathcal{S}_{\text{rev}}$, implying $Q^\pi(s\_, a\_) = \frac{R_{\text{min}} - \epsilon}{1 - \gamma}$. This concludes the proof as $Q^\pi(s, a) > Q^\pi(s\_, a\_)$ for $\epsilon > 0$. □

To extend the discussion to stochastic dynamics, we redefine the reversibility set as $\mathcal{S}_{\text{rev}} = \{(s, a) \mid \mathbb{P}\left(\mathcal{R}_\rho(s') = 1\right) \geq \eta_1\}$, i.e, state-action pairs leading to a reversible state with at least η_1 probability and let $\mathcal{S}_{\text{irrev}} = \{(s, a) \mid \mathbb{P}\left(\mathcal{R}_\rho(s') = 1\right) \leq \eta_2\}$ denote the set of state-action pairs leading to irreversible states with at most η_2 probability. The goal is to show that actions leading to reversible states with high probability will have higher Q-values than those leading to irreversible states with high probability. The surrogate reward function $\tilde{r}$ is defined as:

$$\tilde{r}(s, a, s') = \begin{cases} r(s, a, s'), & \mathcal{R}_\rho(s') = 1 \\ R_{\text{min}} - \epsilon, & \mathcal{R}_\rho(s') = 0 \end{cases} \tag{6}$$

Theorem A.1. *Let $(s, a) \in \mathcal{S}_{rev}$ denote a state-action leading to a reversible state with at least η_1 probability and let $(s\_, a\_) \in \mathcal{S}_{irrev}$ denote a state-action pair leading to an irreversible state with at least $(1 - \eta_2)$ probability. Assuming $\eta_1 > \eta_2/(1 - \gamma)$,*

$$Q^\pi(s, a) > Q^\pi(s\_, a\_)$$

for all $\epsilon > \frac{\eta_2}{\eta_1 - \gamma\eta_1 - \eta_2} (R_{max} - R_{min})$ and for all policies π.

Proof. For any $(s, a) \in \mathcal{S}_{\text{rev}}$, we have the following (using equation 6):

$$\begin{aligned}
Q^\pi(s,a) &= \mathbb{P}\left(\mathcal{R}_\rho(s') = 1\right)\Big(r(s,a) + \gamma \mathbb{E}_{a' \sim \pi(\cdot|s')}\left[Q^\pi(s',a')\right]\Big) + \mathbb{P}\left(\mathcal{R}_\rho(s') = 0\right)\frac{R_{\text{min}} - \epsilon}{1-\gamma} \\
&\geq \mathbb{P}\left(\mathcal{R}_\rho(s') = 1\right)\Big(R_{\text{min}} + \gamma\frac{R_{\text{min}} - \epsilon}{1-\gamma}\Big) + \mathbb{P}\left(\mathcal{R}_\rho(s') = 0\right)\frac{R_{\text{min}} - \epsilon}{1-\gamma} \\
&\geq \eta_1\Big(R_{\text{min}} + \gamma\frac{R_{\text{min}} - \epsilon}{1-\gamma}\Big) + (1-\eta_1)\frac{R_{\text{min}} - \epsilon}{1-\gamma} \\
&= \eta_1 R_{\text{min}} + (1 + \gamma\eta_1 - \eta_1)\frac{R_{\text{min}} - \epsilon}{1-\gamma} \qquad (7)
\end{aligned}$$

where $\mathbb{P}(\mathcal{R}_\rho(s') = 1) \geq \eta_1$. For any $(s, a) \in \mathcal{S}_{\text{irrev}}$,

$$\begin{aligned}
Q^\pi(s,a) &= \mathbb{P}\left(\mathcal{R}_\rho(s') = 1\right)\Big(r(s,a) + \gamma \mathbb{E}_{a' \sim \pi(\cdot|s')}\left[Q^\pi(s',a')\right]\Big) + \mathbb{P}\left(\mathcal{R}_\rho(s') = 0\right)\frac{R_{\text{min}} - \epsilon}{1-\gamma} \\
&\leq \mathbb{P}\left(\mathcal{R}_\rho(s') = 1\right)\frac{R_{\text{max}}}{1-\gamma} + \mathbb{P}\left(\mathcal{R}_\rho(s') = 0\right)\frac{R_{\text{min}} - \epsilon}{1-\gamma} \\
&\leq \eta_2\frac{R_{\text{max}}}{1-\gamma} + (1-\eta_2)\frac{R_{\text{min}} - \epsilon}{1-\gamma} \qquad (8)
\end{aligned}$$

as $\mathbb{P}(\mathcal{R}_\rho(s') = 1) \leq \eta_2)$ by definition of $\mathcal{S}_{\text{irrev}}$. Under the assumption that $\eta_1 > \eta_2/(1-\gamma)$, whenever

$$\begin{aligned}
\eta_1 R_{\text{min}} + (1 + \gamma\eta_1 - \eta_1)\frac{R_{\text{min}} - \epsilon}{1-\gamma} &> \eta_2\frac{R_{\text{max}}}{1-\gamma} + (1-\eta_2)\frac{R_{\text{min}} - \epsilon}{1-\gamma} \\
\eta_1 R_{\text{min}} - \eta_2\frac{R_{\text{max}}}{1-\gamma} &> (\eta_1 - \gamma\eta_1 - \eta_2)\frac{R_{\text{min}} - \epsilon}{1-\gamma} \\
\eta_1(1-\gamma)R_{\text{min}} - \eta_2 R_{\text{max}} &> (\eta_1 - \gamma\eta_1 - \eta_2)(R_{\text{min}} - \epsilon) \\
\epsilon &> \frac{\eta_2}{(\eta_1 - \gamma\eta_1 - \eta_2)}\left(R_{\text{max}} - R_{\text{min}}\right),
\end{aligned}$$

ensures that $Q^\pi(s, a) > Q^\pi(s\_, a\_)$ for all $(s, a) \in \mathcal{S}_{\text{rev}}$ and for all $(s\_, a\_) \in \mathcal{S}_{\text{irrev}}$, finishing the proof. □

The above proof guarantees that for actions that lead to reversible states with high probability will have higher Q-values than actions leading to irreversible states with high probabilities for *all policies*, even under stochastic dynamics. The restrictive assumption of $\eta_1 > \eta_2/(1-\gamma)$ is required to ensure that the Q-value for worst $(s, a) \in \mathcal{S}_{\text{rev}}$ is better than the best $(s, a) \in \mathcal{S}_{\text{irrev}}$. An alternate analysis can be found in [41, Appendix A.2], where the guarantees are only given for the optimal Q-functions but under less stringent assumptions. Improved guarantees are deferred to future work.

A.1.2 On Empirical Bellman Backup Operator

The empirical Bellman backup operator was introduced in subsection 5.2. In this section, we prove that it is a contraction, and analyze the convergence under empirical Bellman backup. For $\hat{\mathcal{R}}_\rho : \mathcal{S} \mapsto [0, 1]$, the empirical Bellman backup operator can be written as:

$$\hat{\mathcal{B}}^\pi Q(s,a) = \mathbb{E}_{s' \sim \mathcal{P}(\cdot|s,a)}\left[\hat{\mathcal{R}}_\rho(s')\left(r(s,a) + \gamma\mathbb{E}_{a' \sim \pi(\cdot|s')}Q(s',a')\right) + \Big(1 - \hat{\mathcal{R}}_\rho(s')\Big)\frac{R_{\text{min}} - \epsilon}{1-\gamma}\right] \qquad (9)$$

Theorem A.2. *Empirical Bellman backup operator in equation 9 is a Contraction under the L_∞ norm.*

Proof. For all $(s, a) \in \mathcal{S} \times \mathcal{A}$, we have the following:

$$\begin{aligned}
\left|\hat{\mathcal{B}}^{\pi} Q(s,a) - \hat{\mathcal{B}}^{\pi} Q'(s,a)\right| &= \gamma \left|\mathbb{E}_{s' \sim \mathcal{P}(\cdot|s,a)} \left[\hat{\mathcal{R}}_{\rho}(s') \mathbb{E}_{a' \sim \pi(\cdot|s')} \left[Q(s',a') - Q'(s',a')\right]\right]\right| \\
&\leq \gamma \left|\mathbb{E}_{s' \sim \mathcal{P}(\cdot|s,a)} \left[\mathbb{E}_{a' \sim \pi(\cdot|s')} \left[Q(s',a') - Q'(s',a')\right]\right]\right| \\
&\qquad \Big(\text{since } \hat{\mathcal{R}}_{\rho}(s') \in [0,1]\Big) \\
&\leq \gamma \max_{(s'',a'') \in \mathcal{S} \times \mathcal{A}} \left|Q(s'',a'') - Q'(s'',a'')\right| \\
&= \gamma \|Q - Q'\|_{\infty}
\end{aligned}$$

Since this holds for all $(s, a) \in \mathcal{S} \times \mathcal{A}$, this implies:

$$||\mathcal{B}^{\pi} Q - \mathcal{B}^{\pi} Q'||_{\infty} = \max_{(s,a) \in \mathcal{S} \times \mathcal{A}} |\mathcal{B}^{\pi} Q(s,a) - \mathcal{B}^{\pi} Q'(s,a)| \leq \gamma ||Q - Q'||_{\infty}$$

The discount factor $\gamma < 1$, proving our claim. □

For a policy π, let Q^{π} be the true Q-value function computed using conventional Bellman backup $\mathcal{B}^{\pi}$ and $\hat{Q}^{\pi}$ be the Q-values computed using the empirical Bellman backup $\hat{\mathcal{B}}^{\pi}$. Being fixed point of the operators, we have $Q^{\pi} = \mathcal{B}^{\pi} Q^{\pi}$ and $\hat{Q}^{\pi} = \hat{\mathcal{B}}^{\pi} \hat{Q}^{\pi}$. The following theorem relates the two:

Theorem A.3. *Assuming that $\|\mathcal{R}_{\rho} - \hat{\mathcal{R}}_{\rho}\|_{\infty} \leq \delta$, the difference between true Q-values and empirical Q-values for any policy π obeys the following inequality:*

$$\left|Q^{\pi}(s,a) - \hat{Q}^{\pi}(s,a)\right| \leq \frac{\delta\left(R_{max} - R_{min} + \epsilon\right)}{(1-\gamma)^2}$$

Proof. Since Q^{π} is the fixed point of $\mathcal{B}^{\pi}$ and $\hat{Q}^{\pi}$ is the fixed point of $\hat{\mathcal{B}}^{\pi}$, we can write:

$$\begin{aligned}
\left|Q^{\pi}(s,a) - \hat{Q}^{\pi}(s,a)\right| &= \left|\mathcal{B}^{\pi} Q^{\pi} - \hat{\mathcal{B}}^{\pi} \hat{Q}^{\pi}\right| \\
&= \Bigg|\mathbb{E}_{s' \sim \mathcal{P}(\cdot|s,a)} \Bigg[\Big(\mathcal{R}_{\rho}(s') - \hat{\mathcal{R}}_{\rho}(s')\Big)\left(r(s,a) - \frac{R_{\text{min}} - \epsilon}{1-\gamma}\right) \\
&\qquad + \gamma \mathbb{E}_{a' \sim \pi(\cdot|s)} \Big[\mathcal{R}_{\rho}(s') Q^{\pi}(s',a') - \hat{\mathcal{R}}_{\rho}(s') \hat{Q}^{\pi}(s',a')\Big]\Bigg]\Bigg|
\end{aligned} \tag{10}$$

Consider the following identity:

$$\begin{aligned}
\mathcal{R}_{\rho}(s') Q^{\pi}(s',a') - \hat{\mathcal{R}}_{\rho}(s') \hat{Q}^{\pi}(s',a') &= \mathcal{R}_{\rho}(s') Q^{\pi}(s',a') - \mathcal{R}_{\rho}(s') \hat{Q}^{\pi}(s',a') \\
&\qquad + \mathcal{R}_{\rho}(s') \hat{Q}^{\pi}(s',a') - \hat{\mathcal{R}}_{\rho}(s') \hat{Q}^{\pi}(s',a') \\
&= \mathcal{R}_{\rho}(s') \Big(Q^{\pi}(s',a') - \hat{Q}^{\pi}(s',a')\Big) \\
&\qquad + \Big(\mathcal{R}_{\rho}(s') - \hat{\mathcal{R}}_{\rho}(s')\Big) \hat{Q}(s',a')
\end{aligned} \tag{11}$$

Plugging Eq 11 in Eq 10, we get:

$$\begin{aligned}
\left|Q^{\pi}(s,a) - \hat{Q}^{\pi}(s,a)\right| &= \Bigg|\mathbb{E}_{s' \sim \mathcal{P}(\cdot|s,a)} \Bigg[\Big(\mathcal{R}_{\rho}(s') - \hat{\mathcal{R}}_{\rho}(s')\Big)\left(r(s,a) + \gamma \mathbb{E}_{a' \sim \pi(\cdot|s')}[\hat{Q}^{\pi}(s',a')] - \frac{R_{\text{min}} - \epsilon}{1-\gamma}\right) \\
&\qquad + \gamma \mathcal{R}_{\rho}(s') \mathbb{E}_{a' \sim \pi(\cdot|s)} \Big[Q^{\pi}(s',a') - \hat{Q}^{\pi}(s',a')\Big]\Bigg]\Bigg|
\end{aligned}$$

Taking the modulus inside the expectation and using the triangle inequality, we get:

$$\begin{aligned}
\left|Q^{\pi}(s,a) - \hat{Q}^{\pi}(s,a)\right| &\leq \mathbb{E}_{s' \sim \mathcal{P}(\cdot|s,a)} \Bigg[\Big|\mathcal{R}_{\rho}(s') - \hat{\mathcal{R}}_{\rho}(s')\Big| \left|r(s,a) + \gamma \mathbb{E}_{a' \sim \pi(\cdot|s')}[\hat{Q}^{\pi}(s',a')] - \frac{R_{\text{min}} - \epsilon}{1-\gamma}\right| \\
&\qquad + \gamma \left|\mathcal{R}_{\rho}(s')\right| \mathbb{E}_{a' \sim \pi(\cdot|s)} \left|Q^{\pi}(s',a') - \hat{Q}^{\pi}(s',a')\right|\Bigg]
\end{aligned}$$

Using the following inequalities: $r(s,a)+\gamma\mathbb{E}_{a'\sim\pi(\cdot|s')}[\hat{Q}^{\pi}(s',a')] \leq R_{\max}/(1-\gamma)$ as the maximum environment reward is $R_{\max}$, $\left|\hat{\mathcal{R}}_{\rho}(s')\right| \leq 1$ as $\hat{\mathcal{R}}_{\rho} \in [0,1]$ and the assumption $\|\mathcal{R}_{\rho} - \hat{\mathcal{R}}_{\rho}\|_{\infty} \leq \delta$, we can complete the proof:

$$\begin{aligned}\left|Q^{\pi}(s,a) - \hat{Q}^{\pi}(s,a)\right| &\leq \delta\frac{R_{\max}-R_{\min}+\epsilon}{1-\gamma} + \gamma\mathbb{E}_{s'\sim\mathcal{P}(\cdot|s,a),a'\sim\pi(\cdot|s)}\left|Q^{\pi}(s',a') - \hat{Q}^{\pi}(s',a')\right| \\ &\leq \delta\frac{R_{\max}-R_{\min}+\epsilon}{1-\gamma} + \gamma\mathbb{E}_{s'\sim\mathcal{P}(\cdot|s,a),a'\sim\pi(\cdot|s)}\left[\delta\frac{R_{\max}-R_{\min}+\epsilon}{1-\gamma} + \gamma\mathbb{E}\ldots\right] \\ &\leq \delta\frac{R_{\max}-R_{\min}+\epsilon}{(1-\gamma)^2}\end{aligned}$$

□

Theorem A.3 gives us a bound on the difference between the true Q-values and Q-values computed using the reversibility estimator $\hat{\mathcal{R}}_{\rho}$ for any policy π. The bound and the proof also suggest that closer $\hat{\mathcal{R}}_{\rho}(s)$ is to $\mathcal{R}_{\rho}$, closer the estimated Q-values are to true ones.

Similar to the empirical Bellman backup, we can define the empirical Bellman optimality operator:

$$\hat{\mathcal{B}}^{*}Q(s,a) = \mathbb{E}_{s'\sim\mathcal{P}(\cdot|s,a)}\left[\hat{\mathcal{R}}_{\rho}(s')\left(r(s,a)+\gamma\max_{a'}Q(s',a')\right) + \left(1-\hat{\mathcal{R}}_{\rho}(s')\right)\frac{R_{\min}-\epsilon}{1-\gamma}\right] \quad (12)$$

The empirical Bellman optimality operator is also a contraction, following a proof similar as that for empirical Bellman backup. Let $\hat{Q}^{*}$ denote the fixed point of Bellman optimality operator, that is $\hat{B}^{*}\hat{Q}^{*} = \hat{Q}^{*}$, and let $\hat{\pi}^{*}$ denote the greedy policy with respect to $\hat{Q}^{*}$. As a final result,

Theorem A.4. *Let π^{*} denote the optimal policy and Q^{*} denote the corresponding optimal Q-value function. Let $\hat{\pi}^{*}$ denote the optimal policy returned by empirical Bellman optimality operator $\hat{\mathcal{B}}^{*}$. Assuming $\|\mathcal{R}_{\rho} - \hat{\mathcal{R}}_{\rho}\|_{\infty} \leq \delta$,*

$$Q^{\hat{\pi}^{*}}(s,a) \geq Q^{*}(s,a) - \frac{2\delta\left(R_{max}-R_{min}+\epsilon\right)}{(1-\gamma)^2}$$

for all $(s,a) \in \mathcal{S}\times\mathcal{A}$.

Proof. For clarity of notation, we will use $Q(\pi)$ denote $Q^{\pi}(s,a)$ and $\hat{Q}(\pi)$ denote $\hat{Q}^{\pi}(s,a)$ for a policy π. Now,

$$Q^{*}(s,a) - Q^{\hat{\pi}^{*}}(s,a) = \left(Q(\pi^{*}) - \hat{Q}(\pi^{*})\right) + \left(\hat{Q}(\pi^{*}) - \hat{Q}(\hat{\pi}^{*})\right) + \left(\hat{Q}(\hat{\pi}^{*}) - Q(\hat{\pi}^{*})\right)$$

Using the fact $\hat{\pi}^{*}$ is the optimal policy with respect to $\hat{\mathcal{B}}^{*}$, we have $\hat{Q}(\pi^{*}) - \hat{Q}(\hat{\pi}^{*}) \leq 0$. This implies that:

$$Q^{*}(s,a) - Q^{\hat{\pi}^{*}}(s,a) \leq \left(Q(\pi^{*}) - \hat{Q}(\pi^{*})\right) + \left(\hat{Q}(\hat{\pi}^{*}) - Q(\hat{\pi}^{*})\right) \quad (13)$$

Using theorem A.3, we have

$$\begin{aligned}\left(Q(\pi^{*}) - \hat{Q}(\pi^{*})\right) &\leq \frac{\delta\left(R_{\max}-R_{\min}+\epsilon\right)}{(1-\gamma)^2} \\ \left(\hat{Q}(\hat{\pi}^{*}) - Q(\hat{\pi}^{*})\right) &\leq \frac{\delta\left(R_{\max}-R_{\min}+\epsilon\right)}{(1-\gamma)^2}\end{aligned}$$

Plugging these in Eq 13, we get

$$Q^{*}(s,a) - Q^{\hat{\pi}^{*}}(s,a) \leq \frac{2\delta\left(R_{\max}-R_{\min}+\epsilon\right)}{(1-\gamma)^2}$$

Rearranging the above bound gives us the statement in the theorem. □

Theorem A.4 gives us the assurance that as long as the estimator $\hat{\mathcal{R}}_{\rho}$ is close to $\mathcal{R}_{\rho}$, the Q-values of the greedy policy obtained by value iteration using $\hat{\mathcal{B}}^{*}$, i.e. $\hat{\pi}^{*}$ will not be much worse than the optimal Q-values. The above result also suggests choosing a smaller ϵ for a smaller gap in performance.

A.2 Detailed Pseudocode

In Algorithms 3 and 4, we provide pseudo-code for the episodic and non-episodic variants of our method PAINT. We build the non-episodic variant upon the MEDAL algorithm [36], which introduces a backward policy whose objective is to match the state distribution of the forward demonstrations (summarized in Section 4).

Algorithm 3: PAINT (Episodic)

```
optional: forward demonstrations $\mathcal{N}$
initialize: $\pi, Q, \mathcal{D}$; // forward agent parameters
initialize $\hat{\mathcal{R}}_\rho, \mathcal{D}_\rho$; // reversibility classifier and dataset of labels
// add demonstrations to replay buffer
$\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{N}$;
while not done do
    $s \sim \rho_0$ // reset environment
    $\mathcal{D}_{\text{new}} \leftarrow \mathcal{D}_{\text{new}} \cup \{s\}$;
    aborted $\leftarrow$ False;
    for $t = 1, 2, \ldots, H$ do
        if not aborted then
            $a \sim \pi(\cdot \mid s)$;
            update $\pi, Q$; // Eq 9
        else
            $a \sim$ unif$(\mathcal{A})$;
        $s' \sim \mathcal{P}(\cdot \mid s, a), r \leftarrow r(s, a)$;
        $\mathcal{D} \leftarrow \mathcal{D} \cup \{(s, a, s', r)\}$;
        if not aborted and $\hat{R}_\rho(s) \leq 0.5$ then
            aborted $\leftarrow$ True;
        $\mathcal{D}_{\text{new}} \leftarrow \mathcal{D}_{\text{new}} \cup \{s'\}$;
        $s \leftarrow s'$;
    // query reversibility labels for newly collected states via Alg. 1
    label $\mathcal{D}_{\text{new}}$;
    $\mathcal{D}_\rho \leftarrow \mathcal{D}_\rho \cup \mathcal{D}_{\text{new}}$;
    $\mathcal{D}_{\text{new}} \leftarrow \emptyset$;
    // train classifier on all labeled data, new and old
    update $\hat{\mathcal{R}}_\rho$;
```

A.3 Environment Details

In this section, we provide details of each of the four irreversible environments, which are visualized in Fig. 4 and Fig. 5.

Maze. In this 2-D continuous-control environment, the agent is a point mass particle that starts in the top left corner of the maze and must reach the bottom right corner. Throughout the environment, there are trenches (marked in black) that the agent must avoid. Entering them is irreversible: the agent can roam freely within the trench but cannot leave it without an environment reset. The agent is placed back at the top left corner upon a reset, which is provided every 500 time-steps. The agent's state space consists of its xy-position, and two control inputs correspond to the change applied to its xy-position. The reward function is defined as $r_t = \mathbb{1}(\|s_t - g\|_2 < 0.1)$, where g is the goal position. We provide 10 demonstrations of the task to the agent at the beginning of training.

Tabletop Organization. This environment modifies the Tabletop Organization task from Sharma et al. [35]. The agent's objective is to grasp the mug and move it to one of the four specified goal positions. Grasping the mug and dropping it off beyond the red boundary is irreversible, i.e., the agent can no longer re-grasp the mug. Upon a reset, which is provided every 200K time-steps or when requested by the agent, the agent is placed back at the center of the table and the mug is placed on its right, just within the red boundary (see Fig. 4). The agent's state space consists of the its own xy-position, the mug's xy-position, an indicator of whether the mug is grasped, the goal position of the mug, and finally, the goal position of the agent after putting the mug down at its goal. There are three control inputs, which apply changes to the agent's xy-position and toggle between grasping, if

Algorithm 4: PAINT with MEDAL [36] (Non-episodic)

```
input: forward demonstrations $\mathcal{N}_f$;
optional: backward demonstrations $\mathcal{N}_b$;
initialize: $\pi_f, Q^f, \mathcal{D}_f$; // forward agent parameters
initialize: $\pi_b, Q^b, \mathcal{D}_b$; // backward agent parameters
initialize $\hat{\mathcal{R}}_\rho, \mathcal{D}_\rho$; // reversibility classifier and dataset of labels
initialize $C(s)$; // state-space discriminator for backward policy
// add demonstrations to replay buffer
$\mathcal{D}_f \leftarrow \mathcal{D}_f \cup \mathcal{N}_f$;
$\mathcal{D}_b \leftarrow \mathcal{D}_b \cup \mathcal{N}_b$;
$\mathcal{D}_{\text{new}} \leftarrow \emptyset$ ;
while not done do
    $s \sim \rho_0$; // reset environment
    $\mathcal{D}_{\text{new}} \leftarrow \mathcal{D}_{\text{new}} \cup \{s\}$;
    // continue till the reversibility classifier detects an irreversible state
    while $\hat{\mathcal{R}}_\rho(s) > 0.5$ do
        // run forward policy for a fixed number of steps, switch to backward policy
        if forward then
            $a \sim \pi_f(\cdot \mid s)$;
            $s' \sim \mathcal{P}(\cdot \mid s, a), r \leftarrow r(s, a)$;
            $\mathcal{D}_f \leftarrow \mathcal{D}_f \cup \{(s, a, s', r\}$;
            update $\pi_f, Q^f$; // Eq 9
        else
            $a \sim \pi_b(\cdot \mid s)$;
            $s' \sim \mathcal{P}(\cdot \mid s, a), r \leftarrow -\log(1 - C(s'))$;
            $\mathcal{D}_b \leftarrow \mathcal{D}_b \cup \{(s, a, s', r)\}$;
            update $\pi_b, Q^b$; // Eq 9
        // train disriminator every $K$ steps
        if train-discriminator then
            // sample a batch of positives $S_p$ from the forward demos $\mathcal{N}_f$, and a batch of
            negatives $S_n$ from backward replay buffer $\mathcal{D}_b$
            $S_p \sim \mathcal{N}_f, S_n \sim \mathcal{D}_b$;
            update $C$ on $S_p \cup S_n$;
        $\mathcal{D}_{\text{new}} \leftarrow \mathcal{D}_{\text{new}} \cup \{s'\}$;
        $s \leftarrow s'$;
    // optionally explore environment
    for explore steps do
        $a \sim$ unif($\mathcal{A}$), $s' \sim \mathcal{P}(\cdot \mid s, a), r \leftarrow r(s, a)$;
        update $\mathcal{D}_f, \mathcal{D}_b$; // use $C(s)$ for the reward labels in $\mathcal{D}_b$
        $\mathcal{D}_{\text{new}} \leftarrow \mathcal{D}_{\text{new}} \cup \{s'\}$;
        $s \leftarrow s'$;
    // query reversibility labels for newly collected states via Alg 1
    label $\mathcal{D}_{\text{new}}$;
    $\mathcal{D}_\rho \leftarrow \mathcal{D}_\rho \cup \mathcal{D}_{\text{new}}$;
    $\mathcal{D}_{\text{new}} \leftarrow \emptyset$;
    // train classifier on all labeled data, new and old
    update $\hat{\mathcal{R}}_\rho$ on $\mathcal{D}_\rho$;
```

the object is nearby (i.e., within a distance of 0.4), and releasing, if the object is currently grasped. The agent's reward function is $r_t = \mathbb{1}(\|s_t - g\|_2 < 0.1)$, i.e., both the agent's xy-position and the mug's xy-position must be close to their targets. We provide 50 forward demonstrations, 50 backward demonstrations, and 1000 examples of (randomly generated) irreversible states to the agent at the beginning of training.

Peg Insertion. This environment modifies the Peg Insertion task from Sharma et al. [35]. The objective of this task is to grasp and insert the peg into the hole in the box. We modified the table so that the raised edges that stop the peg from rolling off the table are removed and the table is significantly narrower. Hence, the peg may fall off the table, which cannot be reversed by the robot. Instead, when the environment is reset, which automatically occurs every 100K time-steps or when requested by the agent, the peg is placed back at one of 15 possible initial positions on the table. The agent's state space consists of the robot's xyz-position, the distance between the robot's gripper fingers, and the object's xyz-position. The agent's action consists of 3D end-effector control and normalized gripper torque. Let s_t^{peg} represent the state of the peg and g^{peg} be its goal state, then the reward function is $r_t = \mathbb{1}(\|s_t^{\text{peg}} - g^{\text{peg}}\|_2 < 0.05)$. We provide the agent with 12 forward demonstrations and 12 backward demonstrations.

Half-Cheetah. We design this environment based on the version from Brockman et al. [9]. In particular, the agent must run at one of six target velocities $\{3, 4, 5, 6, 7, 8\}$, specified to the agent. Every 500 time-steps, the target velocity switches to a different value selected at random. The agent's actions, which correspond to torque control of the cheetah's six joints, are scaled up by a factor of 5. When the agent is flipped onto its back (i.e., its normalized orientation is greater than $2\pi/3$), we label these states irreversible. When the agent is reset, which only occurs when requested, the agent is placed upright again at angle of 0. The agent's observation space consists of the velocity of the agent's center of mass, angular velocity of each of its six joints, and the target velocity. Let v be the velocity of the agent, then the reward function, which is normalized to be between 0 and 1, is $r_t = 0.95 * (8 - v)/8 + 0.05 * (6 - \|a_t\|_2^2)/6$. There are no demonstrations for this task.

A.4 Implementation Details

Below, we provide implementation details of our algorithm PAINT and the baselines. Every algorithm, including ours, has the following components.

Forward policy network. The agent's forward policy is represented by an MLP with 2 fully-connected layers of size 256 in all experimental domains, trained with the Soft Actor-Critic (SAC) [18] algorithm.

Forward critic network. The agent's forward critic is represented by an MLP with 2 fully-connected layers of size 256 in all experimental domains, trained with the Soft Actor-Critic (SAC) [18] algorithm.

Balanced batches. In the Tabletop Organization and Peg Insertion tasks, the agent's forward policy and critic networks are trained with batches that consist of demonstration tuples and of tuples sampled from the agent's online replay buffer. We control the ratio p of demonstration tuples to online tuples with a linearly decaying schedule,

$$p_t = \begin{cases} \frac{(p_T - p_0)}{T} t + p_0 & t < T \\ p_T & t \geq T. \end{cases}$$

In the Tabletop Organization and Peg Insertion tasks, $p_0 = 0.5$, $p_T = 0.1$, and $T = 500$K. We do not train with balanced batches in the Maze task, and simply populate the online replay buffer with the demonstrations at the beginning of training.

Episodic RL (Soft Actor-Critic) [18]. In addition to the policy and critic networks trained with balanced batches, the episodic RL comparison requests for resets every H' time-steps in the Tabletop Organization ($H' = 2000$), Peg Insertion ($H' = 1000$), and Half-Cheetah ($H' = 2000$) tasks.

Safe Model-Based Policy Optimization (SMBPO) [41]. This comparison trains the forward critic with the modified Bellman update $\mathcal{B}^\pi Q(s, a)$ defined in Eqn. 4, where $\epsilon = 0$ in the Maze and Half-Cheetah tasks and $\epsilon = -0.1$ in the Tabletop Organization and Peg Insertion tasks.

Safety Q-functions for RL (SQRL) [37]. This comparison trains an additional safety critic Q_{safe}^π, which estimates the future probability of entering an irreversible state, and the policy is updated with the following augmented objective

$$J_{\text{safe}}^\pi(\nu) = J^\pi + \mathbb{E}_{(s \sim \mathcal{D}), a \sim \pi(\cdot|s)} \left[\nu \left(\epsilon_{\text{safe}}, Q_{\text{safe}}^\pi(s, a)\right)\right],$$

where ν is the Lagrange multiplier for the safety constraint and is updated via dual gradient descent. We only evaluate this comparison in the Maze task, where $\epsilon_{\text{safe}} = 10$.

A.4.1 Forward-Backward Algorithms

In the forward-backward setups, we train a backward policy and critic in addition to their forward counterparts. The details of the backward components that are shared across all methods are described below.

Backward policy network. The agent's backward policy is represented by an MLP with 2 fully-connected layers of size 256 in all experimental domains, trained with the SAC algorithm.

Backward critic network. The agent's backward critic is represented by an MLP with 2 fully-connected layers of size 256 in all experimental domains, trained with the SAC algorithm.

Backward balanced batches. The agent's backward policy and critic networks are trained with batches that consist of demonstration tuples and of tuples sampled from the agent's online replay buffer. We control the ratio p of demonstration tuples to online tuples with a linearly decaying schedule,

$$p_t = \begin{cases} \frac{(p_T - p_0)}{T} t + p_0 & t < T \\ p_T & t \geq T. \end{cases}$$

In the Tabletop Organization and Peg Insertion tasks, $p_0 = 0.5$, $p_T = 0.1$, and $T = 500$K.

Leave No Trace (LNT) [11]. This comparison additionally trains backward policy and critic networks, whose reward function is the sparse indicator of whether the current state is within some threshold of the initial state. The thresholds are the same as those used for the forward reward functions defined in A.3.

Policy switching. Leave No Trace switches from the forward to backward policy if the backward critic's Q-value is lower than ϵ_{LNT} or after 300 time-steps, and switches from the backward to forward policy after 300 time-steps. In Tabletop, $\epsilon_{\text{LNT}} = 0.1$, and in Peg Insertion, $\epsilon_{\text{LNT}} = 0.005$.

Termination condition. Leave No Trace additionally requests a reset if, after 300 time-steps, the backward policy fails to bring the environment within a distance of 0.1 of the initial state.

Matching Expert Distributions for Autonomous Learning (MEDAL) [36]. Like Leave No Trace, MEDAL trains a backward policy and critic. However, instead of returning to the initial state, the backward reward function is whether the current state matches the distribution of demonstration states, formally defined in Eqn. 1.

MEDAL classifier. The classifier C in Eqn. 1 is represented by an MLP with 1 FC layer of size 128.

Policy switching. The algorithm switches policies (i.e., from forward to backward and from backward to forward) after every 300 time-steps.

Reversibility-Aware Exploration (RAE) [15]. RAE trains a self-supervised reversibility estimator, specifically to predict whether a state transition $(s, \tilde{s})$ is more likely than the reverse transition $(\tilde{s}, s)$. RAE generates data for the binary classifier with a windowed approach. For every state trajectory $(s_{t:t+w})$ of length w collected by the agent, all state pairs (s_i, s_j), where $i < j$, are labeled *positive*, and all pairs (s_j, s_i), where $i < j$, are labeled *negative*. For all experimental tasks, we use a window size of $w = 10$ time-steps. With this estimator, the forward critic is trained with the modified Bellman update $\hat{\mathcal{B}}^\pi Q(s, a)$, where $\epsilon = 0$ in the Maze and Half-Cheetah tasks and $\epsilon = -0.1$ in the Tabletop Organization and Peg Insertion tasks.

Reversibility classifier. The classifier $\hat{\mathcal{R}}_\rho$ is represented by an MLP with 1 FC layer of size 128.

Termination condition. In Maze and Half-Cheetah, the termination condition is $\hat{\mathcal{R}}_\rho > 0.5$. In Tabletop Organization and Peg Insertion, the condition is $\hat{\mathcal{R}}_\rho > 0.8$.

Exploration. We augment RAE with uniform-random exploration after the termination condition is met as proposed in our method. In the Maze environment, the agent takes uniform-random actions for the rest of the episode (of length 500). In Tabletop Organization and Peg Insertion, $N_{\text{explore}} = 300$ time-steps. In Half-Cheetah, $N_{\text{explore}} = 500$ time-steps.

In the Tabletop Organization and Peg Insertion tasks, we train an additional backward policy and critic, whose reward functions are defined in terms of the MEDAL classifier. The backward critic is also trained with the modified $\hat{\mathcal{B}}^{\pi} Q(s, a)$, with the same hyperparameters as the forward critic.

MEDAL classifier. The classifier C in Eqn. 1 is represented by an MLP with 1 FC layer of size 128.

Policy switching. The algorithm switches policies (i.e., from forward to backward and from backward to forward) after every 300 time-steps.

PAINT (Ours). PAINT trains a reversibility classifier $\hat{\mathcal{R}}_{\rho}$ and checks whether the current state is estimated to be irreversible. If it is estimated to be irreversible, the agent takes uniform-random actions for H_{explore} time-steps and requests a reset afterward. The forward critic is also trained with the modified Bellman update $\hat{\mathcal{B}}^{\pi} Q(s, a)$, where $\epsilon = 0$ in the Maze and Half-Cheetah tasks and $\epsilon = -0.1$ in the Tabletop Organization and Peg Insertion tasks.

Reversibility classifier. The classifier $\hat{\mathcal{R}}_{\rho}$ is represented by an MLP with 1 FC layer of size 128.

Termination condition. In all tasks, the termination condition is $\hat{\mathcal{R}}_{\rho} > 0.5$.

Exploration. In the Maze environment, the agent takes uniform-random actions for the rest of the episode (of length 500). In Tabletop Organization and Peg Insertion, $N_{\text{explore}} = 300$ time-steps. In Half-Cheetah, $N_{\text{explore}} = 500$ time-steps.

In the Tabletop Organization and Peg Insertion tasks, we train an additional backward policy and critic. The details for the backward policy and critic are the same as in RAE.

A.5 Additional Experimental Results

In this section, we present additional plots to accompany Section 6.2 and additional ablations to accompany Section 6.3.

A.5.1 Additional Plots

In Fig. 9, we plot the task success, number of reset interventions, and number of reversibility labels versus the number of time-steps in each of the four tasks.

A.5.2 Ablations and Sensitivity Analysis

Termination conditions. To evaluate the importance of the reversibility classifier, we study an alternative choice for the termination condition, one based on the Q-value function. Intuitively, the values for irreversible states will be low, as there is no path that leads to the goal from these states. We define the value-based termination condition as $V^{\pi}(s) < \epsilon$ where ϵ is the threshold. We approximate the value $V^{\pi}(s) = \mathbb{E}_{\pi}[Q^{\pi}(s, a)]$ with the trained Q-value function evaluated at $N = 10$ policy actions. We plot the task success in the Tabletop Manipulation task in Fig. 10 (left). After 1M time-steps, the Q-value-based termination condition requires fewer interventions, approximately half of the interventions needed by PAINT, but does not converge to the same final performance as PAINT with the reversibility classifier. Critically, PAINT with Q-value termination also trains a reversibility classifier to generate pseudo-labels for unlabeled states.

Pseudo-labels for unlabeled states. Currently, when updating the agent with Eqn. 9, our method uses the predictions from the stuck classifier as pseudo-labels for the unlabeled states collected during a trial. An alternative choice is labeling them one, i.e., treating them as reversible. We evaluate this choice in the Tabletop Manipulation task, and plot the task success in Fig. 10 (right). After 1M time-steps, the number of reset interventions requested by both methods are similar. However, our method succeeds at the task almost 100% of the time, while the agent trained with pseudo-labels of one only achieves success of around 60%.

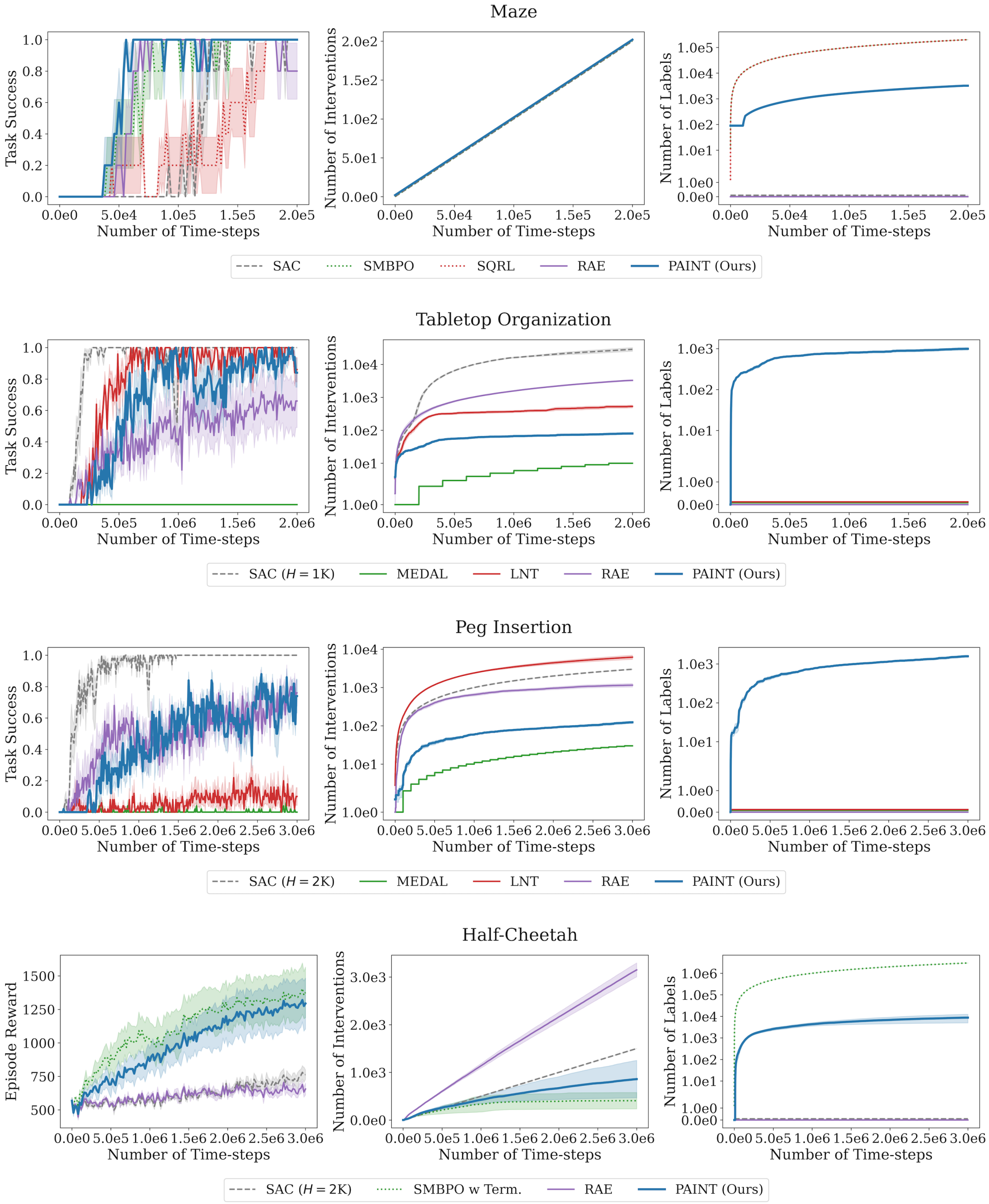

Figure 9: *(left)* Task success versus time. *(middle)* Number of interventions versus time. *(right)* Number of queried labels versus time.

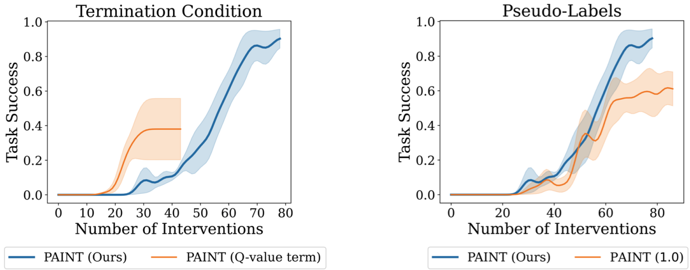

Figure 10: (*left*) We define a termination condition based on the Q-values, which learns with fewer resets but also achieves significantly lower final performance. (*right*) For states collected during a trial, their reversibility labels are still unknown. When training the agent with Eqn. 9, we use predictions from the reversibility classifier as pseudo-labels. We compare to labeling the unlabeled states one, i.e., treating them as reversible.