Learning from Demonstrations via Capability-Aware Goal Sampling

Yuanlin Duan
Rutgers University
yuanlin.duan@rutgers.edu

Yuning Wang
Rutgers University
yw895@cs.rutgers.edu

Wenjie Qiu
Rutgers University
wq37@cs.rutgers.edu

He Zhu
Rutgers University
hz375@cs.rutgers.edu

Abstract

Despite its promise, imitation learning often fails in long-horizon environments where perfect replication of demonstrations is unrealistic and small errors can accumulate catastrophically. We introduce Cago (Capability-Aware Goal Sampling), a novel learning-from-demonstrations method that mitigates the brittle dependence on expert trajectories for direct imitation. Unlike prior methods that rely on demonstrations only for policy initialization or reward shaping, Cago dynamically tracks the agent's competence along expert trajectories and uses this signal to select intermediate steps—goals that are just beyond the agent's current reach—to guide learning. This results in an adaptive curriculum that enables steady progress toward solving the full task. Empirical results demonstrate that Cago significantly improves sample efficiency and final performance across a range of sparse-reward, goal-conditioned tasks, consistently outperforming existing learning from-demonstrations baselines.

1 Introduction

Imitation Learning (IL) offers a compelling paradigm for training agents by leveraging expert demonstrations to address the exploration challenges in Deep Reinforcement Learning (DRL) applications (Arulkumaran and Lillrank, 2024). The simplest IL algorithm is Behavior Cloning (BC), which directly uses demonstrations to supervise policy actions conditioned on the states visited by the expert (Bain and Sammut, 1995; Torabi et al., 2018). However, its applicability remains limited, as even minor deviations from expert behavior can compound over time. Inverse Reinforcement Learning (IRL) has made significant progress by inferring reward functions from demonstrations. Notable approaches such as GAIL (Ho and Ermon, 2016), PWIL (Dadashi et al., 2020), and AdRIL (Eysenbach et al., 2021) aim to match expert and agent state-action distributions directly. More recent advances in offline or offline-to-online reinforcement learning, such as CQL (Kumar et al., 2020) and Cal-QL (Nakamoto et al., 2023), incorporate demonstrations as anchors to regularize learning. These methods penalize value estimates that deviate significantly from demonstrated behavior, helping to mitigate overestimation and instability caused by out-of-distribution actions.

However, existing IL methods often struggle with complex, long-horizon tasks because they fail to reason about which parts of the task the agent has already mastered and which remain challenging. In particular, distribution-matching approaches perform flat matching—attempting to align occupancy measures over the entire trajectory distribution without considering the agent's evolving capabilities. This leads to poor exploration guidance, especially in the early stages of training when the agent seldom reaches meaningful parts of the state space. As a result, the learned reward function tends

Preprint. Under review.

to assign uniformly low rewards, yielding uninformative gradients and hindering effective policy improvement. Some prior work proposes demonstration-guided curriculum learning that trains agents to solve tasks by starting near the goal or high-reward states and gradually expanding to earlier parts of the trajectories (Resnick et al., 2018; Salimans and Chen, 2018; Tao et al., 2024). However, these approaches rely on the ability to reset the agent to arbitrary demonstration states—an assumption impractical in real-world settings due to challenges in replicating physical conditions like joint velocities and angular momentum.

We propose Cago (Capability-Aware Goal Sampling), a new learning-from-demonstrations framework that explicitly aligns the agent's learning process with its evolving capabilities. Unlike prior methods that use demonstrations for direct imitation, reward shaping, or offline pretraining, Cago treats demonstrations as structured roadmaps. It continuously monitors which parts of a demonstration the agent can already reach and leverages this signal to sample intermediate goal states in the demonstration, those at the boudary of the agent's current goal-reaching capabilities. At each episode, a goal-conditioned agent (Liu et al., 2022; Plappert et al., 2018) first attempts to reach the sampled goal and then explores forward from it, generating informative, task-relevant data for policy optimization. This iterative process of capability-aware goal selection and curriculum-aligned exploration enables steadily progress toward solving the full task.

We evaluate Cago across several sparse-reward environments and demonstrate substantial improvements in both sample efficiency and final task performance over existing imitation-based baselines. Our experiments highlight that capability-aware goal sampling provides a powerful signal for structuring learning, particularly in long-horizon tasks.

2 Problem Setup and Background

Reinforcement learning (RL) aims to enable agents to learn optimal behaviors through trial-and-error interactions with an environment. An RL problem is formulated as a Markov Decision Process (MDP), represented as a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{G}, \eta, R, \rho_0)$. The agent operates within a state space $\mathcal{S}$ and takes actions from an action space $\mathcal{A}$, transitioning between states according to the dynamics $\mathcal{T}(s'|s,a)$. $R(s,a) \in \mathbb{R}$ is the reward function and ρ_0 is the initial state distribution. Given a policy π, consider the trajectory $\tau = \{s_0, a_0, s_1, a_1, \ldots\}$ sampled by π, i.e., $s_0 \sim \rho_0$, $a_t \sim \pi(\cdot|s_t)$, and $s_{t+1} \sim \mathcal{T}(\cdot|s_t, a_t)$. The goal of RL is to learn a return-maximizing policy $\pi^* = \arg\max_\pi \mathbb{E}_{\tau \sim \pi(a_t|s_t)} \left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)\right]$ where $\gamma \in (0, 1]$ is the discount factor.

Learning from Demonstrations. In imitation learning, the agent is provided a static dataset of demonstrations $\mathcal{D}_{\text{demo}}$ collected from some expert policy π_{expert}. The goal is to learn a policy π that mimics the expert's behavior by generalizing from these demonstrations. A common approach is behavioral cloning, which casts the problem as supervised learning, training the agent to minimize the discrepancy between its predicted actions and the expert's. An alternative approach is to infer a reward function from expert demonstrations and then optimize a policy through RL on this learned reward. This class of methods, known as inverse RL (IRL) (Abbeel and Ng, 2004), decouples the tasks of reward inference and policy learning. A notable example is Generative Adversarial Imitation Learning (GAIL) (Ho and Ermon, 2016), which frames imitation as a minimax game where a discriminator distinguishes expert and agent behaviors, effectively serving as a learned reward signal for policy optimization. Many recent work in imitation learning can be interpreted through the lens of minimizing a divergence between the expert and agent occupancy measures.

State Reset. Several methods attempt to mitigate the exploration challenge in sparse-reward RL environments by resetting the agent to states from expert demonstrations, thereby bypassing the need to discover those states through the agent's own exploration. These strategies include initializing the agent to states sampled uniformly from demonstration trajectories (Nair et al., 2018; Peng et al., 2018; Hosu and Rebedea, 2016), employing a hand-crafted curriculum (Zhu et al., 2018), or using a reverse curriculum that progressively trains the agent from goal or high-reward states backward (Resnick et al., 2018; Salimans and Chen, 2018; Tao et al., 2024). These approaches assume the ability to reset the agent to arbitrary demonstration states—an assumption that is unrealistic in real-world settings.

Goal-Conditioned RL (GCRL) extends the standard RL framework by conditioning policies on specific target goals, guiding agents toward desired goals. The MDPs are augmented with a goal space $\mathcal{G}$ and are associated with states via a mapping $\eta : \mathcal{S} \rightarrow \mathcal{G}$, ensuring that each state corresponds to an achieved goal. In GCRL, the reward signal from the environment is typically sparse and is defined as: $R(s, a, s', g) = 1\{\eta(s') = g\}$. We assume that each episode has a fixed horizon T and $\mathcal{S} = \mathcal{G}$. The agent's objective is to train a goal-conditioned policy $\pi^G(\cdot|s_t, g)$ to achieve a given goal $g \in \mathcal{G}$ through maximizing the expected cumulative reward $J(\pi) = \mathbb{E}_{g\sim p_g, \tau\sim\pi^G(a_t|s_t,g)}\left[\sum_{t=0}^{T-1}\gamma^t \cdot R(s_t, a_t, s_{t+1}, g)\right]$ where p_g is the goal distribution.

This Paper. We introduce Cago, a novel approach that leverages demonstrations as a scaffold for goal-directed reinforcement learning. Rather than direct imitation, Cago uses demonstrations to guide exploration by training a goal-conditioned policy $\pi^G(a \mid s, g)$ that learns to progressively reach intermediate states g along demonstration trajectories, effectively inducing a curriculum that facilitates steady progress toward solving the full task. In addition, Cago learns a goal predictor $\mathcal{P}(s)$ that infers the final goal state g_T from the current state s. The resulting task policy is defined as $\pi(s) = \pi^G(s, \mathcal{P}(s))$ that enables automatic inference of goal conditions at test time for previously unseen situations.

3 Method

The main idea of Cago is to continuously monitor the agent's evolving capabilities to reach various stages of demonstration trajectories during training. It dynamically selects the most appropriate goal from the demonstrations, conditioned on the agent's current performance ceiling. The selected goal guides online exploration, with the agent first attempting to reach it using its current policy. From there, it continues to explore, collecting task-relevant trajectories in an Go-Explore style (Ecoffet et al., 2019). By anchoring exploration in achievable yet progressively harder goals, this process effectively constructs an implicit curriculum, where the agent is gradually exposed to more challenging states aligned with its growing competence.

3.1 Observation Visit Tracking with Demonstration Alignment

Cago assumes the existence of a limited amount of expert demonstrations $\mathcal{D}_{\text{demo}} : \{\tau^{(i)} = \{(s_0, a_0)^{(i)}, \ldots, (s_{L_i}, a_{L_i})^{(i)}\}\}_{i=1}^{M}$ where M is the number of the demonstrations and L_i is the length of i-th demonstration $\tau^{(i)}$. To select goals at the boundary of the agent's current reaching capabilities, It is crucial to determine the stage at which the agent can accomplish its task completion. Central to our method is maintaining a dictionary $\text{Dict}_{\text{visit}}$ that tracks the visitation frequencies of observations s_i across demonstrations. For each demonstration $\tau^{(i)}$, we initialize an all-zero list of the same length as its steps. Each element in this list records the visitation count of the corresponding observation in the demonstration. At each environment step, the record list is updated to reflect whether the agent has visited observations from the demonstration, based on similarity metrics $\text{sim}(\cdot, \cdot)$ such as L2 distances for state-based environments or mean squared errors (MSE) between images in visual environments. Formally, we define the visitation record dictionary as:

$$\text{Dict}_{\text{visit}} = \tau^{(i)} : [0, 0, \ldots, 0] \in \mathbb{N}^{L_i} \mid i = 1, 2, \ldots, M \tag{1}$$

During online exploration, for each new episode, we first sample a demonstration $\tau^{(i)}$ from $\mathcal{D}_{\text{demo}}$ and reset the environment to the initial state of $\tau^{(i)}$. **Our strategy is more practical in the real-world setting** than related methods (e.g. (Tao et al., 2024; Nair et al., 2018)) that reset the environment to intermediate demonstration states, which are often infeasible to reproduce due to unobservable or difficult-to-control physical factors such as velocity and angular momentum. Given a rollout $\tau = (s_0, s_1, \ldots)$ from the environment, we update $\text{Dict}_{\text{visit}}[\tau^{(i)}]$ as follows:

$$\text{Dict}_{\text{visit}}[\tau^{(i)}][j] \mathrel{+}= 1 \quad \text{if } \text{sim}(s_t, s_j^{(i)}) \leq \epsilon, \quad \forall t \in 1, \ldots, L_\tau, \forall j \in 1, \ldots, L_i \tag{2}$$

where s_t is the agent's observation state at timestep t, L_τ is the total length of the rollout τ, $s_j^{(i)}$ is the j-th observation state in the i-th demonstration $\tau^{(i)}$, $\text{sim}(\cdot, \cdot)$ is the similarity metric (e.g., L2 distance for state-based environments or MSE for image-based environments), and ϵ is a matching threshold. This simple record dictionary effectively tracks the agent's progress and helps identify its goal-reaching capability limits along task demonstrations.

3.2 Capability-Aware Goal Sampling

Cago leverages demonstration-based visitation counts to guide goal selection and trajectory collection in a *capability-aware* manner. Figure 1 illustrates our method. After resetting the environment to the

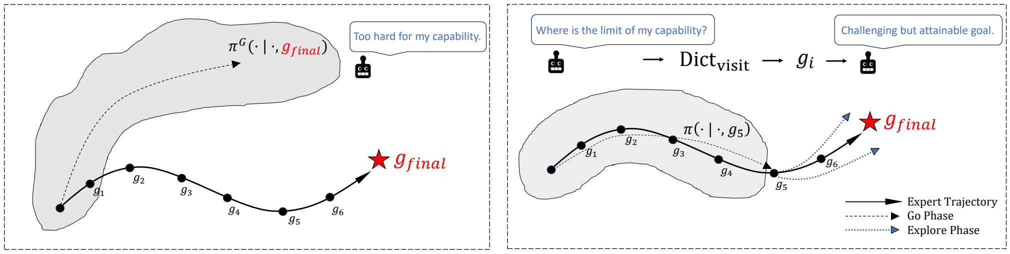


Figure 1: Illustration of the Cago. **Left:** Directly setting the final goal as the agent's target often leads to failure, as the current policy π^G may not yet be capable of reaching it. The shaded region illustrates the set of states currently reachable under π^G. Attempting to reach g_{final} (i.e., executing $\pi^G(\cdot|\cdot, g_{\text{final}})$) causes the agent to diverge from the demonstration trajectory. **Right:** Cago improves learning by leveraging a visitation frequency dictionary $\text{Dict}_{\text{visit}}$ built from demonstrations. Given a demonstration trajectory with subgoals $g_1, g_2, \ldots, g_n$, the agent selects the furthest subgoal g_i that remains within its current capabilities for Go-Explore sampling, enabling a curriculum of progressively more challenging goals aligned with the demonstration.

initial state of a randomly sampled demonstration $\tau^{(i)}$, we select a goal g for the agent to explore:

$$g \sim \mathcal{G}_{\text{cap}}(\pi^G, \tau^{(i)}), \tag{3}$$

where $\mathcal{G}_{cap}(\pi^G, \tau^{(i)})$ denotes a capability-aware goal sampling distribution over subgoals whose reachability is aligned with the current goal-reaching capability of the policy π^G. Cago examines the visitation frequency list to identify the last index where the frequency exceeds a predefined threshold:

$$j^* = \max\left\{ j \mid \text{Dict}_{\text{visit}}[\tau^{(i)}][j] \geq \lambda_{\text{visit}} \right\}, \tag{4}$$

where $\text{Dict}_{\text{visit}}[\tau^{(i)}][j]$ denotes the visitation frequency of j-th observation s_j of $\tau^{(i)}$ under policy π^G and λ_{visit} is a frequency threshold (e.g. 100). This index indicates the latest point in the demonstration that the agent is sufficiently competent at reaching—effectively serving as a proxy for the limit of the agent's current goal-reaching capability. Cago constructs a goal sampling range centered around j^*. The sampled goal is drawn from this range, which allows the agent to either revisit familiar goals or attempt slightly more challenging ones that are just beyond its current capability. This capability-aware goal sampling strategy introduces controlled diversity into the training process and encourages progressive learning, while also avoiding excessively difficult goals that could derail training. The corresponding goal sampling region is defined as:

Algorithm 1 Capability-Aware Goal Sampling (Cago)

1: **Input:** Demonstration $\tau^{(i)}$, Visitation record $\text{Dict}_{\text{visit}}$
2: **Output:** capability-aware goal g
3: Identify capability-aware upper limit point g_i using visitation threshold λ_{visit} (Eq. (4))
4: Define sampling region $\mathcal{G}_{cap}(\pi^G, \tau^{(i)})$ centered around g_i (Eq. (5))
5: Sample subgoal $g \sim \mathcal{G}_{cap}(\pi^G, \tau^{(i)})$
6: **return** g

$$\mathcal{G}_{cap}(\pi^G, \tau^{(i)}) = \left\{ s_k \in \tau^{(i)} \mid |k - j^*| \leq \delta \cdot L_i \right\}, \tag{5}$$

where L_i is the length of $\tau^{(i)}$ and $\delta \in (0, 1]$ controls the window size for goal sampling (e.g., 10% of the demonstration length). Goals are then sampled at random from this set. Our capability-aware goal sampling scheme introduces a curriculum-aligned learning signal that progressively guides the agent with steady improvement toward successful task completion. The overall goal sampling process in Cago is described in Algorithm 1.

3.3 Learning Framework

Go-Explore. Cago trains a goal-conditioned agent following the Go-Explore paradigm (Ecoffet et al., 2019), which divides each episode into two sequential phases: the *Go-phase* and the *Explore-phase*. In the Go phase, the agent is guided towards a sampled goal state g using the goal-conditioned policy

$\pi^G(\cdot|\cdot, g)$, reaching an intermediate state s_E. To improve environment exploration beyond the agent's current capabilities, the Explore-phase takes over from s_E, where an exploration policy π^E is used for the remaining time steps. Since we have access to a limited set of task demonstrations $\mathcal{D}_{\text{demo}}$, we implement π^E as a Behavior Clone (BC) Explorer trained on $\mathcal{D}_{\text{demo}}$. The BC Explorer outputs a stochastic action distribution that enables the agent to balance between exploration and imitation. This two-phase strategy ensures that collected trajectories stay anchored near the demonstration distribution in $\mathcal{D}_{\text{demo}}$, while encouraging exploration. We further analyze the impact of the BC Explorer through ablation studies, detailed in Section 4.

As Cago actively resets environments to initial states drawn from the demonstration set $\mathcal{D}_{\text{demo}}$, a key question is how the agent generalizes beyond $\mathcal{D}_{\text{demo}}$. Our solution is to train the goal-conditioned agent π^G using a richer set of imagined rollouts generated by a world model via model-based RL.

Algorithm 2 The main training framework of Cago

Input: GC Policy π^G, World Model $\widehat{\mathcal{M}}$, Demonstrations $\mathcal{D}_{\text{demo}}$
Initialize replay buffer $\mathcal{D}_{\text{cap}}$ and $\text{Dict}_{\text{visit}}$ (Eq. (1))
Train explorer policy π^E using Behavior Cloning on $\mathcal{D}_{\text{demo}}$
Train goal predictor $\mathcal{P}_\phi$ on $\mathcal{D}_{\text{demo}}$ (Eq. (7))
for $n = 1$ to N_{train} **do**
    Initialize empty trajectory τ
    Randomly sample a demonstration $\tau^{(i)} \in \mathcal{D}_{\text{demo}}$
    Initialize the environment to the initial state of $\tau^{(i)}$
    Sample a capability-aware goal g by Algorithm 1
    for $t = 0$ to L_τ **do**
        if agent has not reached g and $t < T_{go}$ **then**
            $\pi = \pi^G(s, g)$
        else
            $\pi = \pi^E(s)$
        Step in the real environment using π and add this step to τ
        Update $\text{Dict}_{\text{visit}}[\tau^{(i)}]$ (Eq. (2))
    $\mathcal{D}_{\text{cap}} \leftarrow \mathcal{D}_{\text{cap}} \cup \{\tau\}$
    Update $\widehat{\mathcal{M}}$ with $\mathcal{D}_{\text{cap}}$
    Update π^G using imagined rollouts with $\widehat{\mathcal{M}}$

World Model and Policy Training. Cago stores the trajectories generated under the Go-Explore paradigm with capability-aware goal sampling in a dataset $\mathcal{D}_{\text{cap}} = \{(s_t, a_t, s_{t+1})_{t=1}^T\}$ for world model and policy training. A predictive world model $\widehat{\mathcal{M}}$ approximates the transition dynamics $\mathcal{T}(s'|s, a)$ in the real world $\mathcal{M}$ as $\widehat{\mathcal{T}}(s'|s, a)$. Our model learning algorithm is based on the Dreamer backbone (Hafner et al., 2019, 2020, 2023), which updates the world model $\widehat{\mathcal{M}}$ via supervised learning using $\mathcal{D}_{\text{cap}}$. Once the world model is updated, we train the goal-conditioned policy π^G using imagined trajectories generated by the world model $\widehat{\mathcal{M}}$. Intuitively, since $\mathcal{D}_{\text{cap}}$ is collected by exploring around demonstration states in $\mathcal{D}_{\text{demo}}$, the learned model enables the agent to generate imagined trajectories that remain grounded in task-relevant regions of the state space. Each imagined trajectory from the learned world model $\widehat{\mathcal{M}}$ begins at s_0, a state randomly sampled from a trajectory τ in $\mathcal{D}_{\text{demo}} \cup \mathcal{D}_{\text{cap}}$, and is rolled out for H steps using the goal-conditioned policy $\pi^G(a_t|s_t, g)$. The goal state g is selected as a future state s_H from the same trajectory τ. The objective is to train π^G to reinforce trajectories that efficiently reach g in the imagined rollouts from s_0 under the learned dynamics $\widehat{\mathcal{M}}$. To achieve this, we adopt an actor-critic algorithm that leverages a self-supervised temporal distance function $D_t(s, g)$ (Mendonca et al., 2021), which estimates the number of steps required to transition from state s to goal g. The reward function is defined as: $r^G(s, g) = -D_t(s, g)$. This formulation encourages the policy to generate actions that minimize the estimated temporal distance to the goal. The temporal distance estimator D_t is trained by extracting state pairs (s_t, s_{t+k}) from simulated trajectories generated by the world model. The function learns to predict the normalized temporal difference between two states: $D_t\big(\Psi(s_t), \Psi(s_{t+k})\big) \approx \frac{k}{H}$, where Ψ denotes a transformation applied to states (e.g., embedding them into the world model's latent space), and H is the length of the generated rollout. More details on the model-based learning algorithm and the full training procedure for D_t can be found in Appendix B.1 and Appendix B.2.

Goal Predictor. At training time, the goal-conditioned policy $\pi^G(\cdot \mid \cdot, g)$ is trained using intermediate states from demonstration trajectories as goal conditions (recall $\mathcal{S} = \mathcal{G}$). This assumes access to demonstrations, with the final states used as the target goal condition. However, at test time, this assumption no longer holds: for unseen scenarios, the true final goal state is not available. This raises the challenge of how to specify an appropriate goal condition based solely on the agent's current observation. We introduce a goal predictor $\mathcal{P}_\phi$, a learned model that infers a goal state $\hat{g}$ given the current observation s, as

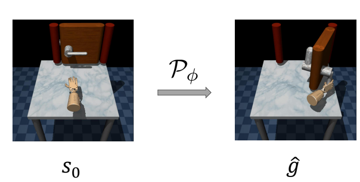


Figure 2: The workflow of the goal predictor $\mathcal{P}_\phi$.

illustrated in Figure 2. The model learns the mapping:

$$\mathcal{P}_\phi : s \mapsto \hat{g}, \quad \text{where } \hat{g} = \mathcal{P}_\phi(s) \tag{6}$$

It is trained using demonstration trajectories $\mathcal{D}_{\text{demo}}$, by minimizing the mean squared error between the predicted goal and the true final observation:

$$\min_{\phi} \mathbb{E}_{(\tau^{(i)}=s_0^{(i)},\ldots,s_L^{(i)})\sim\mathcal{D}_{\text{demo}}} \left\| \mathcal{P}_\phi(s_t^{(i)}) - s_L^{(i)} \right\|_2^2 \tag{7}$$

Once trained, the goal predictor enables π^G to generalize to new tasks. Given a test-time state s^{test}, the predicted goal $\hat{g}^{\text{test}} = \mathcal{P}_\phi(s^{\text{test}})$ serves as the planning target for the agent $\pi = \pi^G(\cdot \mid s^{\text{test}}, \hat{g}^{\text{test}})$. The complete training pipeline of Cago is detailed in Algorithm 2.

Rationale Behind Cago's Design. Let $\mathcal{J}(\pi, \mathcal{M})$ and $\mathcal{J}(\pi^e, \mathcal{M})$ be the expected return of the agent's policy π and expert policy π^e in the real-world MDP $\mathcal{M}$. We want to bound their return difference:

$$\min_{\pi} \ |\mathcal{J}(\pi^e, \mathcal{M}) - \mathcal{J}(\pi, \mathcal{M})|, \tag{8}$$

Let R_{max} be the maximum of the reward with unknown dynamics: $R_{\text{max}} = \max_{(s,a)} \mathcal{R}(s,a)$ and $\rho^{\pi}_{\mathcal{M}}(s,a) = (1-\gamma)\sum_{t=0}^{\infty}\gamma^t P(s_t = s, a_t = a)$ be the discounted state-action visitation distribution of a policy π in the real world MDP $\mathcal{M}$. Suppose that the total variation of learned dynamics model $\widehat{\mathcal{M}}$ from the true transitions $\mathcal{M}$ is bounded such $\mathbb{D}_{\text{TV}}(\mathcal{T}(s,a), \widehat{\mathcal{T}}(s,a)) \le \alpha \quad \forall (s,a) \in \mathcal{S} \times \mathcal{A}$. According to previous work (Rafailov et al., 2021; DeMoss et al., 2023; Kolev et al., 2024), we have:

$$|\mathcal{J}(\pi^e, \mathcal{M}) - \mathcal{J}(\pi, \mathcal{M})| \le \underbrace{\alpha \frac{R_{\text{max}}}{(1-\gamma)^2}}_{\text{model prediction error}} + \underbrace{\frac{R_{\text{max}}}{1-\gamma}\mathbb{D}_{\text{TV}}\left(\rho^{\pi^e}_{\mathcal{M}}, \rho^{\pi}_{\widehat{\mathcal{M}}}\right)}_{\text{adaptation error}} \tag{9}$$

where $\rho^{\pi^e}_{\mathcal{M}}$ is the discounted visitation distribution of the expert policy and $\mathbb{D}_{\text{TV}}$ denote total variation distance. The model prediction error with respect to the true environment dynamics can be reduced by collecting more real-world data. In contrast, the adaptation error depends on the total variation distance between the distribution of trajectories generated by policy π under the learned world model $\widehat{\mathcal{M}}$ and the expert distribution under the true dynamics $\mathcal{M}$. Thus, the learning problem reduces to bounding the deviation between the behavior of the learned policy π under the learned model and the expert behavior under the true environment. To this end, given any H-step trajectory $(s_0, s_1, \ldots, s_H)$ sampled from expert demonstrations $\mathcal{D}_{\text{demo}}$, Cago encourages the agent to match expert behavior by rewarding it for reaching the final state $g = s_H$ starting from s_0 under the learned dynamics model $\widehat{\mathcal{M}}$ (Line 19 in Algorithm 2).

We further show that Cago effectively reduces the model prediction error by leveraging the BC explore policy $\pi^E = \pi^{\text{BC}}$ for data collection. In the following, we use $d_t^{\mathcal{M},\pi}$ to denote the marginal state-action distribution at time t induced by policy π in the environment $\mathcal{M}$. We assume $d_t^{\mathcal{D}_{\text{demo}}} \approx d_t^{\mathcal{M},\pi^e}$, where $\mathcal{D}_{\text{demo}}$ is the dataset of demonstrations generated by the expert π^e, and is sufficiently representative to approximate the true marginal distributions at each timestep. Assuming: (1) π^{BC} accurately approximates the expert policy in $\mathcal{D}_{\text{demo}}$, (2) the world model $\widehat{\mathcal{M}}$ is accurately trained on state transitions induced by π^{BC}, and (3) the learned policy π generates trajectories in $\widehat{\mathcal{M}}$ that closely match the expert's behavior, we can bound the model prediction error along the imagined rollouts generated by π under $\widehat{\mathcal{M}}$:

Theorem 1. *Let $\mathcal{M}$ denote the true dynamics model and $\widehat{\mathcal{M}}$ the learned model. Let π_{BC} be a behavior-cloned policy, and π a new policy. Let $\mathcal{D}_{demo}$ be a dataset of expert demonstrations from an unknown expert policy. Suppose that, for all $t = 0, 1, \ldots, T$, (1) Closeness of behavior cloning: $\mathbb{D}_{\text{TV}}\left(d_t^{\mathcal{M},\pi_{\text{BC}}}, d_t^{\mathcal{D}_{demo}}\right) \le \kappa$, (2) Model learning error under BC: $\mathbb{E}_{(s,a)\sim d_t^{\mathcal{M},\pi_{\text{BC}}}}\left[\mathbb{D}_{\text{TV}}\left(\mathcal{M}(\cdot \mid s,a), \widehat{\mathcal{M}}(\cdot \mid s,a)\right)\right] \le \mu$, and (3) Trajectory distribution closeness: $\mathbb{D}_{\text{TV}}\left(\rho^{\pi^e}_{\mathcal{M}}, \rho^{\pi}_{\widehat{\mathcal{M}}}\right) \le \nu$. Then for all $t = 0, 1, \ldots, T$, we have:*

$$\mathbb{E}_{(s,a)\sim d_t^{\widehat{\mathcal{M}},\pi}}\left[\mathbb{D}_{\text{TV}}\left(\mathcal{M}(\cdot \mid s,a), \widehat{\mathcal{M}}(\cdot \mid s,a)\right)\right] \le \mu + 2\kappa + 2\nu.$$

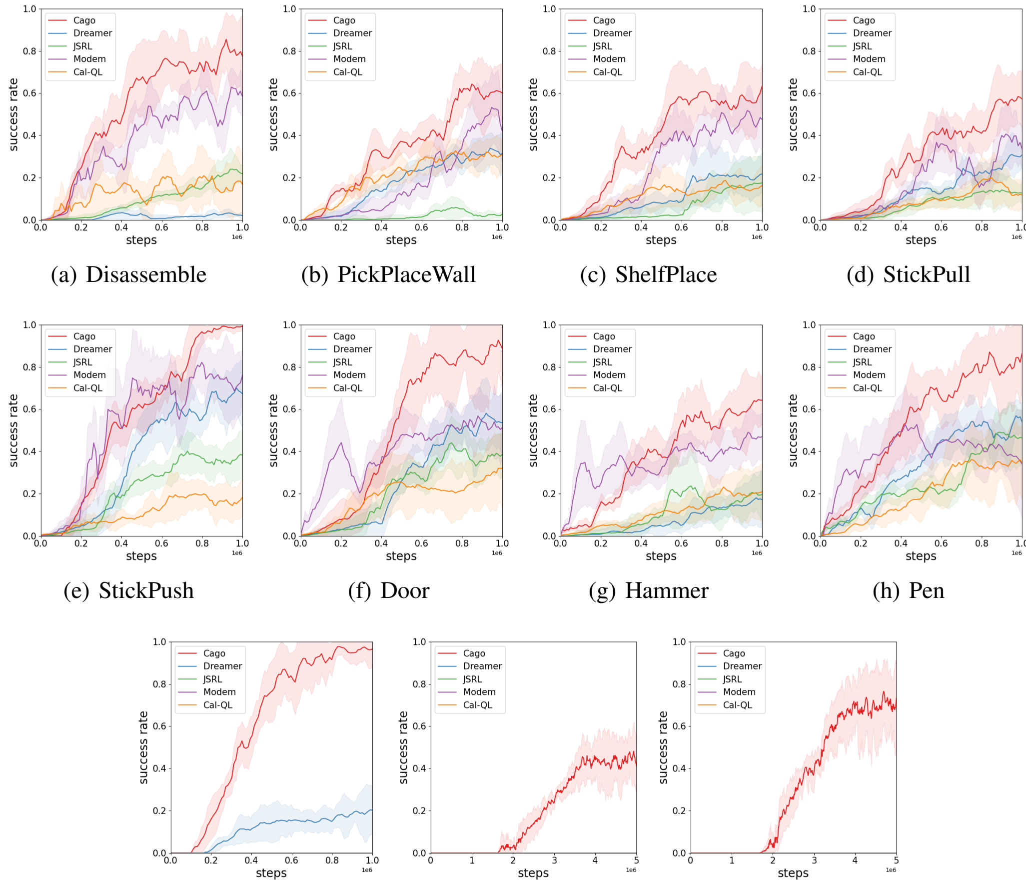
4 Experiments

We evaluate Cago across a diverse set of challenging robotic manipulation environments to address the following research questions: (Q1) Does Cago outperform existing imitation learning baselines that leverage demonstrations in alternative ways? (Q2) Can Cago effectively realize capability-aware goal sampling that aligns with the agent's learning progress? (Q3) How essential are the proposed capability-aware goal sampling and BC-Explorer components to the overall performance of Cago?

Environments. For our experiments, we evaluate and compare Cago against several baselines across three robot environment suites with sparse rewards: MetaWorld (Yu et al., 2020), Adroit (Rajeswaran et al., 2017), and Maniskill (Gu et al., 2023; Tao et al., 2025). We adopt the five "very hard" level environments from MetaWorld, as categorized by Seo et al. (2023): Shelf Place, Disassemble, Stick Pull, Stick Push, Pick Place Wall. These environments are considered the most challenging tasks in Metaworld, requiring precise robotic arm control with only sparse task completion rewards. We also use three dexterous hand manipulation tasks from the Adroit suite: Door, Hammer, Pen. To succeed in these three environments, the agent must perform fine-grained and intricate finger manipulations, enabling the grasping and movement of different objects. We also selected three challenging tasks from the ManiSkill benchmark: PegInsertionSide, StackCube, and PullCubeTool. The sparse-reward ManiSkill environments are the most difficult tasks in our benchmarks due to their high-dimensional state and action spaces. During training, we used only 10 demonstration trajectories per task for the MetaWorld and Adroit environments, and 20 demonstration trajectories per task for the ManiSkill environments. More details about each task can be found in Appendix E.

Baselines. Our approach is developed on top of the **Dreamer** framework (Hafner et al., 2019, 2020; Hu et al., 2023; Duan et al., 2024), making it a key model-based RL baseline for evaluating the performance gains of Cago. Jump-Start Reinforcement Learning (**JSRL**) (Uchendu et al., 2023) is a curriculum-based approach that leverages a guide-policy pretrained from offline data to guide early-stage exploration during online training. At the beginning of each training episode, the agent follows the guide-policy for a number of steps determined by curriculum progression, after which control is handed over to the online policy. In our JSRL implementation, due to the limited number of demonstrations available, training a reliable guide-policy becomes challenging. Therefore, we directly use the demonstration trajectories as the guide-policy. Specifically, we reset the environment to a demonstration initial state, enabling the agent to replicate expert behavior during the initial phase of each episode before switching to the online policy. Our JSRL implementation is also built on top of the Dreamer framework. **MoDem** (Hansen et al., 2022) represents one of the most efficient frameworks in the model-based RL literature. It pretrains its policy using a small set of demonstrations and repeatedly oversamples the demonstrations to train both the world model and the policy. We consider MoDem to be the strongest baseline due to its fast convergence and low data requirements. **Cal-QL** (Nakamoto et al., 2023) is a state-of-the-art algorithm following the offline-to-online RL paradigm. It uses demonstrations to pretrain the Q-function and applies calibration to mitigate performance drop when transitioning from offline to online learning phases. We additionally compare against two imitation learning baselines, **GAIL** (Ho and Ermon, 2016) and **PWIL** (Dadashi et al., 2020), which minimizes the divergence between the expert and agent's state distributions. We also include a comparison between Cago and **RLPD** (Ball et al., 2023), a well-tuned variant of SAC that leverages offline data when learning online. These results are included in Appendix F.1 to retain the clarify of Fig. 3.

Main Results. During training, we uniformly sample a demonstration and reset the environment using the same seed that was used to collect it. All baselines share the same seeds and demonstration data. To evaluate generalization, we test on unseen environment seeds. For these, Cago uses the goal predictor (Section 3.3) to infer goal conditions for the goal-conditioned policy π^G. Each method is evaluated on 100 held-out seeds, and we report the average success rate over these episodes. Figure 3 depicts the mean learning performance of Cago and all baselines in terms of the agent's task success rate averaged over 5 training seeds. On the MetaWorld very hard tasks, Cago consistently outperforms all the baselines in both final performance and learning efficiency. In the Adroit suite, although Modem exhibits rapid early learning due to its behavior cloning (BC) pretraining and oversampling strategy, Cago surpasses it in final performance after 1e6 environment interaction steps. Notably, Dreamer, which shares the same world model and policy architecture as Cago, performs significantly worse, underscoring the effectiveness of the capability-aware goal sampling strategy. Our JSRL baseline, based on the same world model architecture, adopts a uniform curriculum to reduce the



(a) Disassemble (b) PickPlaceWall (c) ShelfPlace (d) StickPull

(e) StickPush (f) Door (g) Hammer (h) Pen

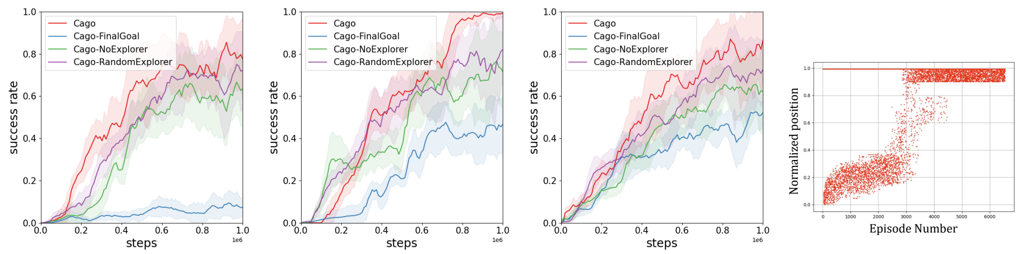
(i) PullCubeTool (j) PegInsertionSide (k) StackCube

Figure 3: Experiment results comparing Cago with the baselines over 5 random seeds. The solid line denotes the average success rate in *evaluation*, while the shaded region signifies the standard deviation.

guide-steps from demonstrations. It lags behind Cago in both learning speed and final success rate, further underscoring the benefits of our goal sampling strategy that adapts to the agent's evolving capabilities. In the ManiSkill environments, given the limited demonstrations, Cago stands out as the only method capable of attaining high success rates. While Cago is evaluated in state-based environments in Figure 3, we further assess its performance in *visual environments* in Appendix F.2, demonstrating that it achieves comparable results to those obtained with state-based inputs.

Capability-Aware Goal Distribution. To answer Q2, we visualize the progression of capability-aware goal sampling throughout the training process in the StickPush environment in Figure 4(d). Each red dot represents the normalized position of a sampled goal within a demonstration trajectory, with 0 indicating the start and 1.0 indicating the final demonstration state. Early in training, the agent predominantly samples goals at lower normalized positions, focusing on subgoals near the beginning of the trajectory that are within its current capabilities. As training advances, goal sampling gradually shifts toward higher normalized positions, indicating the agent's increasing ability to pursue more challenging goals closer to task completion. By continuously targeting goals just at the boundary of the agent's current capability, Cago facilitates efficient learning in sparse-reward, long-horizon tasks.

Ablation study. To answer Q3, we assess the individual contributions of (a) capability-aware goal sampling and (b) the BC-Explorer component to the overall performance. The first ablation, Cago-FinalGoal, retains only (b): it uses BC-based exploration but always selects the final observation from a demonstration in the goal phase of our Go-Explore sampling paradigm, ignoring the agent's current goal-reaching capability. The BC Explorer takes control from the goal-conditioned policy halfway through each rollout. The second ablation, Cago-NoExplorer, keeps only (a): it uses capability-aware



(a) Disassemble (b) StickPush (c) Pen (d) Sampling Progress

Figure 4: Figure(a),(b),(c) are the results of ablation study on the importance of each component of Cago over 5 seeds. Figure(d) shows the progress of capability-aware goal sampling in Stickpush.

goal sampling, but does not explore beyond the sampled goal with the BC Explorer. The third ablation, Cago-RandomExplorer, replaces the BC Explorer with a uniformly random policy during the exploration phase of our Go-Explore-style rollout strategy. We conduct the ablation study on the Disassemble and StickPush tasks from MetaWorld, and the Pen task from Adroit. As shown in Figure 4, removing capability-aware goal sampling significantly degrades performance. Without it, the agent often enters the Explore phase from states far outside the demonstration region, making it difficult for the BC-Explorer to make meaningful progress. The BC-Explorer itself is also crucial, as it accelerates learning by generating high-quality exploratory rollouts.

5 Related Work

Demonstrations are a key tool for improving the efficiency and stability of RL, with prior work integrating them across various stages of the RL pipeline (Arulkumaran and Lillrank, 2024; Nair et al., 2018). A prominent approach uses demonstrations for direct learning via Behavior Cloning (BC) and its variants (Bain and Sammut, 1995; Torabi et al., 2018). The introduction of the Generative Adversarial Imitation Learning (GAIL) algorithm (Ho and Ermon, 2016) has driven significant advances in scalable deep imitation learning methods (Fu et al., 2017; Ghasemipour et al., 2020; Kostrikov et al., 2018; Jena et al., 2020; Finn et al., 2016; Blondé and Kalousis, 2019; Orsini et al., 2021; Eysenbach et al., 2021). Beyond adversarial approaches, several imitation learning algorithms aim to match the state action distributions of the expert and the agent through non-adversarial techniques, such as non-parametric models (Kim and Park, 2018), random network distillation (Wang et al., 2019), support estimation (Brantley et al., 2020), Wasserstein distance minimization (Dadashi et al., 2020), and moment matching (Swamy et al., 2021). Demonstrations have also been used to improve critic learning. Conservative Q-learning (CQL) (Kumar et al., 2020) and Cal-QL (Nakamoto et al., 2023) regularize Q-values using demonstration data to better estimate out-of-distribution actions. Cago introduces a novel use of demonstrations by treating them as a structured roadmap for building an adaptive curriculum, scaffolding the agent's learning to enable steady progress toward solving the full task. Several prior works have explored curriculum design in imitation learning. Yengera et al. (2021); Tao et al. (2024) introduce difficulty scores to rank demonstrations, offering a theoretical framework for selecting optimal trajectories to scaffold learning. Task Phasing (Bajaj et al., 2023) automatically extracts curriculum phases from demonstrations and dynamically transitions the agent through them during training. In contrast, Cago employs a goal-level curriculum that incrementally samples intermediate goals from demonstrations based on the agent's evolving capabilities. For a broader discussion of related work on goal-conditioned RL and state resets, see Appendix C.

6 Conclusion

We introduce Cago, a novel method that leverages demonstrations in a dynamic, goal-guided manner to tackle exploration challenges in sparse-reward environments. By continuously monitoring the agent's capabilities, Cago constructs an adaptive curriculum that incrementally samples intermediate goals from demonstrations, effectively scaffolding learning and enabling steady progress toward solving the full task. Extensive experiments show consistent improvements over baselines. A key limitation of Cago is its reliance on high-quality demonstrations. In future work, we aim to extend Cago to handle suboptimal offline data, enabling robust learning in more practical scenarios.

Reproducibility Statement

The code for Cago is available in the supplemental material. For hyperparameter settings, please refer to Appendix H.

References

Abbeel, P. and Ng, A. Y. (2004). Apprenticeship learning via inverse reinforcement learning. In Brodley, C. E., editor, *Machine Learning, Proceedings of the Twenty-first International Conference (ICML 2004), Banff, Alberta, Canada, July 4-8, 2004*, volume 69 of *ACM International Conference Proceeding Series*. ACM.

Andrychowicz, M., Wolski, F., Ray, A., Schneider, J., Fong, R., Welinder, P., McGrew, B., Tobin, J., Pieter Abbeel, O., and Zaremba, W. (2017). Hindsight experience replay. *Advances in neural information processing systems*, 30.

Arulkumaran, K. and Lillrank, D. O. (2024). A pragmatic look at deep imitation learning. In *Asian Conference on Machine Learning*, pages 58–73. PMLR.

Bain, M. and Sammut, C. (1995). A framework for behavioural cloning. In *Machine intelligence 15*, pages 103–129.

Bajaj, V., Sharon, G., and Stone, P. (2023). Task phasing: Automated curriculum learning from demonstrations. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 33, pages 542–550.

Ball, P. J., Smith, L., Kostrikov, I., and Levine, S. (2023). Efficient online reinforcement learning with offline data. In *International Conference on Machine Learning*, pages 1577–1594. PMLR.

Bellemare, M., Srinivasan, S., Ostrovski, G., Schaul, T., Saxton, D., and Munos, R. (2016). Unifying count-based exploration and intrinsic motivation. *Advances in neural information processing systems*, 29.

Blondé, L. and Kalousis, A. (2019). Sample-efficient imitation learning via generative adversarial nets. In Chaudhuri, K. and Sugiyama, M., editors, *The 22nd International Conference on Artificial Intelligence and Statistics, AISTATS 2019, 16-18 April 2019, Naha, Okinawa, Japan*, volume 89 of *Proceedings of Machine Learning Research*, pages 3138–3148. PMLR.

Brantley, K., Sun, W., and Henaff, M. (2020). Disagreement-regularized imitation learning. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Burda, Y., Edwards, H., Storkey, A., and Klimov, O. (2018). Exploration by random network distillation. *arXiv preprint arXiv:1810.12894*.

Dadashi, R., Hussenot, L., Geist, M., and Pietquin, O. (2020). Primal wasserstein imitation learning. *arXiv preprint arXiv:2006.04678*.

DeMoss, B., Duckworth, P., Hawes, N., and Posner, I. (2023). Ditto: Offline imitation learning with world models. *arXiv preprint arXiv:2302.03086*.

Duan, Y., Mao, W., and Zhu, H. (2024). Learning world models for unconstrained goal navigation. *Advances in Neural Information Processing Systems*, 37:59236–59265.

Ecoffet, A., Huizinga, J., Lehman, J., Stanley, K. O., and Clune, J. (2019). Go-explore: a new approach for hard-exploration problems. *arXiv preprint arXiv:1901.10995*.

Eysenbach, B., Levine, S., and Salakhutdinov, R. R. (2021). Replacing rewards with examples: Example-based policy search via recursive classification. *Advances in Neural Information Processing Systems*, 34:11541–11552.

Finn, C., Christiano, P. F., Abbeel, P., and Levine, S. (2016). A connection between generative adversarial networks, inverse reinforcement learning, and energy-based models. *CoRR*, abs/1611.03852.

Florensa, C., Held, D., Geng, X., and Abbeel, P. (2018). Automatic goal generation for reinforcement learning agents. In *International conference on machine learning*, pages 1515–1528. PMLR.

Fu, J., Luo, K., and Levine, S. (2017). Learning robust rewards with adversarial inverse reinforcement learning. *arXiv preprint arXiv:1710.11248*.

Ghasemipour, S. K. S., Zemel, R., and Gu, S. (2020). A divergence minimization perspective on imitation learning methods. In *Conference on robot learning*, pages 1259–1277. PMLR.

Ghosh, D., Gupta, A., Reddy, A., Fu, J., Devin, C., Eysenbach, B., and Levine, S. (2019). Learning to reach goals via iterated supervised learning. *arXiv preprint arXiv:1912.06088*.

Gu, J., Xiang, F., Li, X., Ling, Z., Liu, X., Mu, T., Tang, Y., Tao, S., Wei, X., Yao, Y., et al. (2023). Maniskill2: A unified benchmark for generalizable manipulation skills. *arXiv preprint arXiv:2302.04659*.

Hafner, D., Lillicrap, T., Ba, J., and Norouzi, M. (2019). Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv:1912.01603*.

Hafner, D., Lillicrap, T., Norouzi, M., and Ba, J. (2020). Mastering atari with discrete world models. *arXiv preprint arXiv:2010.02193*.

Hafner, D., Pasukonis, J., Ba, J., and Lillicrap, T. (2023). Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*.

Hansen, N., Lin, Y., Su, H., Wang, X., Kumar, V., and Rajeswaran, A. (2022). Modem: Accelerating visual model-based reinforcement learning with demonstrations. *arXiv preprint arXiv:2212.05698*.

Ho, J. and Ermon, S. (2016). Generative adversarial imitation learning. *Advances in neural information processing systems*, 29.

Hosu, I. and Rebedea, T. (2016). Playing atari games with deep reinforcement learning and human checkpoint replay. *CoRR*, abs/1607.05077.

Hu, E. S., Chang, R., Rybkin, O., and Jayaraman, D. (2023). Planning goals for exploration. *arXiv preprint arXiv:2303.13002*.

Jena, R., Agrawal, S., and Sycara, K. P. (2020). Addressing reward bias in adversarial imitation learning with neutral reward functions. *CoRR*, abs/2009.09467.

Kim, K. and Park, H. S. (2018). Imitation learning via kernel mean embedding. In McIlraith, S. A. and Weinberger, K. Q., editors, *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 3415–3422. AAAI Press.

Kolev, V., Rafailov, R., Hatch, K., Wu, J., and Finn, C. (2024). Efficient imitation learning with conservative world models. In *6th Annual Learning for Dynamics & Control Conference*, pages 1777–1790. PMLR.

Kostrikov, I., Agrawal, K. K., Dwibedi, D., Levine, S., and Tompson, J. (2018). Discriminator-actor-critic: Addressing sample inefficiency and reward bias in adversarial imitation learning. *arXiv preprint arXiv:1809.02925*.

Kumar, A., Zhou, A., Tucker, G., and Levine, S. (2020). Conservative q-learning for offline reinforcement learning. *Advances in neural information processing systems*, 33:1179–1191.

Liu, M., Zhu, M., and Zhang, W. (2022). Goal-conditioned reinforcement learning: Problems and solutions. *arXiv preprint arXiv:2201.08299*.

McCarthy, R., Wang, Q., and Redmond, S. J. (2021). Imaginary hindsight experience replay: Curious model-based learning for sparse reward tasks. *arXiv preprint arXiv:2110.02414*.

Mendonca, R., Rybkin, O., Daniilidis, K., Hafner, D., and Pathak, D. (2021). Discovering and achieving goals via world models. *Advances in Neural Information Processing Systems*, 34:24379–24391.

Nair, A., McGrew, B., Andrychowicz, M., Zaremba, W., and Abbeel, P. (2018). Overcoming exploration in reinforcement learning with demonstrations. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 6292–6299. IEEE.

Nakamoto, M., Zhai, S., Singh, A., Sobol Mark, M., Ma, Y., Finn, C., Kumar, A., and Levine, S. (2023). Cal-ql: Calibrated offline rl pre-training for efficient online fine-tuning. *Advances in Neural Information Processing Systems*, 36:62244–62269.

Orsini, M., Raichuk, A., Hussenot, L., Vincent, D., Dadashi, R., Girgin, S., Geist, M., Bachem, O., Pietquin, O., and Andrychowicz, M. (2021). What matters for adversarial imitation learning? In Ranzato, M., Beygelzimer, A., Dauphin, Y. N., Liang, P., and Vaughan, J. W., editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 14656–14668.

Oudeyer, P.-Y., Kaplan, F., and Hafner, V. V. (2007). Intrinsic motivation systems for autonomous mental development. *IEEE transactions on evolutionary computation*, 11(2):265–286.

Pathak, D., Agrawal, P., Efros, A. A., and Darrell, T. (2017). Curiosity-driven exploration by self-supervised prediction. In *International conference on machine learning*, pages 2778–2787. PMLR.

Peng, X. B., Abbeel, P., Levine, S., and van de Panne, M. (2018). Deepmimic: example-guided deep reinforcement learning of physics-based character skills. *ACM Trans. Graph.*, 37(4):143.

Pitis, S., Chan, H., Zhao, S., Stadie, B., and Ba, J. (2020). Maximum entropy gain exploration for long horizon multi-goal reinforcement learning. In *International Conference on Machine Learning*, pages 7750–7761. PMLR.

Plappert, M., Andrychowicz, M., Ray, A., McGrew, B., Baker, B., Powell, G., Schneider, J., Tobin, J., Chociej, M., Welinder, P., et al. (2018). Multi-goal reinforcement learning: Challenging robotics environments and request for research. *arXiv preprint arXiv:1802.09464*.

Pong, V. H., Dalal, M., Lin, S., Nair, A., Bahl, S., and Levine, S. (2019). Skew-fit: State-covering self-supervised reinforcement learning. *arXiv preprint arXiv:1903.03698*.

Rafailov, R., Yu, T., Rajeswaran, A., and Finn, C. (2021). Visual adversarial imitation learning using variational models. *Advances in Neural Information Processing Systems*, 34:3016–3028.

Rajeswaran, A., Kumar, V., Gupta, A., Vezzani, G., Schulman, J., Todorov, E., and Levine, S. (2017). Learning complex dexterous manipulation with deep reinforcement learning and demonstrations. *arXiv preprint arXiv:1709.10087*.

Resnick, C., Raileanu, R., Kapoor, S., Peysakhovich, A., Cho, K., and Bruna, J. (2018). Backplay: "man muss immer umkehren". *CoRR*, abs/1807.06919.

Salimans, T. and Chen, R. (2018). Learning montezuma's revenge from a single demonstration. *CoRR*, abs/1812.03381.

Sekar, R., Rybkin, O., Daniilidis, K., Abbeel, P., Hafner, D., and Pathak, D. (2020). Planning to explore via self-supervised world models. In *International conference on machine learning*, pages 8583–8592. PMLR.

Seo, Y., Hafner, D., Liu, H., Liu, F., James, S., Lee, K., and Abbeel, P. (2023). Masked world models for visual control. In *Conference on Robot Learning*, pages 1332–1344. PMLR.

Shyam, P., Jaśkowski, W., and Gomez, F. (2019). Model-based active exploration. In *International conference on machine learning*, pages 5779–5788. PMLR.

Swamy, G., Choudhury, S., Bagnell, J. A., and Wu, S. (2021). Of moments and matching: A game-theoretic framework for closing the imitation gap. In Meila, M. and Zhang, T., editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 10022–10032. PMLR.

Tao, S., Shukla, A., Chan, T., and Su, H. (2024). Reverse forward curriculum learning for extreme sample and demo efficiency. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Tao, S., Xiang, F., Shukla, A., Qin, Y., Hinrichsen, X., Yuan, X., Bao, C., Lin, X., Liu, Y., kai Chan, T., Gao, Y., Li, X., Mu, T., Xiao, N., Gurha, A., Rajesh, V. N., Choi, Y. W., Chen, Y.-R., Huang, Z., Calandra, R., Chen, R., Luo, S., and Su, H. (2025). Maniskill3: Gpu parallelized robotics simulation and rendering for generalizable embodied ai. *Robotics: Science and Systems*.

Torabi, F., Warnell, G., and Stone, P. (2018). Behavioral cloning from observation. *arXiv preprint arXiv:1805.01954*.

Trott, A., Zheng, S., Xiong, C., and Socher, R. (2019). Keeping your distance: Solving sparse reward tasks using self-balancing shaped rewards. *Advances in Neural Information Processing Systems*, 32.

Uchendu, I., Xiao, T., Lu, Y., Zhu, B., Yan, M., Simon, J., Bennice, M., Fu, C., Ma, C., Jiao, J., et al. (2023). Jump-start reinforcement learning. In *International Conference on Machine Learning*, pages 34556–34583. PMLR.

Wang, R., Ciliberto, C., Amadori, P. V., and Demiris, Y. (2019). Random expert distillation: Imitation learning via expert policy support estimation. In Chaudhuri, K. and Salakhutdinov, R., editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 6536–6544. PMLR.

Yengera, G., Devidze, R., Kamalaruban, P., and Singla, A. (2021). Curriculum design for teaching via demonstrations: theory and applications. *Advances in Neural Information Processing Systems*, 34:10496–10509.

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

Zhang, Y., Abbeel, P., and Pinto, L. (2020). Automatic curriculum learning through value disagreement. *Advances in Neural Information Processing Systems*, 33:7648–7659.

Zhu, Y., Wang, Z., Merel, J., Rusu, A. A., Erez, T., Cabi, S., Tunyasuvunakool, S., Kramár, J., Hadsell, R., de Freitas, N., and Heess, N. (2018). Reinforcement and imitation learning for diverse visuomotor skills. In Kress-Gazit, H., Srinivasa, S. S., Howard, T., and Atanasov, N., editors, *Robotics: Science and Systems XIV, Carnegie Mellon University, Pittsburgh, Pennsylvania, USA, June 26-30, 2018*.

Appendix

A Proof of Theorem 1

Theorem 1. Let $\mathcal{M}$ denote the true dynamics model and $\widehat{\mathcal{M}}$ the learned model. Let π_{BC} be a behavior-cloned policy, and π a new policy. Let $\mathcal{D}_{\text{demo}}$ be a dataset of expert demonstrations from an unknown expert policy. Suppose that, for all $t = 0, 1, \ldots, T$, (1) Closeness of behavior cloning: $\mathbb{D}_{TV}\left(d_t^{\mathcal{M},\pi_{\mathrm{BC}}}, d_t^{\mathcal{D}_{\text{demo}}}\right) \leq \kappa$, (2) Model learning error under BC: $\mathbb{E}_{(s,a)\sim d_t^{\mathcal{M},\pi_{\mathrm{BC}}}}\left[\mathbb{D}_{TV}\left(\mathcal{M}(\cdot \mid s,a), \widehat{\mathcal{M}}(\cdot \mid s,a)\right)\right] \leq \mu$, and (3) Trajectory distribution closeness: $\mathbb{D}_{TV}\left(\rho_{\mathcal{M}}^{\pi^e}, \rho_{\widehat{\mathcal{M}}}^{\pi}\right) \leq \nu$. Then for all $t = 0, 1, \ldots, T$, we have:

$$\mathbb{E}_{(s,a)\sim d_t^{\widehat{\mathcal{M}},\pi}}\left[\mathbb{D}_{TV}\left(\mathcal{M}(\cdot \mid s,a), \widehat{\mathcal{M}}(\cdot \mid s,a)\right)\right] \leq \mu + 2\kappa + 2\nu.$$

Proof. By triangle inequality on expectations and total variation:

$$\begin{aligned}
&\mathbb{E}_{(s,a)\sim d_t^{\widehat{\mathcal{M}},\pi}}\left[\mathbb{D}_{\mathrm{TV}}\left(\mathcal{M}(\cdot \mid s,a), \widehat{\mathcal{M}}(\cdot \mid s,a)\right)\right] \\
&\leq \mathbb{E}_{(s,a)\sim d_t^{\mathcal{D}_{\text{demo}}}}\left[\mathbb{D}_{\mathrm{TV}}\left(\mathcal{M}(\cdot \mid s,a), \widehat{\mathcal{M}}(\cdot \mid s,a)\right)\right] + 2\ \mathbb{D}_{\mathrm{TV}}\left(d_t^{\widehat{\mathcal{M}},\pi}, d_t^{\mathcal{D}_{\text{demo}}}\right).
\end{aligned}$$

We now bound each term separately:

- For the first term, apply triangle inequality again:

  $$\begin{aligned}
  \mathbb{E}_{(s,a)\sim d_t^{\mathcal{D}_{\text{demo}}}}\left[\mathbb{D}_{\mathrm{TV}}\left(\mathcal{M}(\cdot \mid s,a), \widehat{\mathcal{M}}(\cdot \mid s,a)\right)\right] &\leq \mathbb{E}_{(s,a)\sim d_t^{\mathcal{M},\pi_{\mathrm{BC}}}}\left[\mathbb{D}_{\mathrm{TV}}\left(\mathcal{M}(\cdot \mid s,a), \widehat{\mathcal{M}}(\cdot \mid s,a)\right)\right] \\
  &\quad + 2\ \mathbb{D}_{\mathrm{TV}}\left(d_t^{\mathcal{D}_{\text{demo}}}, d_t^{\mathcal{M},\pi_{\mathrm{BC}}}\right) \\
  &\leq \mu + 2\kappa,
  \end{aligned}$$

  using assumptions (1) and (2).

- For the second term, use that marginal total variation is bounded by trajectory total variation:

  $$\mathbb{D}_{\mathrm{TV}}\left(d_t^{\widehat{\mathcal{M}},\pi}, d_t^{\mathcal{D}_{\text{demo}}}\right) \leq \mathbb{D}_{\mathrm{TV}}\left(\rho^{\widehat{\mathcal{M}},\pi}, \rho^{\mathcal{M},\pi^e}\right) \leq \nu,$$

  by assumption (3).

Combining:

$$\mathbb{E}_{(s,a)\sim d_t^{\widehat{\mathcal{M}},\pi}}\left[\mathbb{D}_{\mathrm{TV}}\left(\mathcal{M}(\cdot \mid s,a), \widehat{\mathcal{M}}(\cdot \mid s,a)\right)\right] \leq \mu + 2\kappa + 2\nu.$$

□

B Extended Background

B.1 Dreamer World Model

The RSSM consists of an encoder, a recurrent model, a representation model, a transition predictor, and a decoder, as formulated in Equation 10. And it employs an end-to-end training methodology, where its parameters are jointly optimized based on the loss functions of various components, including dynamic transition prediction, reward prediction, and observation encoding-decoding. These components often operate in a latent space rather than the original observation space, as encoded by the World Model. Therefore, during end-to-end training, the losses of all components indirectly optimize the latent space.

The encoder f_E encodes the input state x_t into a embed state e_t, which is then fed with the deterministic state h_t into the representation model q_φ to generate the posterior state z_t. The transition predictor p_φ predicts the prior state $\hat{z}_t$ based on the deterministic state h_t without access to the current input state x_t. Using the concatenation of either (h_t, z_t) or $(h_t, \hat{z}_t)$ as input, the recurrent transition function $f\varphi$ iteratively updates the deterministic state h_t with given action a_t.

$$
\begin{aligned}
\text{Encoder:} \quad & e_t = f_E(e_t|x_t) \\
\text{Recurrent model:} \quad & h_t = f_\varphi(h_{t-1}, z_{t-1}, a_{t-1}) \\
\text{Representation model:} \quad & z_t \sim q_\varphi(z_t|h_t, e_t) \\
\text{Transition predictor:} \quad & \hat{z}_t \sim p_\varphi(\hat{z}_t|h_t) \\
\text{Decoder:} \quad & \hat{x}_t \sim f_D(\hat{x}_t|h_t, z_t)
\end{aligned} \tag{10}
$$

B.2 Temporal Distance Training in LEXA

The goal-reaching reward r^G is defined by the self-supervised temporal distance objective (Mendonca et al., 2021) which aims to minimize the number of action steps needed to transition from the current state to a goal state within imagined rollouts. We use b_t to denote the concatenate of the deterministic state h_t and the posterior state z_t at time step t.

$$b_t = (h_t, z_t) \tag{11}$$

The temporal distance D_t is trained by sampling pairs of imagined states b_t, b_{t+k} from imagined rollouts and predicting the action steps number k between the embedding of them, with a predicted embedding $\hat{e}_t$ from b_t to approximate the true embedding e_t of the observation x_t.

$$\text{Predicted embedding:} \qquad emb(b_t) = \hat{e}_t \approx e_t, \qquad \text{where} \quad e_t = f_E(x_t) \tag{12}$$

$$\text{Temporal distance: } D_t(\hat{e}_t, \hat{e}_{t+k}) \approx k/H \qquad \text{where} \quad \hat{e}_t = emb(b_t) \quad \hat{e}_{t+k} = emb(b_{t+k}) \tag{13}$$

$$r_t^G(b_t, b_{t+k}) = -D_t(\hat{e}_t, \hat{e}_{t+k}) \tag{14}$$

C Extended Related Work

Goal-conditioned reinforcement learning (GCRL) has emerged as a promising paradigm for training agents to achieve diverse tasks by conditioning on desired goals. However, in many real-world settings, agents face sparse reward environments—settings where rewards are only provided upon successful task completion—making exploration especially difficult (Liu et al., 2022; Plappert et al., 2018; Ghosh et al., 2019). Without frequent learning signals, agents struggle to discover successful behaviors through random trial and error (Florensa et al., 2018; Trott et al., 2019; McCarthy et al., 2021). To address this, prior works have introduced various strategies for efficient exploration. Hindsight Experience Replay (Andrychowicz et al., 2017) relabels failed trajectories with achieved goals to create meaningful learning signals. Other methods rely on intrinsic motivation or curiosity-based objectives, such as prediction error (Pathak et al., 2017; Zhang et al., 2020) or novelty signals (Oudeyer et al., 2007; Bellemare et al., 2016) to incentivize exploration in the absence of extrinsic rewards. Approaches like Random Network Distillation (Burda et al., 2018) and Go-Explore (Ecoffet et al., 2019) further enhance state coverage by identifying rarely visited regions of the state space. Recent advances also integrate planning with learned dynamics models (Shyam et al., 2019; Sekar et al., 2020; Hu et al., 2023) or optimize goal proposals based on entropy or trajectory diversity (Pong et al., 2019; Pitis et al., 2020). Despite these innovations, learning in sparse-reward environments remains a fundamental challenge, especially in high-dimensional or long-horizon tasks, driving continued research in both goal-setting strategies and efficient exploration.

To address the inherent exploration difficulties in sparse-reward reinforcement learning environments, a number of methods have explored the use of expert demonstration states as informative priors for guiding agent learning. A common strategy involves resetting the agent from states sampled along demonstration trajectories (Nair et al., 2018; Peng et al., 2018; Hosu and Rebedea, 2016), enabling the agent to experience these expert regions of the state space without exhaustive exploration. Some approaches employ structured curricula to sequence these resets, either through manually designed progressions (Zhu et al., 2018) or automated strategies such as reverse curricula, which gradually increase the exploration horizon by training the agent from goal states backward (Resnick et al., 2018; Salimans and Chen, 2018; Tao et al., 2024). While effective in simulation, these methods rely on the ability to precisely reset the environment to arbitrary demonstration states—a requirement that poses significant challenges in physical systems, where replicating exact configurations, including latent dynamics like joint velocities, is often infeasible. Rather than forcibly placing the agent into expert states—a strong form of intervention—Cago encourages the agent to actively reach intermediate goals that are sampled to match its current competence. This self-directed learning process allows the agent to internalize problem-solving skills more effectively, promoting deeper understanding through its own attempts rather than directly resetting. While demonstrations provide useful guidance, true mastery requires the agent to explore and overcome challenges through trial and error.

D Limitations and Future Work

While Cago shows promising results in leveraging demonstrations to improve exploration and learning efficiency in sparse-reward, long-horizon tasks, several limitations remain. First, our method assumes access to high-quality demonstration trajectories that adequately cover the task space. In scenarios where demonstrations are noisy, suboptimal, or limited in diversity, the effectiveness of goal selection may degrade.

In future work, we aim to explore ways to relax the reliance on high-quality demonstrations by integrating learning from suboptimal or partial demonstrations. Additionally, incorporating uncertainty-aware models for both agent capability estimation and subgoal selection could improve robustness and adaptability. Extending Cago to multi-agent or real-world robotics scenarios is another promising direction, where the complexity of coordination and physical constraints introduces new challenges for efficient demonstration utilization and goal guidance.

E Environment Detail

We evaluate and compare Cago against several baselines across three robot environment suites using sparse rewards: MetaWorld (Yu et al., 2020), Adroit (Rajeswaran et al., 2017), and Maniskill (Gu et al., 2023; Tao et al., 2025). In this section, we provide more details about each benchmark and the specific experimental setup.

E.1 MetaWorld

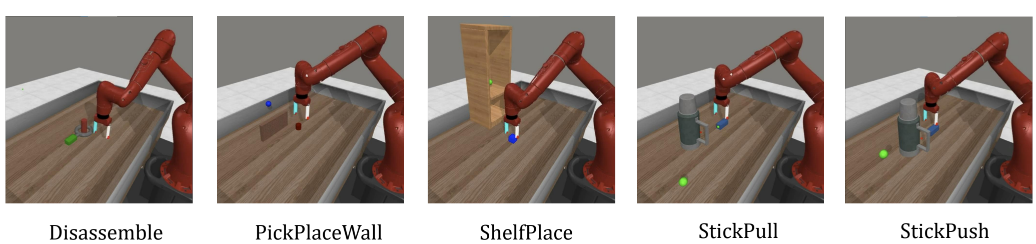
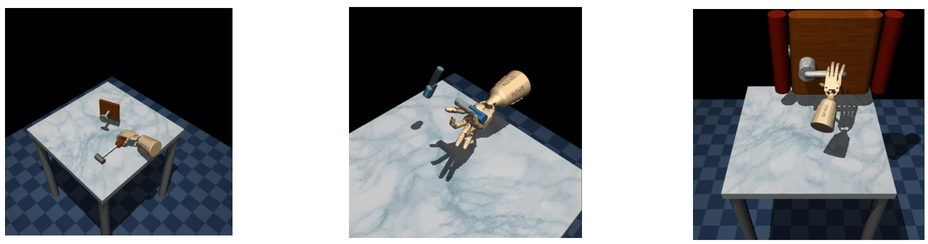


Figure 5: 5 very hard level environments from MetaWorld

MetaWorld (Yu et al., 2020) is a widely used benchmark suite designed to evaluate the generalization and manipulation capabilities of reinforcement learning algorithms in robotic control tasks. It consists of a diverse collection of continuous control environments simulated using the MuJoCo physics engine. Each task requires a robotic arm to interact with objects in the scene to achieve goal-directed behavior under sparse or dense reward settings. MetaWorld includes 50 distinct tasks of varying difficulty, ranging from simple reaching to complex object manipulation. In our experiments, we focus on the "very hard" subset of tasks identified in prior studies (Seo et al., 2023; Hansen et al., 2022), which are characterized by sparse rewards, delayed feedback, and the need for precise low-level control, making them particularly suitable for benchmarking sample efficiency and generalization in demonstration-augmented learning frameworks. The five very hard tasks we choose are: Shelf Place, Disassemble, StickPull, Stick Push, Pick Place Wall. Shelf Place: The agent must grasp an object and place it accurately onto a shelf, requiring precise vertical and lateral arm control. Disassemble: The task involves picking a nut out of the a peg, demanding a strong grasp and directional pulling motion. Stick Pull: The robot needs to grasp a stick and pull a bottle, requiring fine force control and coordinated motion. Stick Push: The goal is to grasp a stick and push a bottle, emphasizing controlled contact and alignment with the target location. Pick Place Wall: The agent must pick up an object and place it over a wall barrier onto a specified target location, combining lifting, positioning, and obstacle avoidance. We use the L2-distance to calculate the similarity between observations to judge if the agent has reached the demonstration observation. The threshold for similarity is 0.05. We use 10 demonstrations for training.

E.2 Adroit



Hammer Pen Door

Figure 6: 3 environments from Adroit Suite

The Adroit Suite (Rajeswaran et al., 2017) is a set of high-dimensional dexterous hand manipulation tasks that emphasize fine motor control, contact-rich interactions, and sparse-reward learning. It is built upon a 24-DoF ShadowHand robotic hand, presenting significant challenges in both control and generalization. Each task requires the agent to coordinate multiple fingers and joints to manipulate objects with high precision under partial observability and complex dynamics. We use three environments from this suit. Hammer: The agent must grasp a hammer and use it to drive a nail into a box. This task demands stable object manipulation, precise tool orientation, and effective force transmission. Pen: The objective is to reorient and position a pen in an assigned direction. It requires careful control of finger articulation and rotational dexterity. Door: The task involves unlatching and opening a door by manipulating the handle and applying a pulling motion. It tests the agent's ability to perform multi-stage interactions and coordinate wrist and finger movement to exert torque in the correct direction. We use the Mean square error between images rendered to calculate the similarity to judge if the agent has reached the demonstration observation. Each rendered image will be reshaped to a size (100,100,3). We use 10 demonstrations for training.

E.3 Maniskill

ManiSkill (Gu et al., 2023; Tao et al., 2025) is a comprehensive benchmark suite designed to evaluate generalizable robotic manipulation skills in simulation, emphasizing real-world task diversity, object variety, and generalization across instances. It provides high-quality 3D environments with continuous control, supporting both visual and proprioceptive observations. The benchmark is particularly challenging under sparse reward settings, as tasks often require multi-step reasoning, long-horizon planning, and precise control to accomplish. We pick three environments from this benchmark. PullCubeTool: Given an L-shaped tool that is within the reach of the robot, the agent needs to leverage the tool to pull a cube that is out of it's reach. PegInsertionSide: The robot must align and insert a peg into a side-entry slot. The task demands precise pose estimation, spatial reasoning, and careful control to avoid misalignment or jamming. StackCube: This task involves picking up a cube and accurately stacking it on top of another. We use the Mean square error between images rendered to calculate the similarity to judge if the agent has reached the demonstration observation. Each rendered image will also be reshaped to a size (100,100,3). We use 20 demonstrations for

Figure 7: 3 environments from Maniskill

training. On StackCube and PegInsertionSide, we scale up the position(x,y,z) 10 times and normalize the degree of griper opening for more stable learning. We set the clearance of the hole to 0.01 in PegInsertionSide so that the peg could be inserted more easily.

F More Experiments

F.1 More Baselines

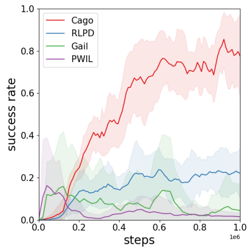
Generative Adversarial Imitation Learning (**GAIL**) (Ho and Ermon, 2016; Arulkumaran and Lillrank, 2024) adopts an adversarial learning framework where a discriminator is trained to differentiate between expert and agent trajectories; the discriminator's output is then used as the reward signal for the agent. Primal Wasserstein Imitation Learning (**PWIL**) (Dadashi et al., 2020; Arulkumaran and Lillrank, 2024) formulates imitation as a primal optimization problem that minimizes the Wasserstein distance between expert and agent trajectory distributions. It constructs a shaped reward function directly from this distance, encouraging the agent to produce expert-like behaviors. We also compapre Cago with Reinforcement Learning with Prior Data (**RLPD**) (Ball et al., 2023), a state-of-the-art baseline improving the efficiency of online reinforcement learning by leveraging offline data. We run these baselines over 5 random experimental seeds and report the average success rate.

As the results shown in Figure 8, Cago generally outperforms the baseline approaches (RLPD, GAIL, and PWIL) across a diverse set of manipulation tasks. In tasks such as Disassemble, StickPull, Hammer, and Pen, Cago demonstrates significantly faster convergence and higher final success rates, indicating its superior learning efficiency and robustness. Particularly in Maniskill environments, Cago is the only method that achieves meaningful learning progress, while all baselines fail to get any success, highlighting the importance of capability-aware goal sampling in challenging, sparse-reward environments. Although GAIL achieves the best performance in the ShelfPlace environment, this success is not representative of its overall effectiveness. In all other tasks, GAIL performs poorly, exhibiting highly unstable learning process and low final success rates.

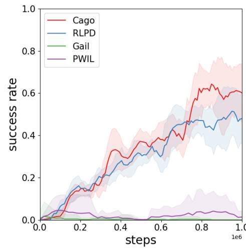
F.2 Visual Input Experiments

While Cago is primarily evaluated in state-based environments, we further assess its applicability to high-dimensional visual settings. Specifically, we extend our framework to raw pixel observations by replacing the vector-based states and goals with RGB image inputs of size (64, 64, 3), resulting in a variant referred to as Cago-Visual. In this setting, both the policy and the goal predictor $\mathcal{P}_\phi$ operate on image representations. We benchmark Cago-Visual against Modem-Visual, a baseline that similarly utilizes image-based observations. As shown in Figure 9, Cago-Visual not only retains performance similar to the original Cago, but also consistently outperforms Modem-Visual, highlighting the robustness of our method in visual domains.

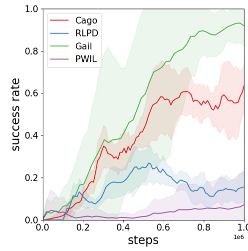
To evaluate the generalization ability of our visual goal predictor $\mathcal{P}\phi$, we visualize its predicted goal images given initial observations at test time in Figure 10. For each task, we compare the predicted goal (middle column) to the ground-truth final observation from a demonstration trajectory (right column). Importantly, these demonstrations are collected from unseen seeds that were never used during training. These results show that $\mathcal{P}\phi$ is capable of accurately inferring the final goal state



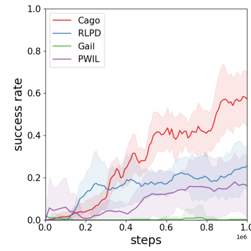
(a) Disassemble



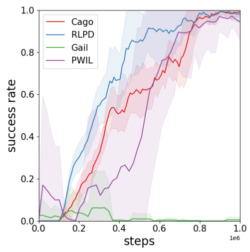
(b) PickPlaceWall



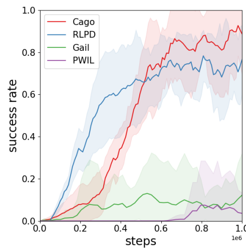
(c) ShelfPlace



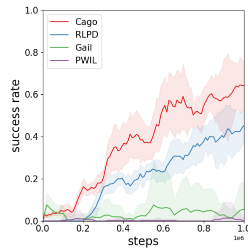
(d) StickPull



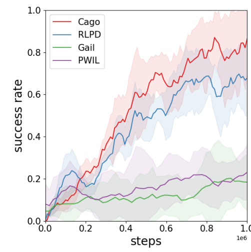
(e) StickPush



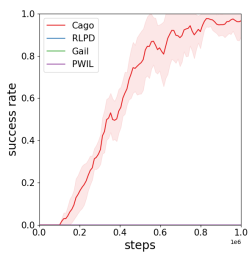
(f) Door



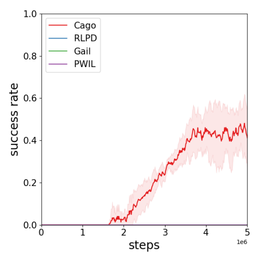
(g) Hammer



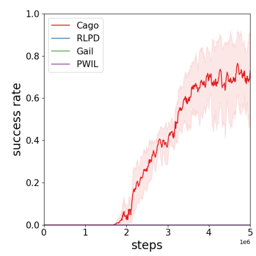
(h) Pen



(i) PullCubeTool

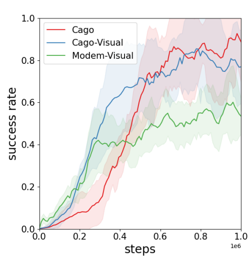


(j) PegInsertionSide

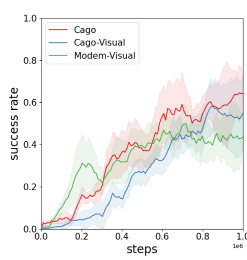


(k) StackCube

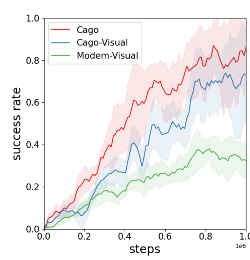
Figure 8: Experiment results comparing Cago with the Gail, PWIL and RLPD over 5 random seeds.



(a) Door



(b) Hammer



(c) Pen

Figure 9: Visual input experiment results over 5 random seeds.

purely from a single initial image observation, even in unseen evaluation seeds. This predictive capability allows the goal-conditioned policy π^G to finish tasks effectively for any environment seeds.

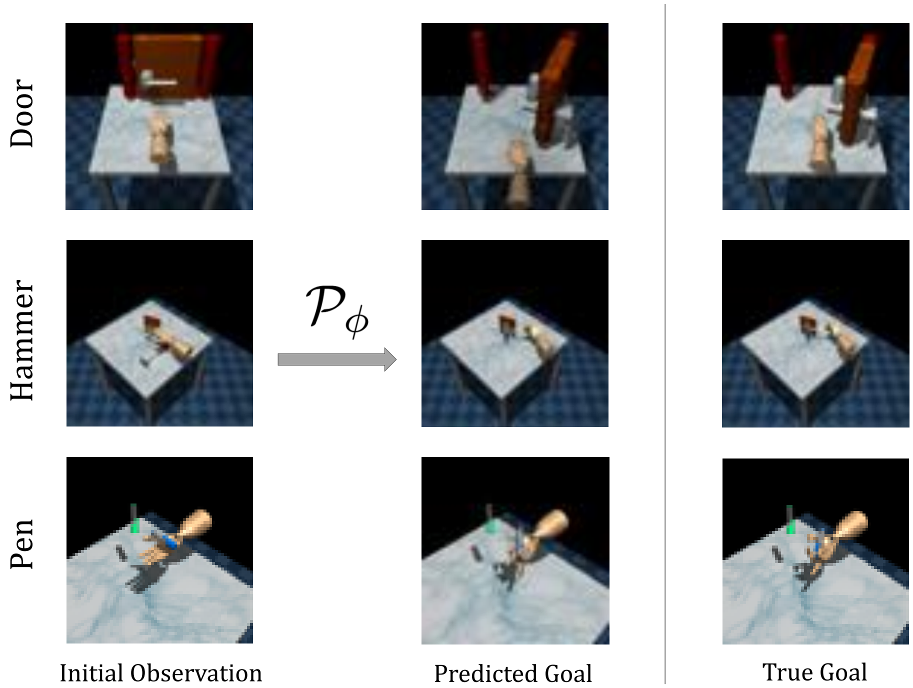


Figure 10: Visual goal prediction results from our learned goal predictor $\mathcal{P}_\phi$. Each row corresponds to a different task (Door, Hammer, Pen). From left to right: the agent's initial observation at test time, the goal image predicted by $\mathcal{P}_\phi$, and the ground-truth final observation from a demonstration trajectory. Notably, these demonstrations are drawn from unseen seeds not used during training. The predicted goals closely match the actual final states, illustrating strong generalization of $\mathcal{P}_\phi$ to novel environment seeds.

G Runtime

G.1 Experiment total runtimes

Table 1: Runtimes per experiment.

Environment	Runtime (h)	Benchmark	Steps
ShelfPlace	72	MetaWorld	1e6
Disassemble	72	MetaWorld	1e6
StickPull	72	MetaWorld	1e6
StickPush	72	MetaWorld	1e6
PickPlaceWall	72	MetaWorld	1e6
Door	80	Adroit	1e6
Hammer	85	Adroit	1e6
Pen	83	Adroit	1e6
PullCubeTool	78	ManiSkill	1e6
PegInsertionSide	143	ManiSkill	5e6
StackCube	155	ManiSkill	5e6

G.2 Computation Time for Updating the Cago Visitation Record Dictionary

In this section, we analyze the computational cost associated with updating the visitation record dictionary. Let the length of a sampled trajectory be denoted as L_τ, and let $\tau^{(i)}$ represent the demonstration trajectory associated with the same environment reset seed, having length L_i. The visitation record dictionary $\text{Dict}_{\text{visit}}$ is updated according to Equation 2:

$$\text{Dict}_{\text{visit}}[\tau^{(i)}][j] \mathrel{+}= 1 \quad \text{if } \text{sim}(s_t, s_j^{(i)}) \leq \epsilon, \quad \forall t \in 1, \ldots, L_\tau, \forall j \in 1, \ldots, L_i$$

This update rule implies that for each step in the sampled trajectory, a similarity check is performed against all steps in the corresponding demonstration trajectory. Thus, the time required to perform an update of $\text{Dict}_{\text{visit}}$ can be approximated by:

$$\text{Time(Update)} \approx \text{Total Steps} \times L_i \times \text{Time(Similarity Calculation)}$$

The computational cost is therefore influenced by three main factors: the total number of interaction steps, the length of each demonstration trajectory, and the cost of computing the similarity metric. Importantly, the similarity function $\text{sim}(\cdot, \cdot)$ differs by environment, which directly affects computation time. For MetaWorld environments, we utilize L2-distance in the state vector space (i.e., low-dimensional numerical vectors). This calculation is computationally efficient, typically requiring only simple element-wise operations over vector entries. In contrast, for the Adroit and Maniskill environments, the similarity is computed based on MSE in the image space. This involves pixel-wise comparison over image observations, which increases the computational load due to the large input dimensionality (e.g., $100 \times 100 \times 3$).As a result, while the update rule remains structurally the same, the actual runtime overhead for image-based similarity can be substantially higher than that for state-based similarity. The table below summarizes the total runtime, update time, and similarity function used for each environment:

Table 2: Computation time and similarity function for updating the visitation dictionary $\text{Dict}_{\text{visit}}$.

Environment	Steps	Runtime (h)	Update Time (h)	Similarity
ShelfPlace	1e6	72	0.05	State L2
Disassemble	1e6	72	0.05	State L2
StickPull	1e6	72	0.05	State L2
StickPush	1e6	72	0.05	State L2
PickPlaceWall	1e6	72	0.05	State L2
Door	1e6	80	5.3	Image MSE
Hammer	1e6	85	5.3	Image MSE
Pen	1e6	83	5.3	Image MSE
PullCubeTool	1e6	78	2.8	Image MSE
PegInsertionSide	5e6	143	13.8	Image MSE
StackCube	5e6	155	14.2	Image MSE

H Hyperparameters

We adopt the default hyperparameters from the LEXA backbone model-based RL (MBRL) agent—such as the learning rate, optimizer, and network architecture—and maintain them consistently across all environments. The primary hyperparameter tuning for Cago focuses on the following aspects: (1) the episode length L_τ; (2) the proportion of L_τ allocated to the goal-directed phase T_{go}; (3) the number of demonstrations N_{demo} used for both dictionary construction and environment resetting; (4) the visit frequency threshold λ_{visit} used in Algorithm 1 for filtering goal candidates; and (5) the similarity calculate metrics in Equation 2; (6) the similarity threshold ϵ in Equation 2.

Table 3: Hyperparameters of Cago.

Environment	L_τ	T_{go} $rate$	N_{demo}	λ_{visit}	sim$(\cdot,\cdot)$	ϵ
MetaWorld all environments	150	0.7	10	200	State L2	0.05
Adroit-Door	200	0.7	10	100	Image MSE	100.0
Adroit-Hammer	500	0.7	10	200	Image MSE	100.0
Adroit-Pen	200	0.7	10	100	Image MSE	200.0
PullCubeTool	100	0.7	10	100	Image MSE	100.0
PegInsertionSide	100	0.7	20	200	Image MSE	100.0
StackCube	100	0.7	20	200	Image MSE	100.0

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes] , [No] , or [NA] .
- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for NA).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will be asked to also include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While "[Yes] " is generally preferable to "[No] ", it is perfectly acceptable to answer "[No] " provided a proper justification is given (e.g., "error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering "[No] " or "[NA] " is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

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

   Answer: [Yes]

   Justification: Our abstract and introduction accurately represent the primary contributions of the paper, which include the development of the Cago algorithm designed to solve sparse-reward tasks through a capability-aware goal picking strategy. The introduction outlines the key challenges in GCRL, specifically with sparse rewards, and how Cago addresses these by improving the quality of online sampled trajectories using demonstration. These claims are well-supported by the theoretical underpinnings and experimental results presented in the paper, reflecting the scope and impact of the proposed method.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: Our paper thoroughly discusses the limitations of the Cago framework in Appendix. We highlight the reliance on high-quality demonstrations. We also outline future directions to address these issues, such as learning from suboptimal demonstrations and incorporating uncertainty-aware models.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

   Answer: [Yes]

   Justification: Yes, our paper provides the full set of assumptions and a complete (and correct) proof in the Appendix A for Theorem 1.

   Guidelines:

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

4. **Experimental result reproducibility**

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

Answer: [Yes]

Justification: In the Experiment section and appendix of our paper, we elaborate on the procedure and configuration of our experiments. This includes the sources and modifications of all testing environments, and implementation methods for all baselines, the devices and memory utilized, as well as specific values of hyperparameters employed. Concurrently, we have open-sourced our code; please refer to the Reproducibility Statement section for further details.

Guidelines:

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

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

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

   Answer: [Yes]

   Justification: As we mentioned in the previous justification, we have not only open-sourced our code but also provided detailed steps and settings for reproducing our main experimental results. In the Experiment section and Appendix, we elaborate on the sources and modifications of the environments, baseline implementation details, and Cago implementation specifics.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: We provide comprehensive details regarding the hyperparameters essential for understanding the experiments, including those specific to our Cago framework. The table presented (Table. 3) outlines these hyperparameters for each task, facilitating reproducibility and comparison.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [Yes]

   Justification: We conducted each experiment a minimum of five times using different random seeds, and upon plotting the results, as demonstrated in the Experiment section, we incorporated the experimental error. The solid line denotes the average success rate, while the shaded region signifies the standard deviation among the repeated experimental outcomes.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: We clearly specifies the computer resources (8 Nvidia A100 GPU) and the amount of GPU memory required (approximately 2.4GB).

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: The research conducted in our paper aligns with the NeurIPS Code of Ethics. We have thoroughly reviewed the guidelines and ensured that our research adheres to ethical standards. Additionally, we have implemented measures to safeguard anonymity and comply with pertinent laws and regulations.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

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

    Answer: [NA]

    Justification: Our research aims to address the learning efficiency problem in Reinforcement Learning (RL) within the GCRL environment. It is currently in the theoretical research stage and has minimal societal impact.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

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

    Answer: [NA]

    Justification: Our paper poses no such risks.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: Our paper properly credits the creators or original owners of assets used, including code, data, and models.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

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

    Answer: [Yes]

    Justification: We have documented our code and provided detailed instructions on its usage, licenses, and permissible scope of use. Additionally, we have included the documentation alongside the assets to ensure accessibility and clarity for users.

    Guidelines:

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

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

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

    Answer: [NA]

    Justification: Our paper not involve crowdsourcing nor research with human subjects.

    Guidelines:

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

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

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

    Answer: [NA]

    Justification: Our paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

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

    Answer: [NA]

    Justification: Our method does not involve LLMs as any important, original, or non-standard components.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.