HINDSIGHT-DICE: STABLE CREDIT ASSIGNMENT FOR DEEP REINFORCEMENT LEARNING

Akash Velu[*]   **Skanda Vaidyanath**[*]   **Dilip Arumugam**
Department of Computer Science
Stanford University
{avelu,svaidyan,dilip}@stanford.edu

ABSTRACT

Oftentimes, environments for sequential decision-making problems can be quite sparse in the provision of evaluative feedback to guide reinforcement-learning agents. In the extreme case, long trajectories of behavior are merely punctuated with a single terminal feedback signal, leading to a significant temporal delay between the observation of a non-trivial reward and the individual steps of behavior culpable for achieving said reward. Coping with such a credit assignment challenge is one of the hallmark characteristics of reinforcement learning. While prior work has introduced the concept of hindsight policies [21] to develop a theoretically motivated method for reweighting on-policy data by impact on achieving the observed trajectory return, we show that these methods experience instabilities which lead to inefficient learning in complex environments. In this work, we adapt existing importance-sampling ratio estimation techniques for off-policy evaluation to drastically improve the stability and efficiency of these so-called hindsight policy methods. Our hindsight distribution correction facilitates stable, efficient learning across a broad range of environments where credit assignment plagues baseline methods.

1 Introduction

Reinforcement learning is the classic paradigm for addressing sequential decision-making problems [47]. Naturally, while inheriting the fundamental challenge of generalization across novel states and actions from supervised learning, general-purpose reinforcement-learning agents must also contend with the additional challenges of exploration and credit assignment. While much initial progress in the field was driven largely by innovative machinery for tackling credit assignment [45, 46, 44] alongside simple exploration heuristics (ε-greedy exploration, for example), recent years have seen a reversal with the bulk of attention focused on a broad array of exploration methods (spanning additional heuristics as well as more principled approaches) [51, 38, 15], and relatively little consideration given to issues of credit assignment. This lack of interest in solution concepts, however, has not stopped the proliferation of reinforcement learning into novel application areas characterized by long problem horizons and sparse reward signals; indeed, the current reinforcement learning from human feedback (RLHF) paradigm [28] is now a widely popularized example of an environment that operates in perhaps the harshest setting where a single feedback signal is only obtained after the completion of a long trajectory. [1]

This combination of long horizons with delayed rewards most directly impacts an agent's ability to efficiently perform credit assignment [33, 45, 46] and thereby worsens the overall sample complexity needed to synthesize an optimal policy. Therefore, while a reinforcement-learning agent must also contend with the challenges of generalization and exploration, it is perhaps the credit assignment challenge that poses the greatest impediment to sample-efficient reinforcement learning. While theoretical results encapsulating settings like that of the RLHF paradigm are nascent [11, 37], practical and demonstrably-scalable mechanisms for efficient credit assignment are missing from the literature.

With this reality in mind, our work extends the relatively recent solution concept of hindsight policies as introduced by Harutyunyan et al. [21], which allow an agent to reason counterfactually about past decisions at particular states in

---

[*]Equal contribution.

[1]Our implementation is available at https://github.com/skandavaidyanath/credit-assignment.

light of observed future outcomes. While the natural course of action for algorithm design based on the provision of such a hindsight policy is to reweight the on-policy data collected for agent updates, we highlight a key issue where the resulting importance-sampling ratio is highly unstable, compromising the quality of learning. Even when ratio clipping is applied, as one often does in off-policy policy-gradient methods [42], such a heuristic lends stability at the heavy cost of incredibly dampened learning speed.

To remedy this instability, we turn to prior work tackling the orthogonal problem of off-policy policy evaluation (OPE) [39, 24, 52] wherein the ratio between state visitation probabilities of a behavior policy and target policy has emerged over recent years as a key quantity of interest for yielding principled, practical OPE algorithms [19, 29, 17, 30, 31, 36, 54, 58, 59, 14] whose variance does not suffer exponentially in the problem horizon, as it would with a naive application of importance sampling. Inspired by the potential for technical progress in this separate arena to be recycled and operationalized towards achieving more efficient credit assignment techniques, we select one such approach from the OPE literature, Dual stationary DIstribution Correction Estimation (DualDICE) [36], and adapt the importance ratio approximation technique for hindsight policies. Our resulting **H**indsight **DI**stribution **C**orrection **E**stimation (**H-DICE**) approach empowers policy-gradient methods [42] to empirically avoid the same catastrophic instability that arises by naively computing the importance ratio between the current and hindsight policies. We empirically demonstrate over three different and challenging benchmarks tasks that our method outperforms baseline methods, which either do not incorporate any form of credit assignment or do so in a naive fashion, by a large margin both in terms of the achieved final rewards and the speed of convergence.

The paper proceeds as follows: in Section 2 we detail our problem formulation as well as background information on credit assignment with hindsight policies. In Section 3, we outline our approach before presenting a detailed empirical evaluation and discussion of results in Section 4. We conclude with an overview of related work in Section 5 and future work in Section 6.

2 Preliminaries & Background

In this section, we begin by specifying our problem formulation before providing brief background information on the credit assignment problem broadly as well as the specific use of hindsight policies to address the challenge.

For any natural number $N \in \mathbb{N}$, we denote the index set as $[N] \triangleq \{1, 2, \ldots, N\}$. For any set $\mathcal{X}$, we denote the set of all probability distributions with support on $\mathcal{X}$ as $\Delta(\mathcal{X})$. For another arbitrary set $\mathcal{Y}$, we denote the class of all functions mapping from $\mathcal{X}$ to $\mathcal{Y}$ as $\{\mathcal{X} \to \mathcal{Y}\}$.

2.1 Problem Formulation

We formulate a sequential decision-making problem as an infinite-horizon, discounted Markov Decision Process (MDP) [8, 40] defined by $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{T}, \mu, \gamma \rangle$. Here $\mathcal{S}$ denotes a set of states, $\mathcal{A}$ is a set of actions, $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is a deterministic reward function providing evaluative feedback signals to the agent, $\mathcal{T} : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ is a transition function prescribing distributions over next states, $\mu \in \Delta(\mathcal{S})$ is an initial state distribution, and $\gamma \in [0, 1)$ is the discount factor communicating a preference for near-term versus long-term rewards. Beginning with an initial state $s_0 \sim \mu$, for each timestep $t \in \mathbb{N}$, the agent observes the current state $s_t \in \mathcal{S}$, selects action $a_t \sim \pi(\cdot \mid s_t) \in \mathcal{A}$, enjoys a reward $r_t = \mathcal{R}(s_t, a_t)$, and transitions to the next state $s_{t+1} \sim \mathcal{T}(\cdot \mid s_t, a_t) \in \mathcal{S}$.

Our reinforcement-learning approach is grounded in policy search [56, 48, 27, 35, 42] wherein the agent's behavior is represented by a stationary, stochastic policy $\pi_\theta : \mathcal{S} \to \Delta(\mathcal{A})$, parameterized by $\theta \in \Theta \subset \mathbb{R}^d$, which encodes a pattern of behavior mapping individual states to distributions over possible actions. The overall performance of π_θ in any MDP $\mathcal{M}$ when starting at state $s \in \mathcal{S}$ and taking action $a \in \mathcal{A}$ is assessed by its associated action-value function $Q^{\pi_\theta}(s, a) = \mathbb{E} \left[\sum_{t=0}^{\infty} \gamma^t \mathcal{R}(s_t, a_t) \mid s_0 = s, a_0 = a \right]$, where the expectation integrates over randomness in the action selections and transition dynamics. Taking the corresponding value function as $V^{\pi_\theta}(s) = \mathbb{E}_{a \sim \pi_\theta(\cdot \mid s)} [Q^{\pi_\theta}(s, a)]$ and letting $\Pi_\Theta \triangleq \{\pi_\theta \mid \theta \in \Theta\} \subset \{\mathcal{S} \to \Delta(\mathcal{A})\}$ denote the policy class parameterized by Θ (a subset of all stochastic policies), we define the optimal policy $\pi^\star$ as achieving supremal value with respect to this class $V^\star(s) = \sup_{\pi \in \Pi_\Theta} V^\pi(s) = \max_{a^\star \in \mathcal{A}} Q^\star(s, a^\star)$ and $Q^\star(s, a) = \sup_{\pi \in \Pi_\Theta} Q^\pi(s, a)$ for all $s \in \mathcal{S}$ and $a \in \mathcal{A}$. As the agent interacts with the environment over the course of $K \in \mathbb{N}$ episodes, we let $\tau_k = (s_1^{(k)}, a_1^{(k)}, r_1^{(k)}, \ldots) \sim \rho^{\pi_{\theta_k}}$ be the random variable denoting the trajectory experienced by the agent in the kth episode sampled from the induced trajectory distribution $\rho^{\pi_{\theta_k}}$ of policy π_{θ_k}, for any $k \in [K]$. Alternatively, one may also examine the visitation of a policy π_θ through its discounted, stationary
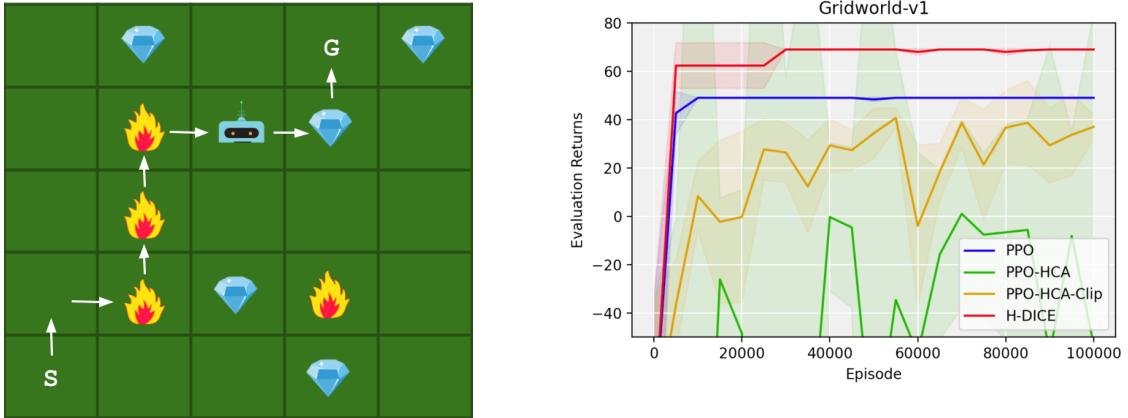
Figure 1: *Left:* GridWorld-v1 environment, in which the agent starts at square S and must collect diamonds to obtain positive rewards, and avoid fire which result in negative rewards. The reward signal is delayed so that the agent receives the sum of all rewards upon episode termination, which occurs either when the agent transitions into the G square or the episode times out. Solid arrows denote an example trajectory in which the agent steps into fire three times and collects one diamond before reaching the G square. The robot indicates the position of the agent at a point in this trajectory, referenced in Section 4.5. *Right:* Learning curves of our proposed method, H-DICE, and baseline methods (PPO, PPO-HCA, and PPO-HCA-Clip) on Gridworld-v1. H-DICE achieves the maximum reward quickly and stably compared to other methods. Vanilla PPO is unable to disentangle the good and bad actions even in this simple environment since it does not explicitly model credit assignment and this leads to a suboptimal policy. While using unclipped HCA ratios to compute advantages (PPO-HCA) results in highly unstable performance, clipping the ratios (PPO-HCA-Clip) improves results slightly.

state distribution: $d^{\pi_\theta}(s) = (1 - \gamma) \sum_{t=0}^{\infty} \gamma^t \mathbb{P}^\pi(s_t = s)$, where $\mathbb{P}^\pi(s_t = \cdot) \in \Delta(\mathcal{S})$ denotes the distribution over states visited by π_θ at each timestep t. Finally, in alignment with prior work [7, 21], we use $Z(\tau)$ to denote the random variable representing the return of a trajectory τ.

2.2 A Motivating Example for Credit Assignment

Consider the gridworld setting depicted in Figure 1, wherein an agent accrues positive rewards for collecting diamonds and negative rewards for moving through fire. Importantly, however, the agent only receives a nonzero reward signal – which is the sum of the rewards accrued in each individual transition – at the end of a trajectory when it enters the G square and no receives rewards for the intermediary steps.

In the example trajectory illustrated in Figure 1, the agent transitions through three fire squares before acquiring a diamond and reaching G, upon which it receives a large negative reward (due to the three steps into the fire). A typical policy gradient method would place "blame" for this poor outcome on *each* action in the trajectory and would correspondingly reduce the probability that these actions are taken by the policy. In fact, due to a discount factor $\gamma < 1$, the algorithm would assign greater blame to actions which are temporally closer to the final return, resulting in a larger decrease in the probability of the action which picked up the diamond compared to the actions which put the agent in fire; this, however, is undesirable. Despite the final negative return, picking up the diamond is the optimal action in the corresponding square. Ideally, this action should be made more likely, while the actions moving the agent into fire should be reduced in likelihood, as they are culpable for the negative return.

Credit assignment aims to perform exactly this type of reasoning; not all actions are equally responsible for a future outcome, and determining which actions are responsible in yielding a particular end result – which is the goal of credit assignment – can enable more efficient policy learning and improved policy performance.

2.3 Hindsight Credit Assignment

More generally, the credit assignment problem revolves around the fixed behavior policy of an agent and asks *what is the impact of being in a particular state and executing a particular action on observed future outcomes?* [21, 5]. Typically, the future outcome of interest is the full return $Z(\tau)$ associated with a complete trajectory τ, although one

could also opt to examine intermediate returns observed from any incident states or state-action pairs. Unfortunately, various aspects of the reinforcement-learning setting exacerbate the credit assignment problem and make it exceedingly difficult to accurately answer this fundamental question in a straightforward manner. Long horizons enable feedback structures where nontrivial signals can emerge after a significant delay from the culpable step of behavior. Environments whose underlying transition structure is highly stochastic induce large variance in the resulting trajectories sampled from any policy, making it difficult to obtain accurate estimates of the overall return needed to assign credit or blame. In the context of policy search, initial policies whose actions carry little information about the incident trajectory returns can make initial thrust toward efficiently allocating credit to observed experiences challenging [5]. Moreover, credit assignment must often be adjudicated in the presence of a single sampled trajectory but, ideally, should factor in this stochasticity to give appropriate credit to those steps of behavior that could have occurred under a different random rollout [55].

While classic approaches to addressing credit assignment would naively distribute credit based on temporal recency to interesting events or outcomes [26, 45, 44], a more recent and cogent mechanism has emerged through the use of a *hindsight policy* [21]. For the purposes of this work, a hindsight policy $h_\omega^{\pi_\theta} : \mathcal{S} \times \mathbb{R} \to \Delta(\mathcal{A})$ induced by a behavior policy π_θ and parameterized by $\omega \in \Omega \subset \mathbb{R}^m$ is a return-conditioned policy that yields the distribution over actions taken by policy π_θ from a given state and in light of an observed trajectory return. Inspired by the hindsight experience replay technique [3] for goal-based reinforcement learning, a hindsight policy allows an agent to consider the full space of possible actions that could have occurred in hindsight based on the final trajectory return. This effect becomes apparent from the resulting identity for the advantage function [6] computed by policy-gradient methods [35, 42, 43]:

$$A^{\pi_\theta}(s,a) = Q^{\pi_\theta}(s,a) - V^{\pi_\theta}(s) = \mathbb{E}\left[\left(1 - \frac{\pi_\theta(a \mid s)}{h_\omega^{\pi_\theta}(a \mid s, Z(\tau))}\right) Z(\tau)\right]. \tag{1}$$

If, in hindsight, observing a return $Z(\tau)$ makes a particular action less likely, then the *hindsight ratio* $\frac{\pi_\theta(a|s)}{h_\omega^{\pi_\theta}(a|s,Z(\tau))} > 1$ and detracts from the advantage of the action; in contrast, an action that makes the return $Z(\tau)$ more plausible in hindsight results in an increased advantage signal as $\frac{\pi_\theta(a|s)}{h_\omega^{\pi_\theta}(a|s,Z(\tau))} < 1$. In the next section, we discuss the practical challenges that emerge when trying to estimate this hindsight ratio so as to integrate this credit assignment information within a policy-gradient method.

3 Approach

We begin by examining the natural, straightforward approach to hindsight ratio estimation and highlight stability issues that arise from using such an estimator. We then proceed to introduce one alternative avenue for hindsight ratio estimation inspired by prior work in OPE. Finally, we conclude by introducing our H-DICE approach, which leverages this ratio estimator for demonstrably stable credit assignment.

3.1 Direct Density Ratio Instability

As detailed in Section 2.3, the HCA framework derives an expression for advantage function $A^{\pi_\theta}(s,a)$ of a policy π_θ, which, through the return-conditioned hindsight function h^{π_θ}, explicitly assigns the credit each action receives for achieving a particular return from the current state. Given on-policy trajectory data $\mathcal{D}^{\pi_\theta}$, a straightforward way to compute this advantage would be to first fit a hindsight function $h_\omega^{\pi_\theta}$ to the data via supervised learning (where ω are trainable parameters). The hindsight ratio is then calculated via $\frac{\pi_\theta(a|s)}{h_\omega^{\pi_\theta}(a|s,Z(\tau))}$ for each state-action pair (s,a) in a trajectory τ, and $A^\pi(s,a)$ can subsequently be computed using expression 1, as done in the approach of Harutyunyan et al. [21].

However, direct estimation of density ratios can be highly unstable and, as shown in the right of Figure 1 and Section 4.5, the hindsight ratio is no exception. Using Proximal Policy Optimization (PPO) [42] as a base policy-gradient algorithm, we empirically observe that values the ratio takes through the course of training vary greatly, hindering policy learning. PPO-HCA, a method which directly estimates the hindsight ratio, performs poorly and exhibits high variance across seeds. Clipping the ratios to be between zero and one, a common way to stabilize density ratios [42], improves the stability of learning but still results in suboptimal performance, as seen by the results of PPO-HCA-Clip. Thus, the question remains: how can we compute the hindsight ratio in a *stable* manner?

3.2 Distribution Correction Estimation

In this work, we offer one candidate answer to the above question by taking inspiration from the off-policy policy evaluation (OPE) literature, where recent methods have been developed to accurately and soundly compute the

importance sampling ratio between two state-action distributions. Specifically, we leverage the following observation from [36] which serves as a core building block for their OPE algorithm: defining $\mathcal{C} \triangleq [0, C]$ for a bounded constant $C \in \mathbb{R}_{\geq 0}$ and a function $x : \mathcal{S} \times \mathcal{A} \to \mathcal{C}$, the solution $x^*(s, a)$ of the following optimization problem

$$\min_{x:S \times A \to \mathcal{C}} \frac{1}{2} \mathbb{E}_{(s,a) \sim d^{\mathcal{D}}} [x(s,a)^2] - \mathbb{E}_{(s,a) \sim d^{\pi}} [x(s,a)] \tag{2}$$

is given by $x^*(s, a) = \frac{d^{\pi}(s,a)}{d^{\mathcal{D}}(s,a)}$. In the context of their work, $d^{\pi}(s, a)$ is the discounted state-action visitation distribution of a target policy π of interest while $d^{\mathcal{D}}(s, a)$ is free to be an arbitrary empirical distribution over state-action pairs representing a static, offline dataset $\mathcal{D}$ (which potentially emerges from sampling a single behavior policy, multiple policies, or some other sampling oracle). In the OPE setting, the second term of Equation 2 is problematic as collecting on-policy rollouts of π is not possible. Consequently, the DualDICE method [36] goes on to extend on this optimization and remedy the issue. In contrast, the on-policy nature of the credit assignment problem, whose core question is always asked with respect to a current behavior policy of interest, makes Equation 2 a sufficient and viable approach for hindsight ratio estimation. We leave to future work the question of how alternative techniques for density-ratio estimation [13] might be applied towards achieving stability and improved agent performance in the context of HCA.

3.3 Hindsight Distribution Correction Estimation

Our H-DICE method for stable computation of the hindsight ratio $\frac{\pi_\theta(a|s)}{h_\omega^{\pi_\theta}(a|s, Z(\tau))}$ adapts the observation from the previous section to our setting by instead focusing on the following optimization:

$$\min_{\phi:S \times A \times Z \to \mathcal{C}} \frac{1}{2} \mathbb{E}_{(s,a,z) \sim \mathcal{D}^{h^{\pi_\theta}}} [\phi(s,a,z)^2] - \mathbb{E}_{\substack{(s,a) \sim d^{\pi_\theta} \\ z \sim \psi(z)}} [\phi(s,a,z)] \tag{3}$$

where $\mathcal{D}^{h^{\pi_\theta}}$ denotes an empirical state-action-return distribution formed by drawing states under the d^{π_θ} visitation of π_θ, returns of π_θ observed from state s according to $\chi^{\pi_\theta}(\cdot|s)$, and actions from the hindsight policy $h_\omega^{\pi_\theta}(\cdot \mid s, z)$. Meanwhile, $\psi(z)$ is an arbitrary distribution over the returns that, in our experiments, we model as a uniform distribution over all possible returns in the environment, which results in simplified hindsight ratio calculation. In the main results presented in Section 4, we set the range threshold $C \in \mathbb{R}_{\geq 0}$ for ϕ to be 1; however, as shown in Appendix B, the choice of C is largely inconsequential and numerous values result in similar performance.

While Appendix A elaborates in more detail, we briefly mention here that the solution to this optimization, $\phi^\star$, is given by:

$$\phi^\star(s, a, z) = \frac{\pi_\theta(a|s)}{\chi^{\pi_\theta}(z|s) h_\omega^{\pi_\theta}(a|s, z)}.$$

This allows us to arrive at the following expression for the hindsight ratio of interest:

$$\frac{\pi_\theta(a|s)}{h_\omega^{\pi_\theta}(a|s, z)} = \phi^\star(s, a, z) \chi^{\pi_\theta}(z|s) \tag{4}$$

Importantly, the right hand side of Equation 4 can be estimated from on-policy data $\mathcal{D}^{\pi_\theta}$ collected under π_θ. This involves learning the following three models, which we instantiate as neural networks:

1. **Return predictor** $\chi_\eta^{\pi_\theta}$: We learn the distribution over returns incurred by policy π_θ given a state $s \in \mathcal{S}$ via supervised learning using $\mathcal{D}^{\pi_\theta}$. Note that our approach can be seen as a Monte-Carlo estimation of the return distribution and $\chi_\eta^{\pi_\theta}$ could also be learned via distributional reinforcement learning [7], an avenue we leave to future work.

2. **Hindsight policy** $h_\omega^{\pi_\theta}$: Just as in Harutyunyan et al. [21], we learn a hindsight policy via supervised learning using on-policy data $\mathcal{D}^{\pi_\theta}$.

3. **Hindsight DICE model** ϕ_ν: Lastly, our estimate of $\phi^\star$ in Equation 4 is given by ϕ_ν optimized via Equation 3. Naturally, this requires $\mathcal{D}^{\pi_\theta}$ along with $\chi_\eta^{\pi_\theta}$ and $h_\omega^{\pi_\theta}$.

As all the aforementioned models used to compute hindsight ratios are (either implicitly or explicitly) conditioned on the current policy, π_θ, we must reset the weights of the corresponding function approximators after each policy update and before collecting fresh data from the latest policy. In our experiments, since these models are implemented as lightweight, fully-connected networks with few layers, we find that this does not adversely affect wall-clock time or the computational efficiency of our algorithm.

Algorithm 1 Hindsight Distribution Correction Estimation

Initialize π_{θ_1}
for iteration $i = 1 \ldots N$ **do**
    Initialize $h_\omega^{\pi_{\theta_i}}$, ϕ_ν, and $\chi_\eta^{\pi_{\theta_i}}$
    Collect dataset $\mathcal{D}^{\pi_{\theta_i}}$ of on-policy trajectories with π_{θ_i}
    Train $\chi_\eta^{\pi_{\theta_i}}$ and $h_\omega^{\pi_{\theta_i}}$ by supervised learning with $\mathcal{D}^{\pi_{\theta_i}}$
    Train H-DICE ϕ_ν by Equation 3
    Obtain θ_{i+1} via PPO update on π_{θ_i} with advantages in Equation 1
end for

Like DualDICE, our approach for estimating the hindsight ratio inherits an avoidance of explicitly computing importance-sampling weights, thereby alleviating the instability that arises from direct density ratio estimation observed in Section 3.1. The full algorithm for policy learning using H-DICE is detailed in Algorithm 1. Note that although H-DICE and all baseline methods use PPO [42] as the base policy learning algorithm, our approach is agnostic to this choice and compatible with any policy-gradient algorithm. In contrast to many standard PPO implementations which learn a value-function alongside a policy to compute advantages, H-DICE does not learn a value-function to compute advantages as a consequence of Equation 1.

4 Experiments

In this section, we evaluate H-DICE on complex discrete and continuous control tasks and compare it with baselines. We also closely investigate the learned hindsight ratios and explore varying update schedules for the hindsight, return and DICE models (referred to jointly as "auxiliary models"). For the sake of brevity, we defer additional experiments on the impact of the constant C, using different distributions for ψ, and using the H-DICE advantage estimator on dense reward settings to Appendix Sections B, C and D respectively. We begin by providing the details of our evaluation domains and chosen baselines.

4.1 Domains

We evaluate methods on the following domains:

GridWorld: The GridWorld environments, shown in Figures 1 and 8, are environments we develop which pose a difficult credit-assignment challenge. In each square, the agent can take actions representing the cardinal directions. Each square in the grid may contain a diamond or a fire; transitioning into a diamond square gives the agent a reward of +20, while running into a fire square gives the agent a reward of -100. Moreover, each step the agent takes incurs a reward of -1. An episode ends either when the agent transitions into a specific goal square, or when the episode length passes a predetermined constant threshold. We create two such grid world settings: GridWorld-v1, a smaller grid with a maximum episode length of 50, and GridWorld-v2, a larger grid with a maximum episode length of 100. Intuitively, an optimal agent will accumulate diamonds as quickly as possible while avoiding fire, and eventually navigate to the goal square to finish the episode.

The credit-assignment challenge in this domain arises from the fact that all rewards are delayed until the very end of an episode; in other words, the sum of the rewards received for each individual transition is given as a single cumulative reward at the end of the episode, and all intermediate rewards given to the agent are zero. We note that delaying the reward in this manner violates the Markov property of the MDP reward function and, technically, would be better modeled as a partially-observable MDP (POMDP) [25] where rewards are a function of a latent underlying state not observable to the agent. However, the optimal policy for this particular POMDP (and future delayed-reward settings we will introduce) can still be learned purely as a function of the partial observation, obviating the need for considering the underlying latent state or redefining states around the full trajectory. We also note that this approach of delaying rewards in an MDP while keeping the rest of the structure the same has been studied extensively in prior work [4, 23, 41, 16].

This delayed reward setting forces a good agent to disentangle the fact that collecting diamonds results in positive rewards while moving through fire results in bad rewards by assigning credit to individual actions within an episode. As we demonstrate in the following sections, this is a challenging task for algorithms which do not perform explicit credit assignment.

LunarLander: To evaluate methods on a more complex discrete action environment, we turn to the LunarLander-v2 benchmark from OpenAI Gym [9], in which an agent attempts to successfully land a spaceship within a specified
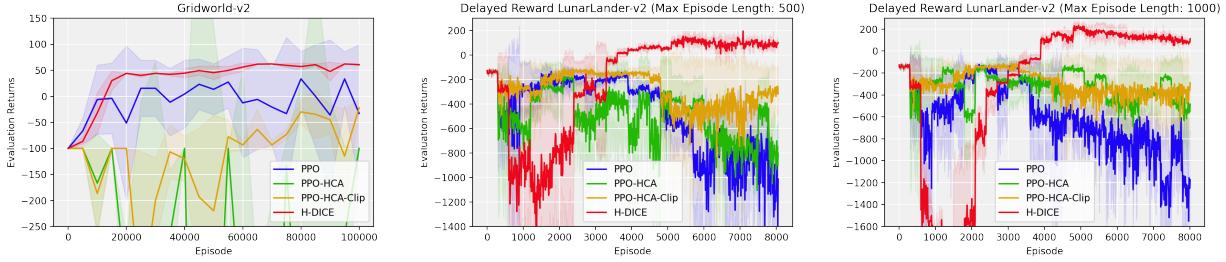
Figure 2: Training curves of H-DICE and baseline methods in Gridworld-v2 (left), and delayed reward LunarLander-v2 with maximum episode lengths of 500 (middle) and 1000 (right). The LunarLander-v2 environment is considered "solved" when returns surpass 200. All three environments utilize a discrete action space. H-DICE outperforms baseline methods in all three environments.

zone. Again, to impose a credit-assignment challenge in this setting, the environment is modified such that rewards are delayed to the end of the episode. We additionally evaluate the agent with maximum episode lengths of 500 and the default length of 1000. The environment is considered "solved" when the agent achieves a return of 200.

MuJoCo Gym Suite: Lastly, we benchmark methods on complex continuous control settings using four environments from the MuJoCo suite [53]: HalfCheetah, Humanoid, Walker2d, and Swimmer. Again, all rewards are delayed to episode termination and, additionally, episodes in all environments are truncated in length to a maximum of 100 steps; we additionally evaluate all methods in HalfCheetah with a maximum episode length of 50, as part of an ablation study.

4.2 Baselines

We compare H-DICE to three baseline approaches:

Vanilla PPO: Our first baseline is PPO with GAE-advantage calculation [42, 43], a method which does not perform explicit credit-assignment.

PPO-HCA: This baseline calculates the advantage as proposed by return-conditioned HCA [21] instead of the typical Generalized Advantage Estimate [43] used in PPO, but computes the hindsight ratio directly by simply dividing the policy and hindsight policy likelihoods. As shown in the results, this method of computing hindsight ratios is highly unstable and results in poor policy performance.

PPO-HCA-Clip: This baseline is identical to PPO-HCA, but attempts to stabilize the hindsight ratio (which is still computed by dividing the policy and hindsight policy ratios) by clipping the ratio to be between zero and one. We choose these bounds since we find smaller clip ranges improves performance and since the density ratios in H-DICE are between zero and one by virtue of the fact that, by default when $C = 1$, the hindsight DICE model and return predictor models output values between zero and one.

4.3 Experimental Setup

We tune H-DICE and each baseline method via a limited grid search over critical hyperparameters. Specifically, for each domain, we search over the entropy bonus coefficient and the amount of data collected before each policy update for each method. For HCA methods and H-DICE, we additionally tune the number of epochs for updating the hindsight model, return predictor, and DICE model. Lastly, for the vanilla PPO baseline, we also tune the coefficient on the value-function loss. We refer the reader to Appendix E for a detailed account of hyperparameters used. All results are averaged over three seeds; the solid lines in the plots correspond to the mean return and the shaded region to one standard deviation from the mean.

4.4 How does H-DICE perform compared to baselines on complex continuous control and discrete control tasks in the delayed reward setting

Figures 1, 2 and 3 display learning curves of H-DICE and baseline algorithms in a variety of different discrete and continuous control environments. The vertical axis denotes the returns achieved by the agent averaged over 10 evaluation episodes whereas the horizontal axis denotes the number of episodes elapsed since the beginning of training.
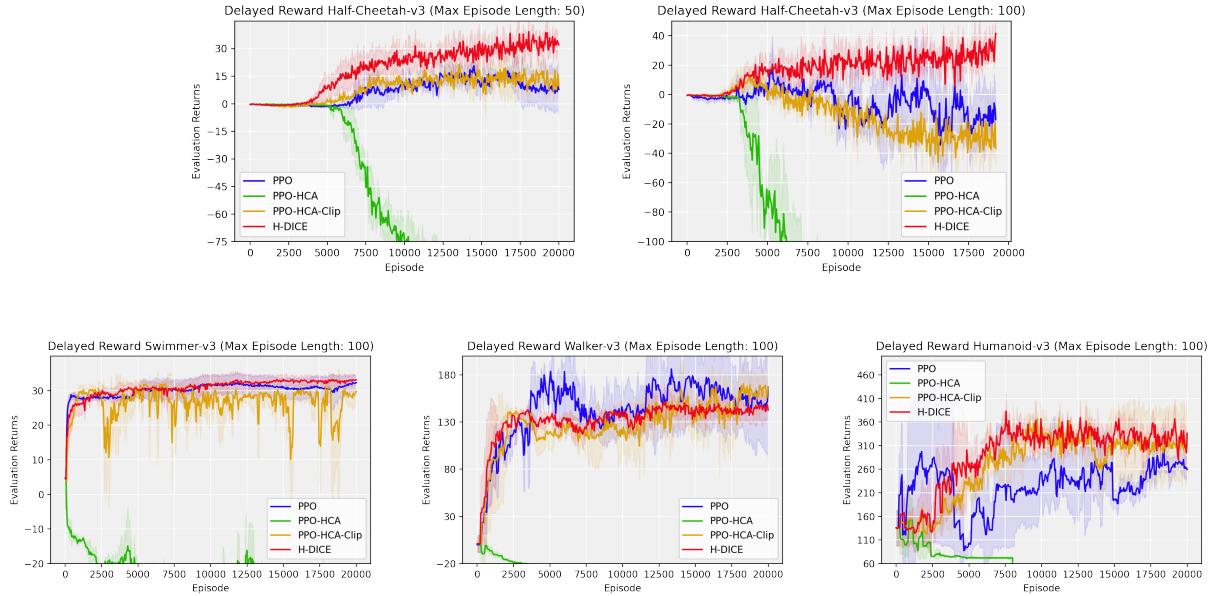
Figure 3: Training curves of H-DICE and baseline methods in Half-Cheetah with a maximum episode length of 50 and 100, and in other MuJoCo environments with a maximum episode length of 100. H-DICE outperforms baseline methods in the more difficult Half-Cheetah and Humanoid environments, and performs similarly to PPO and PPO-HCA-Clip on Swimmer and Walker.

H-DICE achieves higher final returns and converges faster than baseline methods in the vast majority of scenarios. PPO notably struggles to achieve high returns in many of these domains and is typically outperformed by H-DICE in final returns and learning efficiency. The exception to this is in the Swimmer and Walker environments, where PPO, PPO-HCA-Clip, and H-DICE perform similarly. This demonstrates the necessity for explicit credit-assignment in settings where the lack of immediate reward feedback requires reasoning about actions were responsible for generating the final return. As illustrated by the poor results of PPO-HCA across domains, computing hindsight ratios directly by dividing the policy and hindsight policy likelihoods results in instability issues and leads to suboptimal performance. The variance across seeds is extremely high, and the achieved returns oscillate drastically particularly in the GridWorld and LunarLander settings. Stabilizing the hindsight ratios via clipping, the strategy used by PPO-HCA-Clip, alleviates this instability in most settings, as evidenced by both the improvement in returns and the reduction in variance as compared to PPO-HCA. As it still explicitly models credit-assignment, PPO-HCA-Clip also outperforms PPO in several settings, but is clearly outperformed by H-DICE across most environments, demonstrating the need for more sophisticated methods for stable computation of the hindsight ratio.

We also note several interesting trends in the relationship between episode length and method performance. As illustrated by the final achieved rewards in the LunarLander and Half-Cheetah plots in Figures 2 and 3 respectively, the difference in performance between H-DICE and baseline methods (particularly PPO) is more pronounced when the maximum episode length is longer. This is likely explained by the fact that credit assignment becomes increasingly important over longer horizons, as the effects of earlier actions can manifest much later in the episode, and because the temporal distance between when actions are executed and when the agent receives a nonzero reward is greater.

4.5 Do the hindsight ratios learned by H-DICE encode meaningful credit assignment trends?

To answer this question, we take a fully trained H-DICE model on GridWorld-v1 and analyze the learned π_θ and h_{π_θ} values at a critical state in a trajectory – namely, when the agent is in the position shown in Figure 1. In this position, there are two interesting actions the agent can take: left or right, each of which leads to vastly different outcomes. Conditioned on the achieved final return, the agent must assign credit back to the action taken at this step to determine if it was culpable.

In the table below, we show values corresponding to the (hypothetical) action taken by the agent at this state (s), the (hypothetical) achieved final return (z), the probability of taking that action under policy π_θ at this state and the hindsight probability, $h^{\pi_\theta}(a \mid s, z)$, of choosing this action at this state given the final return and the current policy.

From these values, we also show the naive ratio calculation of $\frac{\pi_\theta(a|s)}{h_\omega^{\pi_\theta}(a|s,z)}$ and the same ratio obtained via H-DICE using Equation 4.

Table 1: Table showing the learned values from H-DICE at the end of training in GridWorld-v1 and the corresponding ratios obtained from naive division and H-DICE.

Action (a)	Return (z)	$\pi_\theta(a \mid s)$	$h_\omega^{\pi_\theta}(a \mid s)$	Direct Ratio	H-DICE Ratio
LEFT	-100	0.002	0.349	0.006	0.139
LEFT	69	0.002	0.163	0.013	0.015
RIGHT	-100	0.845	0.210	4.019	0.139
RIGHT	69	0.845	0.434	1.949	0.014

Going LEFT from the given state would transition the agent into the fire and likely lead to a return of approximately -100 at the end of the episode. In contrast, moving RIGHT would enable the agent obtain a diamond and is likely lead to a high positive return, such as 69, at the end of the episode. The probability of selecting these respective actions and the corresponding hindsight probabilities are accurately captured in columns 3 and 4 of Table 1. Specifically, note that the probability of choosing the action LEFT when we achieve a reward of -100 is a lot higher than if the reward were 69. A similar trend holds true for the RIGHT action. In short, we observe that the hindsight model qualitatively models the conditional distribution of actions given returns accurately.

The final two columns of the table show the direct $\frac{\pi_\theta(a|s)}{h_\omega^{\pi_\theta}(a|s,z)}$ ratio values which are computed by simply dividing the two required probabilities, and the approximated ratio obtained via H-DICE using Equation 4. We observe that the direct ratio – which is a more accurate estimate of the hindsight ratio of interest – has much larger variation: values fluctuate from 0.006 to 4.019 even in a simple environment such as GridWorld-v1. On the other hand, ratios computed using H-DICE are much lower variation and only lie between 0.014 and 0.139 while still mirroring the trends of the direct ratio. For instance, for the RIGHT action, the direct ratio is 4.019 and 1.949 for returns of -100 and 69 respectively. The corresponding H-DICE ratios are 0.139 and 0.014, which are correlated to the true ratios. We posit that this ability to capture the direct ratio's trends while reducing variance is a core reason for the improved performance of H-DICE compared to PPO-HCA. Note, however, that there are times when inaccuracies in auxiliary models prevent H-DICE from reflecting the trends of the direct ratio; an instance of such discrepancies can be seen in the ratios computed for the LEFT action. Nevertheless, as backed by the results in Section 4, the stability which results from H-DICE is crucial to achieving strong empirical results in challenging credit-assignment settings. Lastly, while simple settings such as GridWorld-v1 enable us to interpret model outputs at the level of individual state-action pairs, we suspect that these issues are only exacerbated in more complex environments, further underscoring the importance of credit assignment in deep RL.

4.6 Can we update the auxiliary models with off-policy schedules?

A potential drawback of H-DICE stems from the fact that four models must be updated at once to learn the hindsight ratio, which can lead to greater training overhead and challenges with hyperparameter tuning. This issue is exacerbated by the fact that all these models must be updated in an on-policy fashion as per the methodology detailed in Section 3.3. This overhead could be alleviated if the auxiliary models could be updated with off-policy schedules, i.e., less frequently than the policy network, thereby allowing them to be trained with data collected from previous versions of the policy. To this end, we aim to study the impact of updating the auxiliary models with various off-policy schedules on final performance.

We perform an ablation on GridWorld-v2 and HalfCheetah environment where we progressively slow down the updates to the auxiliary models relative to the update frequency of the policy. In the plots shown in Figure 4, the curves labeled as Nx indicate that the auxiliary models are all updated N times slower than the policy model, and as a result with N times more data (which will contain trajectories collected from previous iterations of the policy). Note that auxiliary models' weights are still reset before their updates.

As seen in Figure 4, there is hardly any performance loss when updating with off-policy schedules. In some instances, for example when the auxiliary models are updated 50x less frequently than the policy in HalfCheetah, there see a small improvement over updating all models in lockstep. We conjecture that this similarity in performance may be a result of the trade-off between updating models in a fully on-policy fashion but limiting the volume of training data for the auxiliary models, and being more off-policy which enables models to be trained with more data.
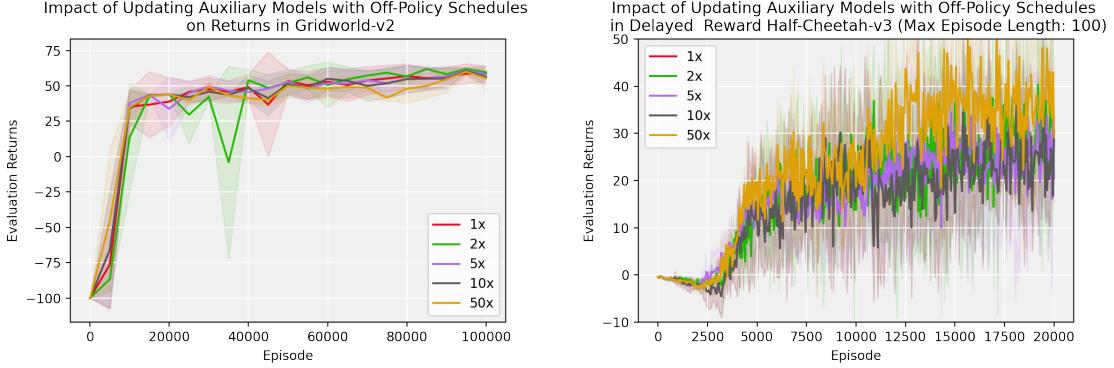
Figure 4: Training curves of varying off policy schedules on GridWorld-v2 and HalfCheetah. There is almost no performance loss when updating with off-policy schedules and in some cases we can even see improvements.

Finally, we note that updating the auxiliary models in an off-policy fashion strays away from the theoretical justification for H-DICE developed in Section 3.3, but enables H-DICE to be computationally efficient with little to no cost to empirical performance.

5 Related Work

The classic workhorse for addressing credit assignment in reinforcement learning has been the eligibility trace [26, 45, 46] which awards credit in a geometrically-decaying fashion to states or state-action pairs on the basis of temporal recency to surprising events, as measured by the temporal-difference (TD) error. While these traditional approaches laid the groundwork for the field, the scale of modern problems makes them woefully insufficient and impractical. More recently, a new class of emphatic TD methods have emerged [50, 20, 57, 2, 12] which includes a so-called interest function that enables more nuanced and non-linear attribution of credit for incident states along a sampled trajectory. The theory surrounding these methods, however, operate in general terms without concrete specification of such an interest function. Since scalable methods for leveraging interest functions when learning option initiation sets exist [49], future work may find it fruitful to learn interest functions using our H-DICE approach in the context of emphatic TD methods. Finally, recent work on expected eligibility traces [55, 12] acknowledges the trajectory as a random variable and aims to learn eligibility traces that award credit to those realizations of the trajectory that were not observed but could have occurred under the given behavior policy; indeed, the hindsight policy studied in this work can be shown to emerge through an information-theoretic treatment of credit assignment where such randomness is accounted for [5]. Still, expected eligibility traces suffer from the same temporal recency heuristic that plagues traditional eligibility traces, whereas the hindsight policy enables non-linear re-weighting of on-policy data.

Our work extends the hindsight credit assignment (HCA) work of Harutyunyan et al. [21], rectifying instabilities in using hindsight policies for credit assignment. While Alipov et al. [1] find that an alternative form of clipping applied to hindsight probabilities can improve stability in Atari games, our experiments demonstrate across a broad spectrum of domains that clipping can greatly stall the speed at which learning progresses with HCA. In contrast, an alternative and scalable approach to tackling credit assignment emerges through the concept of return redistribution [4, 23, 41, 16]. Unfortunately, practical agents employing return redistribution rely on recurrent network architectures whose performance and susceptibility to vanishing gradients rises as the problem horizon increases. Although methods such as Gangwani et al. [16] and Ren et al. [41] do not use recurrent network architectures, they use simplistic methods such as uniform reward decomposition or least squares reward decomposition, which do not necessarily have to assign any meaningful or interpretable credit to the transitions. In contrast, hindsight policies for data re-weighting retain an elegant idea from classic temporal-difference learning; namely, that local information can be stitched together across timesteps to yield globally effective solutions. There have also been recent approaches that explore the relationship between causality and counterfactuals to the RL credit assignment problem [32, 10]; while applying such powerful tools from causal inference does seem like a plausible and even promising avenue for credit assignment in RL, the main drawback to these existing approaches is assuming a perfect causal model of the environment and, with greater limitation, the ability to perform perfect counterfactual inference queries where one factor of variation is manipulated *ceteris paribus* or while keeping everything else fixed. In contrast, H-DICE is theoretically grounded within the framework of importance sampling [21] without such cumbersome assumptions.

6   Conclusion

In this work, we presented H-DICE, a novel algorithm for *stable* credit assignment in deep reinforcement learning. Building on the Hindsight Credit Assignment (HCA) framework introduced by Harutyunyan et al. [21] and noting the instabilities that arise when directly estimating hindsight density ratios, we take inspiration from methods in the off-policy evaluation literature [36] to develop a method which computes the quantities needed to perform credit assignment via HCA in a stable fashion. We showed that in challenging benchmarks with delayed rewards, algorithms that do not assign credit or do so in a naive fashion struggle to perform well while H-DICE converges to higher returns much faster and more consistently than baseline methods.

Our work utilizes a formulation of HCA in which the hindsight policy is conditioned on future outcome information in the form of returns. While this performs well in delayed-reward settings, it requires a diversity of returns to be observed during training in order for the hindsight policy to model something meaningful beyond the policy itself. Hence, it is likely that H-DICE will struggle in sparse-reward settings where returns are either zero or one due to the lack of diversity in observed returns. To address this orthogonal exploration challenge in future work, we hope to extend this work to state-conditioned HCA, a formulation in which the hindsight policy is conditioned on future states in a trajectory rather than the return. Anticipating that it is easier to achieve diversity in state visitation rather than a wide range of potential returns, we suspect this variant is likely to fare better in sparse-reward, language conditioned and goal-reaching settings.

While we conducted some preliminary experiments in an effort to extend H-DICE to operate in an off-policy fashion, the core algorithm is still an on-policy algorithm built atop conventional policy-gradient methods. An interesting line of future-work will be to extend credit assignment to off-policy algorithms such as DQN [34] and SAC [18] which are typically more data efficient than on-policy algorithms. By bridging ideas from OPE to address the credit assignment challenge, we hope to open the door for further symbiotic progress moving forward on both fronts. Developing credit assignment methods that are able to perform expressive, yet stable updates will be crucial for efficient deep RL, in a variety of settings in which meaningful reward feedback is rarely received.

References

[1] Vyacheslav Alipov, Riley Simmons-Edler, Nikita Putintsev, Pavel Kalinin, and Dmitry Vetrov. Towards practical credit assignment for deep reinforcement learning. *arXiv preprint arXiv:2106.04499*, 2021.

[2] Nishanth Anand and Doina Precup. Preferential Temporal Difference Learning. In *International Conference on Machine Learning*, pages 286–296. PMLR, 2021.

[3] Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, OpenAI Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. *Advances in Neural Information Processing Systems*, 30, 2017.

[4] Jose A Arjona-Medina, Michael Gillhofer, Michael Widrich, Thomas Unterthiner, Johannes Brandstetter, and Sepp Hochreiter. RUDDER: Return decomposition for delayed rewards. *Advances in Neural Information Processing Systems*, 32, 2019.

[5] Dilip Arumugam, Peter Henderson, and Pierre-Luc Bacon. An Information-Theoretic Perspective on Credit Assignment in Reinforcement Learning. *arXiv preprint arXiv:2103.06224*, 2021.

[6] Leemon C. Baird III. Advantage Updating. Technical report, Wright Laboratory, Wright-Patterson Air Force Base, 1993.

[7] Marc G. Bellemare, Will Dabney, and Rémi Munos. A distributional perspective on reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 449–458. JMLR.org, 2017.

[8] Richard Bellman. A Markovian Decision Process. *Journal of Mathematics and Mechanics*, pages 679–684, 1957.

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

[10] Lars Buesing, Theophane Weber, Yori Zwols, Sebastien Racaniere, Arthur Guez, Jean-Baptiste Lespiau, and Nicolas Heess. Woulda, coulda, shoulda: Counterfactually-guided policy search, 2018.

[11] Niladri Chatterji, Aldo Pacchiano, Peter Bartlett, and Michael Jordan. On the theory of reinforcement learning with once-per-episode feedback. *Advances in Neural Information Processing Systems*, 34:3401–3412, 2021.

[12] Veronica Chelu, Diana Borsa, Doina Precup, and Hado van Hasselt. Selective credit assignment. *arXiv preprint arXiv:2202.09699*, 2022.

[13] Kristy Choi, Chenlin Meng, Yang Song, and Stefano Ermon. Density ratio estimation via infinitesimal classification. In *International Conference on Artificial Intelligence and Statistics*, pages 2552–2573. PMLR, 2022.

[14] Bo Dai, Ofir Nachum, Yinlam Chow, Lihong Li, Csaba Szepesvári, and Dale Schuurmans. CoinDICE: Off-policy confidence interval estimation. *Advances in neural information processing systems*, 33:9398–9411, 2020.

[15] Adrien Ecoffet, Joost Huizinga, Joel Lehman, Kenneth O. Stanley, and Jeff Clune. Go-explore: a new approach for hard-exploration problems, 2021.

[16] Tanmay Gangwani, Yuan Zhou, and Jian Peng. Learning guidance rewards with trajectory-space smoothing, 2020.

[17] Carles Gelada and Marc G Bellemare. Off-policy deep reinforcement learning by bootstrapping the covariate shift. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3647–3655, 2019.

[18] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor, 2018.

[19] Assaf Hallak and Shie Mannor. Consistent on-line off-policy evaluation. In *International Conference on Machine Learning*, pages 1372–1383. PMLR, 2017.

[20] Assaf Hallak, Aviv Tamar, Rémi Munos, and Shie Mannor. Generalized emphatic temporal difference learning: Bias-variance analysis. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016.

[21] Anna Harutyunyan, Will Dabney, Thomas Mesnard, Mohammad Gheshlaghi Azar, Bilal Piot, Nicolas Heess, Hado P van Hasselt, Gregory Wayne, Satinder Singh, Doina Precup, and Remi Munos. Hindsight credit assignment. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/195f15384c2a79cedf293e4a847ce85c-Paper.pdf.

[22] Ashley Hill, Antonin Raffin, Maximilian Ernestus, Adam Gleave, Anssi Kanervisto, Rene Traore, Prafulla Dhariwal, Christopher Hesse, Oleg Klimov, Alex Nichol, Matthias Plappert, Alec Radford, John Schulman, Szymon Sidor, and Yuhuai Wu. Stable baselines. https://github.com/hill-a/stable-baselines, 2018.

[23] Chia-Chun Hung, Timothy Lillicrap, Josh Abramson, Yan Wu, Mehdi Mirza, Federico Carnevale, Arun Ahuja, and Greg Wayne. Optimizing agent behavior over long time scales by transporting value. *Nature communications*, 10(1):5223, 2019.

[24] Nan Jiang and Lihong Li. Doubly robust off-policy value evaluation for reinforcement learning. In *International Conference on Machine Learning*, pages 652–661. PMLR, 2016.

[25] Leslie Pack Kaelbling, Michael L Littman, and Anthony R Cassandra. Planning and Acting in Partially Observable Stochastic Domains. *Artificial Intelligence*, 101(1-2):99–134, 1998.

[26] A Harry Klopf. *Brain Function and Adaptive Systems: A Heterostatic Theory*. Air Force Cambridge Research Laboratories, Air Force Systems Command, United States Air Force, 1972.

[27] Vijay Konda and John Tsitsiklis. Actor-Critic Algorithms. *Advances in Neural Information Processing Systems*, 12, 1999.

[28] Nathan Lambert, Louis Castricato, Leandro von Werra, and Alex Havrilla. Illustrating Reinforcement Learning from Human Feedback (RLHF), 2022. URL https://huggingface.co/blog/rlhf.

[29] Qiang Liu, Lihong Li, Ziyang Tang, and Dengyong Zhou. Breaking the curse of horizon: Infinite-horizon off-policy estimation. *Advances in Neural Information Processing Systems*, 31, 2018.

[30] Yao Liu, Adith Swaminathan, Alekh Agarwal, and Emma Brunskill. Off-policy policy gradient with state distribution correction. *arXiv preprint arXiv:1904.08473*, 2019.

[31] Yao Liu, Pierre-Luc Bacon, and Emma Brunskill. Understanding the curse of horizon in off-policy evaluation via conditional importance sampling. In *International Conference on Machine Learning*, pages 6184–6193. PMLR, 2020.

[32] Thomas Mesnard, Théophane Weber, Fabio Viola, Shantanu Thakoor, Alaa Saade, Anna Harutyunyan, Will Dabney, Tom Stepleton, Nicolas Heess, Arthur Guez, Éric Moulines, Marcus Hutter, Lars Buesing, and Rémi Munos. Counterfactual credit assignment in model-free reinforcement learning, 2021.

[33] Marvin Minsky. Steps toward Artificial Intelligence. *Proceedings of the IRE*, 49(1):8–30, 1961.

[34] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning, 2013.

[35] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937. PMLR, 2016.

[36] Ofir Nachum, Yinlam Chow, Bo Dai, and Lihong Li. DualDICE: Behavior-Agnostic Estimation of Discounted Stationary Distribution Corrections. *NeurIPS*, 2019.

[37] Aldo Pacchiano, Aadirupa Saha, and Jonathan Lee. Dueling RL: Reinforcement Learning with Trajectory Preferences. In *International Conference on Artificial Intelligence and Statistics*, pages 6263–6289. PMLR, 2023.

[38] Deepak Pathak, Pulkit Agrawal, Alexei A. Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction, 2017.

[39] Doina Precup, Richard S Sutton, and Satinder P Singh. Eligibility Traces for Off-Policy Policy Evaluation. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 759–766, 2000.

[40] Martin L. Puterman. *Markov Decision Processes—Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., New York, NY, 1994.

[41] Zhizhou Ren, Ruihan Guo, Yuan Zhou, and Jian Peng. Learning long-term reward redistribution via randomized return decomposition, 2022.

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

[43] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation, 2018.

[44] Satinder P Singh and Richard S Sutton. Reinforcement learning with replacing eligibility traces. *Machine learning*, 22(1-3):123–158, 1996.

[45] Richard S. Sutton. *Temporal Credit Assignment in Reinforcement Learning*. PhD thesis, University of Massachusetts Amherst, 1984.

[46] Richard S Sutton. Learning to Predict by the Methods of Temporal Differences. *Machine Learning*, 3(1):9–44, 1988.

[47] Richard S Sutton and Andrew G Barto. *Introduction to Reinforcement Learning*. MIT Press, 1998.

[48] Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy Gradient Methods for Reinforcement Learning with Function Approximation. *Advances in Neural Information Processing Systems*, 12, 1999.

[49] Richard S Sutton, Doina Precup, and Satinder Singh. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112(1-2):181–211, 1999.

[50] Richard S Sutton, A Rupam Mahmood, and Martha White. An emphatic approach to the problem of off-policy temporal-difference learning. *The Journal of Machine Learning Research*, 17(1):2603–2631, 2016.

[51] Haoran Tang, Rein Houthooft, Davis Foote, Adam Stooke, Xi Chen, Yan Duan, John Schulman, Filip De Turck, and Pieter Abbeel. Exploration: A study of count-based exploration for deep reinforcement learning, 2017.

[52] Philip Thomas and Emma Brunskill. Data-efficient off-policy policy evaluation for reinforcement learning. In *International Conference on Machine Learning*, pages 2139–2148. PMLR, 2016.

[53] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033, 2012. doi: 10.1109/IROS.2012.6386109.

[54] Masatoshi Uehara, Jiawei Huang, and Nan Jiang. Minimax weight and Q-function learning for off-policy evaluation. In *International Conference on Machine Learning*, pages 9659–9668. PMLR, 2020.

[55] Hado van Hasselt, Sephora Madjiheurem, Matteo Hessel, David Silver, André Barreto, and Diana Borsa. Expected eligibility traces. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 9997–10005, 2021.

[56] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3):229–256, 1992.

[57] Huizhen Yu. On convergence of emphatic temporal-difference learning. In *Conference on Learning Theory*, pages 1724–1751. PMLR, 2015.

[58] Ruiyi Zhang, Bo Dai, Lihong Li, and Dale Schuurmans. GenDICE: Generalized Offline Estimation of Stationary Values. In *International Conference on Learning Representations*, 2020.

[59] Shangtong Zhang, Bo Liu, and Shimon Whiteson. GradientDICE: Rethinking generalized offline estimation of stationary values. In *International Conference on Machine Learning*, pages 11194–11203. PMLR, 2020.

A   Solution to the optimization in Equation 3

In this section, we show that the solution to the following optimization introduced in Section 3.3:

$$\min_{\phi:S\times A\times Z\to\mathcal{C}} \frac{1}{2}\mathbb{E}_{(s,a,z)\sim\mathcal{D}^{h^{\pi_\theta}}}\left[\phi(s,a,z)^2\right] - \mathbb{E}_{\substack{(s,a)\sim d^{\pi_\theta}\\z\sim\psi(z)}}\left[\phi(s,a,z)\right]$$

is given by

$$\phi^*(s,a,z) = \frac{d^{\pi_\theta}(s,a)\psi(z)}{\mathcal{D}^{h^{\pi_\theta}}(s,a,z)}$$

One can also consult the salient exposition of Nachum et al. [36] for an alternative rationale and calculation but specialized to their OPE setting instead of our credit assignment focus. We begin by expanding the expectations inside the objective:

$$\frac{1}{2}\mathbb{E}_{(s,a,z)\sim\mathcal{D}^{h^{\pi_\theta}}}\left[\phi(s,a,z)^2\right] - \mathbb{E}_{\substack{(s,a)\sim d^{\pi_\theta}\\z\sim\psi(z)}}\left[\phi(s,a,z)\right]$$

$$= \frac{1}{2}\int_{s\in S,a\in A,z\in Z}\mathcal{D}^{h^{\pi_\theta}}(s,a,z)\phi(s,a,z)^2 ds da dz - \int_{s\in S,a\in A,z\in Z}d^{\pi_\theta}(s,a)\psi(z)\phi(s,a,z)ds da dz$$

$$= \int_{s\in S,a\in A,z\in Z}\left(\frac{1}{2}\mathcal{D}^{h^{\pi_\theta}}(s,a,z)\phi(s,a,z)^2 - d^{\pi_\theta}(s,a)\psi(z)\phi(s,a,z)\right)ds da dz$$

Minimizing this quantity can be achieved by enforcing first-order optimality conditions. We first differentiate the above integral using the Leibniz Integral Rule, which allows for the derivative to move inside the integral:

$$\int_{s\in S,a\in A,z\in Z}\frac{\partial}{\partial\phi}\left(\frac{1}{2}\mathcal{D}^{h^{\pi_\theta}}(s,a,z)\phi(s,a,z)^2 - d^{\pi_\theta}(s,a)\psi(z)\phi(s,a,z)\right)ds da dz$$

$$= \int_{s\in S,a\in A,z\in Z}\left(\mathcal{D}^{h^{\pi_\theta}}(s,a,z)\phi(s,a,z) - d^{\pi_\theta}(s,a)\psi(z)\right)ds da dz$$

We now equate this expression for the derivative to zero and solve for ϕ. Similar to the proof in DualDICE [36], we look for the value of $\phi(s,a,z)$ such that this gradient is equal to zero pointwise over all state-action-return triples and get:

$$\phi^*(s,a,z) = \frac{d^{\pi_\theta}(s,a)\psi(z)}{\mathcal{D}^{h^{\pi_\theta}}(s,a,z)}$$

which we can then write as:

$$\phi^*(s,a,z) = \frac{d^{\pi_\theta}(s)\pi_\theta(a\mid s)\psi(z)}{\mathcal{D}^{h^{\pi_\theta}}(s)\chi^{\pi_\theta}(z\mid s)h_\omega^{\pi_\theta}(a\mid s,z)}$$

$$\implies \phi^*(s,a,z) = \frac{\pi_\theta(a\mid s)}{\chi^{\pi_\theta}(z\mid s)h_\omega^{\pi_\theta}(a\mid s,z)} \tag{5}$$

The final simplification follows from the fact that $\psi(z)$ is chosen to be a uniform distribution over all possible returns; hence, $\psi(z)$ is a constant term independent of the precise realiziation z, therefore, can be ignored. Ignoring this constant simply changes the magnitude of all policy updates – this is preferred over dividing by the constant, which again, introduces the risk of instability.

For instance, the policy updates are smaller in magnitude when the advantage $A(s, a) \geq 0$ because:

$$\frac{d^{\pi_\theta}(s)\pi_\theta(a \mid s)\psi(z)}{\mathcal{D}^{h^{\pi_\theta}}(s)\chi^{\pi_\theta}(z \mid s)h_\omega^{\pi_\theta}(a \mid s, z)} \leq \frac{d^{\pi_\theta}(s)\pi_\theta(a \mid s)}{\mathcal{D}^{h^{\pi_\theta}}(s)\chi^{\pi_\theta}(z \mid s)h_\omega^{\pi_\theta}(a \mid s, z)}$$

$$\implies \frac{\pi_\theta(a \mid s)\psi(z)}{h_\omega^{\pi_\theta}(a \mid s, z)} \leq \frac{\pi_\theta(a \mid s)}{h_\omega^{\pi_\theta}(a \mid s, z)}$$

$$\implies \left(1 - \frac{\pi_\theta(a \mid s)\psi(z)}{h_\omega^{\pi_\theta}(a \mid s, z)}\right) A(s, a) \geq \left(1 - \frac{\pi_\theta(a \mid s)}{h_\omega^{\pi_\theta}(a \mid s, z)}\right) A(s, a)$$

and vice-versa when $A(s, a) < 0$.

$d^{\pi_\theta}(s) = \mathcal{D}^{h^{\pi_\theta}}(s)$ follows from the fact that h^{π_θ} is the hindsight policy learned from the state-visitation distribution of π_θ. Empirically, we use the same data that we collected from the policy to train the hindsight distribution as well and we update them with the same schedule.

This finally gives us the following expression for the hindsight ratio of interest:

$$\frac{\pi_\theta(a \mid s)}{h_\omega^{\pi_\theta}(a \mid s, z)} = \phi^*(s, a, z)\chi^{\pi_\theta}(z \mid s) \tag{6}$$

There are several choices for $\psi(z)$, which in principle can be any valid distribution over the returns; a good choice of $\psi(z)$ would enable easy computation of the hindsight ratio of interest. There are several potential choices for $\psi(z)$, including $\chi^\pi(z \mid s, a)$ or $\chi^\pi(z \mid s)$. Using the first distribution would require values of ϕ to be divided by values of this distribution, risking instability issues that arise when having probabilities or likelihoods as a divisor (indeed, this is the problem we aimed to avoid in the first place). Sampling returns with respect to the latter distribution is especially appealing since it conveniently cancels out with the return predictor distribution which multiples with the output of ϕ to obtain the hindsight ratio in Equation 6. However, as shown in Appendix C, sampling returns from $\chi^\pi(z \mid s)$ in the second expectation empirically yields poorer performance than sampling returns from a uniform distribution. We leave other choices for $\psi(z)$ to future work.

Note that we cannot use an arbitrary distribution to model the returns in $\mathcal{D}^{h^{\pi_\theta}}(s, a, z)$ because we need to decompose this term into $\mathcal{D}^{h^{\pi_\theta}}(s)\chi^{\pi_\theta}(z \mid s)h^{\pi_\theta}(a \mid s, z)$. Here, it is important that the returns are sampled from the policy π_θ and that the distribution χ^{π_θ} is the distribution of returns achieved from a state *under policy* π_θ, as this is what the hindsight policy is conditioned on.

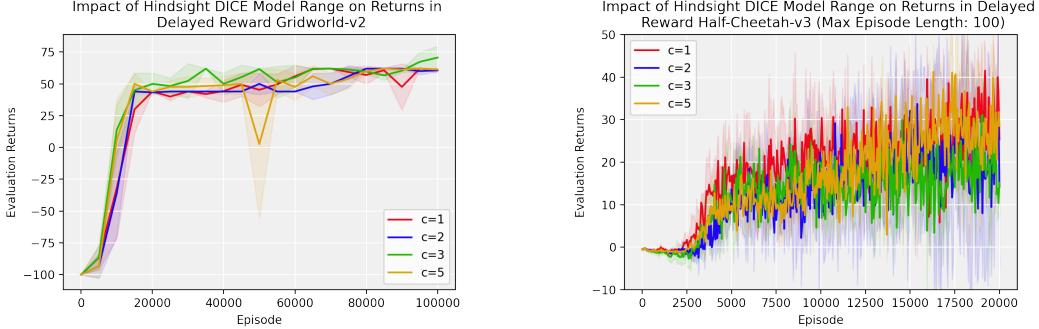# B   Ablation: Impact of Hindsight DICE Model Range

Figure 5: Impact of the range of the Hindsight DICE on policy performance in Gridworld-v2 (left) and Delayed Reward Half-Cheetah with a maximum episode length of 100. For a given value of C, the output range of the Hindsight DICE model is set to be $[0, C]$. The performance of H-DICE is similar across choices of C.

Figure 5 studies the impact of the output range of the Hindsight DICE model (which is controlled by C, as detailed in Section 3.3) on policy performance. All policies are trained with H-DICE, and the only axis of variation across experiments in each environment is the choice of C. Results are averaged across 3 seeds for each choice of C. Policy performance is similar across choices of C in the Gridworld-v2 and delayed reward Half-Cheetah environments indicating that the choice of this hyperparameter is not very important.

We further note that when $C = 1$, the hindsight ratio calculate by H-DICE lies in the range $[0, 1]$. As a result, the model is not capable of making "double-sided" updates, i.e., down-weighting an action at a state even when the observed returns from that state are positive (and vice versa). This is no longer the case when $C > 1$ but this boost in expressivity risks a potential cost to stability as evidenced by the results of PPO-HCA.

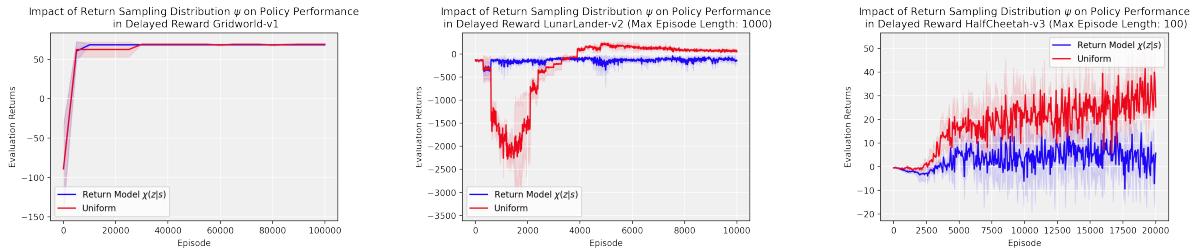# C   Ablation: Choice of ψ return distribution

Figure 6: Impact of return sampling distribution ψ on policy performance in Gridworld-v1 and delayed reward LunarLander-v2 and Half-Cheetah-v3. Sampling rewards with a uniform distribution outperforms sampling with respect to the conditional distribution $\chi(z|s)$ in the more difficult LunarLander and HalfCheetah settings.

Figure 6 examines the impact of ψ – the distribution from which returns are sampled in the second expectation in expression 3 – on final performance in Gridworld-v1, LunarLander-v2 with delayed rewards, and Half-Cheetah-v3 with delayed rewards. All policies again are trained with H-DICE, with only the choice of ψ changing between experiments. As justified in Appendix A, we study two options for ψ: a uniform distribution over returns, and the state-conditioned return distribution $\chi^{\pi_\theta}(z|s)$. Note that computing the hindsight ratio using the uniform distribution requires $\chi^{\pi_\theta}(z|s)$ to be multiplied with $\phi^*(s, a, z)$, while using $\chi^{\pi_\theta}(z|s)$ enables the hindsight ratio to be directly obtained from $\phi^*(s, a, z)$. Despite both choices being theoretically justified, the results in Figure 6 clearly demonstrate that sampling with the uniform distribution results results in superior performance as compared to sampling from $\chi^{\pi_\theta}(z|s)$. We hypothesise that this is because the Hindsight DICE model alone is not sufficient to capture the complex credit assignment structures in an environment and the scaling offered by the return model is crucial in learning more expressive hindsight ratios. This is further supported by the fact that both choices of ψ perform similarly in the GridWorld but using the state-conditioned return distributions struggles in more complex environments like LunarLander and HalfCheetah.
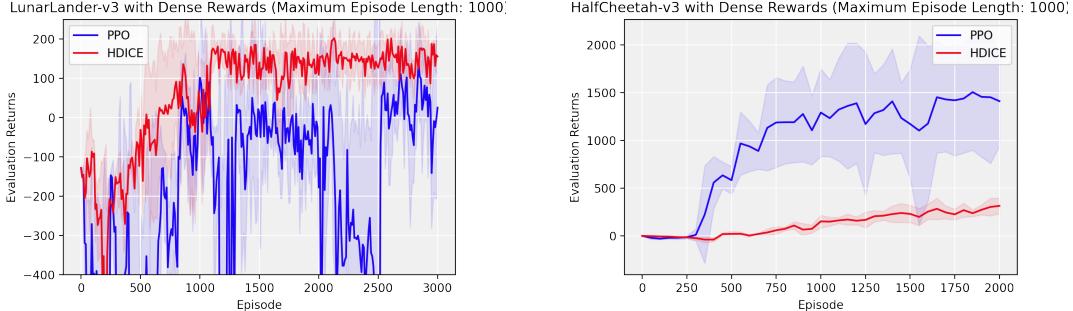
D   Using H-DICE in dense reward settings

Figure 7: Performance of H-DICE and PPO in LunarLander-v3 and HalfCheetah-v3 (both with maximum episode lengths of 1000) in dense reward settings. Note that these are the default settings of the environments. H-DICE learns stably and quickly in LunarLander, outperforming PPO. However, H-DICE despite learning stably, is much slower than PPO in HalfCheetah.

In Figure 7, we compare the performance of H-DICE with PPO in a dense reward setting, where we don't anticipate a pressing need for effective credit assignment. We seek to answer whether H-DICE can be a competitive substitute to the GAE advantage calculation often used with PPO in a wide range of settings. From the results we can see that H-DICE learns more stably and achieves marginally better performance than PPO on LunarLander over three seeds but is outperformed by PPO on HalfCheetah. We also note that H-DICE was more stable, in terms of variation across seeds, than PPO in both environments.

E   Training Details and Hyperparameters

In this section, we provide training details and hyperparameters for all environments and all methods.

To aid in the stability of training, $\chi_\eta^{\pi_\theta}, h_\omega^{\pi_\theta}$, and ϕ_ν utilize techniques such input normalization and clipping of gradient norms. We additionally limit the output range of ϕ_ν by applying a sigmoid at the output to improve training stability (a technique also used by [36]), and also find that collecting large amounts of data between updates improved performance.

For training the policy model, we used the usual PPO loss combined with the advantage calculation from 1. To train the Hindsight DICE model, we used the loss from equation 3. To train the hindsight model, we used a cross-entropy loss or a Gaussian log-likelihood loss depending on whether the actions were discrete or continuous. For the return model, we used a Gaussian log-likelihood loss to predict the mean returns from that state and used this distribution to obtain the probability of the given returns.

An extensive list of hyperparameters and the values used for each domain are provided below. While we swept across some of the key hyperparameters such as value loss coefficient, entropy coefficent, number of epochs, etc., other hyperparameters were chosen from Stable Baselines [22]. This is by no means an extensive hyperparameter sweep, especially for the HCA and H-DICE models which have additional hyperparameters as compared to vanilla PPO.

We used PyTorch for our experiments and training was done on single NVIDIA RTX A5000 GPU. The code will be released on acceptance.

E.1   GridWorld experiments

Table 2: Hyperparameters for the GridWorld experiments for all methods

	PPO	PPO-HCA	PPO-HCA-Clip	H-DICE
Entropy Coefficient	0.1	0.1	0.1	0.1
Value Loss Coefficient	1e-4	-	-	-
PPO Learning Rate	3e-4	3e-4	3e-4	3e-4
Update Every Episodes	50	50	50	50
Epsilon Clip	0.2	0.2	0.2	0.2
PPO Epochs	30	30	30	30
GAE Lambda	0.95	-	-	-
Gamma	0.99	0.99	0.99	0.99
PPO Max Grad Norm	-	-	-	-
HCA Max Grad Norm	-	10	10	10
Density Pred Max Grad Norm	-	-	-	10
Return Pred Max Grad Norm	-	-	-	10
Hindsight Policy Epochs	-	10	10	10
Hindsight DICE Model Epochs	-	-	-	10
Return Model Epochs	-	-	-	10
Auxiliary Models Learning Rate	-	3e-4	3e-4	3e-4
Auxiliary Models Batchsize	-	256	256	256
Return Model Normalize Targets	-	-	-	True

The PPO policy was initialized with a trunk of 2 hidden layers with 64 ReLU activated hidden units each. The value head and actor head were initialized as separate heads on top of this for PPO. There is no value head for the HCA and H-DICE models.

For the HCA and H-DICE models, each auxiliary model was initialized with 2 layers with 128 ReLU activated hidden units. In the table above, "auxiliary models" refers to the Hindsight Policy, Hindsight DICE Model and Return predictor together.

E.2   LunarLander experiments

The hyperparameters that changed from Table 2 are presented below. The rest of the hyperparameters are the same.

Table 3: Hyperparameters for the LunarLander-v2 experiments for all methods

	PPO	PPO-HCA	PPO-HCA-Clip	H-DICE
Entropy Coefficient	0.0	0.0	0.0	0.01
Value Loss Coefficient	0.5	-	-	-
PPO Learning Rate	3e-4	3e-5	3e-4	3e-4
Update Every Episodes	300	300	300	300
PPO Epochs	80	80	80	80
PPO Max Grad Norm	0.5	0.5	0.5	0.5
Hindsight Epochs	-	20	20	20
Hindsight DICE Model Epochs	-	-	-	1
Return Model Epochs	-	-	-	20

This time, the PPO policy was initialized with a trunk of 3 hidden layers with 128 ReLU activated hidden units each.

For the HCA and H-DICE models, each auxiliary model was initialized with 2 layers with 128 ReLU activated hidden units.

E.3 HalfCheetah experiments

Once again, the hyperparameters that changes from Table 2 are presented below and the other hyperparameters stay the same.

Table 4: Hyperparameters for the HalfCheetah-v3 experiments for all methods

	PPO	PPO-HCA	PPO-HCA-Clip	H-DICE
Entropy Coefficient	0.01	0.01	0.01	0.01
Value Loss Coefficient	0.5	-	-	-
PPO Learning Rate	3e-4	3e-5	3e-4	3e-4
Update Every Env Steps	6144	6144	6144	6144
PPO Epochs	80	80	80	80
PPO Max Grad Norm	0.5	0.5	0.5	0.5

The PPO policy was initialized with a trunk of 3 hidden layers with 128 ReLU activated hidden units each.

For the HCA and H-DICE models, each auxiliary model was initialized with 2 layers with 128 ReLU activated hidden units.

F   GridWorld-v2

We provide an image of GridWorld-v2 in this section. Note that the dynamics and reward structure of GridWorld-v2 are the same as in GridWorld-v1 as described in Section 4.1 with the exception of the maximum episode length, which is set to 100 in GridWorld-v2 (as opposed to 50 in GridWorld-v1).
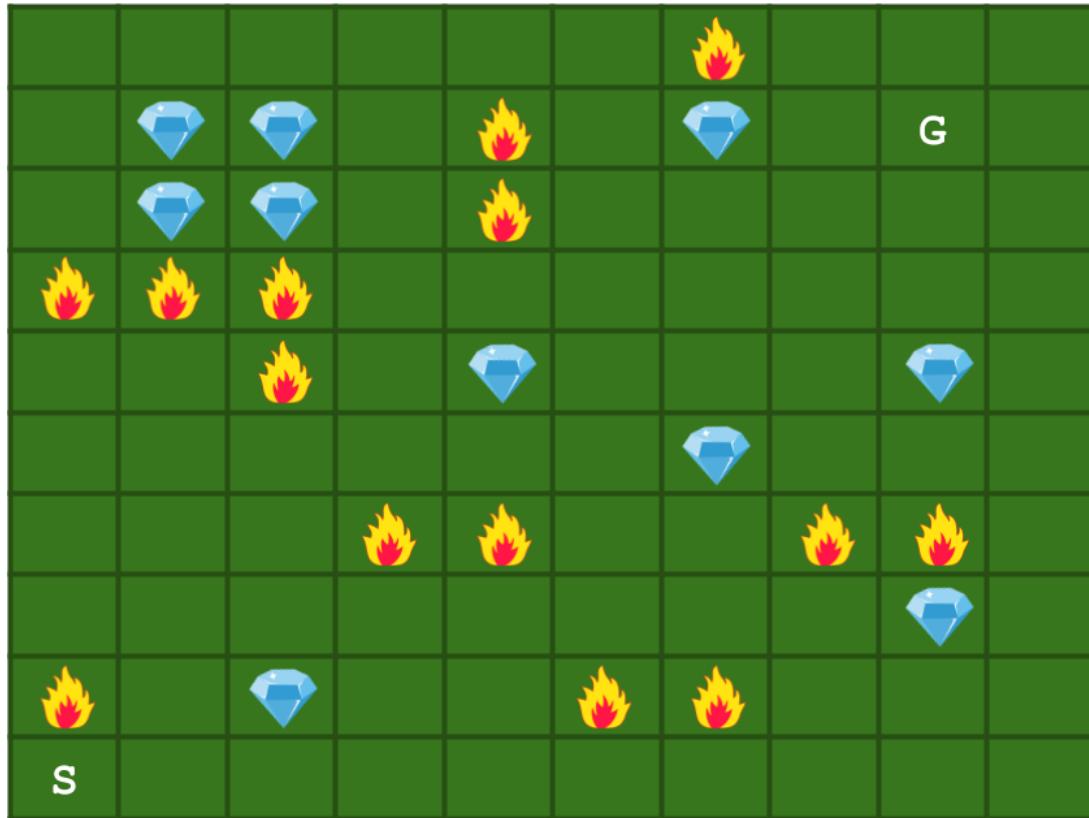

Figure 8: The GridWorld-v2 Environment