Decoupling Exploration and Exploitation in Reinforcement Learning

Anonymous Authors[1]

Abstract

Intrinsic rewards are commonly applied to improve exploration in reinforcement learning. However, these approaches suffer from non-stationary reward shaping and strong dependency on hyperparameters. In this work, we propose Decoupled RL (DeRL) which trains separate policies for exploration and exploitation. DeRL can be applied with on-policy and off-policy RL algorithms. We evaluate DeRL algorithms in two exploration-focused environments with five types of intrinsic rewards. We show that DeRL can be more robust to scaling of intrinsic rewards and converge to the same evaluation returns than intrinsically motivated baselines in fewer interactions.

1. Introduction

Exploration is one of the essential challenges in reinforcement learning (RL). RL algorithms often use simple randomised methods, e.g. applying ϵ-greedy policies (Watkins, 1989) or adding random noise to continuous actions (Lillicrap et al., 2015). Such exploration techniques are inefficient on tasks where rewards are sparse. One category of exploration techniques which has been found to be particularly effective in sparse-reward environments are intrinsic rewards (Chentanez et al., 2005; Oudeyer et al., 2007; Pathak et al., 2017; Burda et al., 2018b; Raileanu & Rocktäschel, 2020). These can serve as strong incentives for exploration and assist training in hard exploration problems where a steady stream of rewards is missing.

Two common approaches to define intrinsic rewards r^i are self-supervised predictions in the environment (Schmidhuber, 1991; Pathak et al., 2017; Burda et al., 2018b; Raileanu & Rocktäschel, 2020) and computing (pseudo-) counts of states (Strehl & Littman, 2008; Ostrovski et al., 2017; Tang et al., 2017). These intrinsic rewards are added to the extrinsic reward r^e provided by the environment to define a combined reward signal $r = r^e + \lambda r^i$ with some weighting factor λ. However, adding such an intrinsic reward to the extrinsic objective and optimise for this combined feedback introduces three challenges. (1) Intrinsic rewards are designed to incentivise exploration and naturally decay as more exploration has been completed. Therefore, the intrinsic rewards are constantly changing. Such **non-stationary reward shaping** violates the Markov assumption and can cause major instabilities due to the addition of a moving target. (2) Careful selection of λ for **intrinsic reward scaling is essential** for their effective application. If intrinsic rewards are too large, they might heavily distort training for an extended period of time and effectively introduce non-stationary noise to the optimisation. Small intrinsic rewards will have no impact and not incentivise exploration as intended. (3) These exploration methods are designed to decay throughout training, but their **speed of decay is essential** to training. Similar to the scale of the intrinsic exploration bonuses, slowly decaying intrinsic rewards might disrupt training whereas quickly vanishing intrinsic rewards might have no significant impact on exploration.

These challenges lead to a significant dependency of intrinsic rewards on hyperparameters to determine their scale and decay. Besides significant sensitivity to hyperparameters, these dependencies also make the application of intrinsic rewards highly task-dependent because the optimal scale and decay strongly depends on the scale of extrinsic rewards and required exploration in the respective task. All these properties make the practical application of such methods difficult (Taïga et al., 2019).

Motivated by these challenges and success in off-policy RL (Degris et al., 2012; Silver et al., 2014; Haarnoja et al., 2018; Fujimoto et al., 2018), we propose to separate the RL training into two separate policies. We train a policy π_β for exploration with intrinsic rewards, which we will refer to as the *exploration policy*. Alongside, we train an *exploitation policy* π_e only using extrinsic rewards on the data collected by the exploration policy. We refer to this approach as **Decoupled RL** (DeRL). Using such decoupling alleviates many of the challenges of previous application of intrinsic rewards. The exploration policy is optimised using the combined objective of extrinsic and intrinsic rewards as for typical intrinsically motivated RL, but it is not required to be an effective policy for application in the task. Instead,

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

π_β is only trained to generate data which can be used to train the exploitation policy. Therefore, DeRL can be less sensitive to scale and speed of decay of intrinsic rewards and π_e is not affected by non-stationary reward shaping as it is only trained using extrinsic rewards.

We implement and evaluate on-policy and off-policy versions of DeRL with five types of intrinsic rewards in two learning environments that focus on exploration. The exploitation policy is able to converge to the optimal policy using fewer interactions in a series of tasks compared to traditional intrinsically motivated RL baselines. We also analyse the sensitivity of DeRL and RL baselines to the scale of intrinsic rewards and verify the dependency of these methods on the scale of intrinsic rewards. However, we observe that DeRL still suffers from significant instabilities caused by the off-policy optimisation of the exploitation policy π_e when π_e and π_β diverge, and propose future work to address these shortcomings.

2. Related Work

A variety of computational methods have been proposed to replicate the exploration incentive of curiosity for RL (Barto, 2013; Oudeyer & Kaplan, 2009; Oudeyer et al., 2008). The underlying idea behind these approaches is fairly simple: agents should be incentivised to explore novel or poorly understood parts of the environment. Therefore, an agent learns an intrinsic reward which rewards the agent for visiting novel states. Over time, the agent should become less "curious" with completed exploration and exploitation will gradually take over. There are two common branches of intrinsic rewards for exploration: (1) count-based and (2) prediction-based curiosity. Below, we will provide an overview over several of these intrinsic reward approaches.

2.1. Count-based Curiosity

Count-based intrinsic rewards define an intrinsic reward which is inverse proportional to the count of visitations of encountered states, as shown in Equation (1).

$$r_t^i := \frac{1}{\sqrt{N(s_t)}} \tag{1}$$

Thereby, agents are incentivised to visit states within the environment which are less frequently encountered. Likewise, agents are discouraged from visiting frequently encountered states which are deemed less valuable for exploration. While this approach is easily applicable in small, discrete state spaces, pseudo-counts have to computed for large or continuous state spaces where encountering any state multiple times is rare. These pseudo-counts can be computed using various techniques to group highly similar states together.

Density models: Bellemare et al. (2016) and Ostrovski et al. (2017) achieve such grouping with a density model predicting the visitations of states. Such pseudo-counts improved exploration and thereby learning progress considerably in multiple Atari games (Bellemare et al., 2013).

Hash functions: Tang et al. (2017) propose a different technique to generalise pseudo-counts across similar states by using a locality-sensitive hash function (Andoni & Indyk, 2008) for states which is more efficient to compute compared to deep density models. The paper specifically proposes the SimHash (Charikar, 2002) function measuring angular distance based on the sign of a randomised mapping.

2.2. Prediction-based Curiosity

A separate approach defines intrinsic rewards using predictions in the environment. Schmidhuber (1991) proposed an intrinsic reward defined as the prediction error of predicting the next state given the current state and action. However, stochastic and thereby unpredictable dynamics within the environment would lead to the exploration signal constantly remaining high at the face of unpredictability. While a concept of boredom has been proposed to decay rewards for "not learnable" dynamics, this remains a major challenge of these approaches and is commonly called the "noisy TV problem" (Burda et al., 2018a).

Intrinsic curiosity module (ICM): Pathak et al. (2017) propose the intrinsic curiosity module to learn efficient state representations $\phi(s)$ and assign an intrinsic reward for the prediction error of the next state

$$r_t^i := \left(\widehat{\phi}(s_{t+1}) - \phi(s_{t+1})\right)^2 \tag{2}$$

where ϕ is a learned self-supervised state-representation function trained using an inverse-dynamics objective: given a representation of current state, $\phi(s_t)$, and next state, $\phi(s_{t+1})$, predict the applied action a_t. Through this feature representation, the model only learns to encode information of states which might be affected through the agent's actions. Using such representation, the prediction error is based on the prediction $\widehat{\phi}(s_{t+1})$ of the next state given current state s_t and applied action a_t.

Rewarding impact-driven exploration (RIDE): Raileanu & Rocktäschel (2020) propose reward impact-driven exploration which aims to reward the agent for applying actions which lead to significant change in the environment. Such impact is defined as the difference between embeddings of consecutive states, where the embedding function is trained using an inverse dynamics model identical to ICM (Pathak et al., 2017). In order to avoid the agent going back and forth between a group of states, an episodic state-count is

added to the objective:

$$r_t^i := \frac{(\phi(s_{t+1}) - \phi(s_t))^2}{\sqrt{N_{ep}(s_{t+1}}} \tag{3}$$

where ϕ is the trained state embedding and N_{ep} represents the episodic pseudo-count of states.

Random network distillation (RND): Burda et al. (2018b) propose a simplified prediction-based intrinsic reward for exploration based on two observations: (1) training and computation of complex models of dynamics is expensive and (2) static random feature representation is surprisingly effective for state predictions (Burda et al., 2018a). Random network distillation proposes to compute a representation of encountered states trained to mimic a randomly initialised, fixed target representation

$$r_t^i := \left(\widehat{\phi}(s_t) - \phi(s_t)\right)^2 \tag{4}$$

where ϕ is a randomised state representation function and $\widehat{\phi}$ is trained to represent ϕ.

2.3. Off-Policy Reinforcement Learning

RL trains a policy to collect data and to maximise the extrinsic returns in the environment. Exploration and exploitation typically apply different policies, but these are often derived from each other, e.g. a greedy policy for exploitation. Such training leads to the *exploration-exploitation problem* as the amount of exploration and exploitation has to be carefully balanced. We propose to separate the training of two distinct policies for exploration and exploitation, but training the exploitation policy using experience generated by the exploration policy requires off-policy RL. Off-policy RL is concerned with the optimisation of a policy π_e using experience generated within the environment by following a separate behaviour policy π_β. Separating exploration and exploitation enables the exploration policy to be trained with targeted exploration without modifying the training objective of the exploitation policy. However, one major challenge of off-policy RL is *distribution shift*, caused by differences of behaviour policy and trained policy (Fujimoto et al., 2018). Such differences cause the state distributions and action distributions of both policies to diverge and can cause significant instabilities during training.

2.4. Unbiased Task Policy

Whitney et al. (2021) recently proposed to separately train an exploration and task policy using off-policy RL. In their work, they train a task policy separate from any intrinsic rewards and concurrently train an exploration policy using intrinsic rewards. Their exploration policy is defined as a factored policy and directly tied to the task policy. However, in their work they solely focused on off-policy task policies using soft Double-DQN (van Hasselt et al., 2016) and further optimisations focused on fast adaptation of the exploration policy. Intrinsic rewards are defined using a simple count-based definition which is also used for optimisitic initialisation (Rashid et al., 2020) but no further intrinsic reward definitions are considered. In contrast, we consider a range of five different definitions of intrinsic rewards and find that on-policy training of the task policy leads to better results with off-policy correction.

3. Decoupled Reinforcement Learning

In this work, we propose to decouple exploration and exploitation into two separate policies. We train an exploration policy π_β with the intent to explore the environment. Using the data collected by the exploration policy, a separate exploitation policy π_e is trained.

Formally, an agent trains a exploration policy π_β which is trained to maximise the sum of intrinsic and extrinsic rewards to achieve exploration,

$$\pi_\beta := \arg\max_\pi \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \cdot \left(R_t^e + \lambda R_t^i\right) \mid A_t \sim \pi(S_t)\right] \tag{5}$$

$$= \arg\max_\pi \mathbb{E}\left[G_t^{e+i} \mid A_t \sim \pi(S_t)\right] \tag{6}$$

with G^{e+i} denoting the discounted returns computed using the combination of extrinsic and intrinsic rewards with scaling factor λ and discount factor $\gamma \in [0, 1)$. During training of π_β, experience samples $(s_t, a_t, r_t^e, s_{t+1})$ with only extrinsic rewards are collected in a set $\mathcal{D}$.

In addition to this typical intrinsically motivated RL, we propose a separate exploitation policy π_e which is trained to maximise only expected cumulative extrinsic rewards using experience accumulated in $\mathcal{D}$,

$$\pi_e := \arg\max_\pi \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R_t^e \mid (S_t, A_t, R_t^e, S_{t+1}) \sim \mathcal{D}\right] \tag{7}$$

$$= \arg\max_\pi \mathbb{E}\left[G_t^e \mid (S_t, A_t, R_t^e, S_{t+1}) \sim \mathcal{D}\right] \tag{8}$$

with G_t^e denoting the discounted extrinsic returns.

Both exploration π_β and exploitation policy π_e can be trained using any RL algorithm given the defined objectives. Generally, the exploration policy π_β is optimised as typical RL with intrinsic rewards, whereas π_e is trained every T_{Dec} timesteps on experience from $\mathcal{D}$ which is generated by π_β's interaction in the environment. Note that $\mathcal{D}$ only contains extrinsic rewards and is off-policy data for the optimisation of π_e as it was generated by following π_β. Below, we propose two methods to apply such decoupled RL in an on-policy and off-policy fashion.

Algorithm 1 Decoupled On-Policy Learning

Initialise: parameters ϕ, θ and π_β
$\mathcal{D} \leftarrow \emptyset$
$i \leftarrow 0$
for $ep = 0, \ldots N_{ep}$ **do**
  $a_t \sim \pi_{beh}(s_t)$
  $s_{t+1}, r_t^e \leftarrow$ environment step with a_t
  Update π_{beh}
  $\mathcal{D} \leftarrow \mathcal{D} \cup (s_t, a_t, r_t^e, s_{t+1})$
  $i \leftarrow i + 1$
  if $i \mod T_{Dec} = 0$ **then**
    Update ϕ with Equation (9) and $\mathcal{D}$
    Update θ with Equation (12) and $\mathcal{D}$
    $\mathcal{D} \leftarrow \emptyset$
  end if
end for

Algorithm 2 Decoupled Off-Policy Learning

Initialise: parameters θ and π_β
$\mathcal{D} \leftarrow \emptyset$
$i \leftarrow 0$
for $ep = 0, \ldots N_{ep}$ **do**
  $a_t \sim \pi_{beh}(s_t)$
  $s_{t+1}, r_t^e \leftarrow$ environment step with a_t
  Update π_{beh}
  $\mathcal{D} \leftarrow \mathcal{D} \cup (s_t, a_t, r_t^e, s_{t+1})$
  $i \leftarrow i + 1$
  if $i \mod T_{Dec} = 0$ **then**
    Update θ with Equation (13) and $\mathcal{D}$
  end if
end for

3.1. Decoupled On-Policy RL

Training of π_e uses generated data $\mathcal{D}$ from π_β which is off-policy data for π_e. In order to apply on-policy RL algorithms to train π_e, off-policy correction can be applied to account for differences in trajectory distributions of both policies.

One technique for off-policy correction is *importance sampling* (IS). In the following, we assume π_e is trained using an on-policy actor-critic RL algorithm with state value function V, parameterised by θ, and policy π_e, parameterised by ϕ. The latter are optimised by minimising the actor loss given in Equation (9)

$$\mathcal{L}(\phi) = \mathbb{E}\Big[-\rho(A_t|S_t) \log \pi_e(A_t|S_t;\phi)\, \widehat{G}(S_t) \mid (S_t, A_t, R_t^e, S_{t+1}) \sim \mathcal{D}\Big] \quad (9)$$

with $\widehat{G}(S_t)$ being a bootstrapped return estimate

$$\widehat{G}(S_t) = (R_t^e + \gamma V(S_{t+1};\theta) - V(S_t;\theta)) \quad (10)$$

and IS weights

$$\rho(A_t|S_t) = \frac{\pi_e(A_t|S_t;\phi)}{\pi_\beta(A_t|S_t)} \quad (11)$$

Similarly, the value loss for π_e using IS weights can be defined as follows:

$$\mathcal{L}(\theta) = \mathbb{E}\Big[\rho(A_t|S_t)\ \big(V(S_t;\theta) - (R_t^e + \gamma V(S_{t+1};\theta))\big)^2 \mid (S_t, A_t, R_t^e, S_{t+1}) \sim \mathcal{D}\Big] \quad (12)$$

IS weights ρ can cause instabilities during off-policy training through exploding weights when $\pi_e(A_t|S_t;\phi) \gg \pi_\beta(A_t|S_t)$ or vanishing weights for $\pi_e(A_t|S_t;\phi) \ll \pi_\beta(A_t|S_t)$. In particular, such instabilities occur when one policy assigns approximately zero probability for some action. Various techniques have been proposed to address such instabilities, including clipping of importance weights (Munos et al., 2016; Espeholt et al., 2018) to minimise vanishing or exploding gradients.

The pseudocode for the optimisation of π_e using on-policy RL can be found in Algorithm 1.

3.2. Decoupled Off-Policy RL

Instead of optimising π_e using on-policy RL algorithms with off-policy correction such as importance sampling, any off-policy RL algorithm can be applied without the need for any correction. In this work, we consider optimising π_e using off-policy Deep Q-Networks (DQN) (Mnih et al., 2015). For DQN optimisation, the following Q-learning loss is minimised

$$\mathcal{L}(\theta) = \mathbb{E}\Big[\big(Q(S_t, A_t;\theta) - \bar{Q}(S_t)\big)^2 \mid (S_t, A_t, R_t^e, S_{t+1}) \sim \mathcal{D}\Big] \quad (13)$$

with target Q-values

$$\bar{Q}(S_t) = (R_t^e + \gamma \max_{a'} Q(S_{t+1}, a';\bar{\theta})) \quad (14)$$

and $\bar{\theta}$ denoting the parameters of the periodically update target network. Pseudocode for off-policy optimisation of π_e can be found in Algorithm 2. Note that $\mathcal{D}$ is only used for a single update in decoupled on-policy learning, whereas in off-policy optimisation $\mathcal{D}$ represents a replay buffer (Lin, 1992) which is continually filled with experience.

4. Evaluation

We evaluate on- and off-policy DeRL using a variety of learning environments, RL algorithms and intrinsic rewards. In particular, we investigate the following two hypotheses: (1) DeRL leads to **higher returns** or demonstrates

to **improved sample-efficiency**, converging to comparable returns than intrinsically motivated RL baselines in fewer environment interactions, and (2) DeRL is **more robust** than intrinsically motivated RL baselines to varying scale λ of the intrinsic rewards.

4.1. Intrinsic Rewards

We apply a series of intrinsic rewards to evaluate intrinsically motivated RL and compare such exploration techniques to our DeRL algorithms.

Count-based: We consider two count-based intrinsic rewards. **Dict-Count** directly stores and increments state occurrences in a table. **Hash-Count** additionally groups states based on a locality-sensitive hash function (Tang et al., 2017). For both approaches, the intrinsic reward is defined as given in Equation (1).

Prediction-based: Besides count-based intrinsic exploration definitions, we consider **ICM** (Pathak et al., 2017), **RIDE** (Raileanu & Rocktäschel, 2020) and **RND** (Burda et al., 2018b) as prediction-based approaches. ICM and RIDE both use the same inverse dynamics optimisation to compute state-representations and define intrinsic rewards based on next state prediction error and consecutive state differences, respectively. In contrast, RND initialises a random state representation and trains a network to mimic such representation. For more details, see Section 2.

4.2. Algorithms

Baselines: As baselines, we consider on-policy RL algorithms Advantage Actor-Critic (A2C) (Mnih et al., 2016) and Proximal Policy Optimisation (PPO) (Schulman et al., 2017). Both algorithms are trained using the combined reward $r_t = r_t^e + \lambda r_t^i$ with some weighting factor λ and intrinsic reward definition as given above.

DeRL: For our decoupled RL optimisation, we consistently train π_β using A2C as we found it to be more robust than PPO. We train π_β using Dict-Count and RND for intrinsic rewards in DeepSea tasks and use Dict-Count and ICM as intrinsic rewards in Hallway tasks. These intrinsic reward definitions were chosen as they achieved the highest returns for baseline algorithms in these respective environments. For the optimisation of π_e, we consider A2C and PPO for on-policy DeRL and Deep Q-Networks (DQN) (Mnih et al., 2015) for off-policy DeRL. We refer to these algorithms as DeA2C, DePPO and DeDQN.

4.3. Environments

All baselines and DeRL algorithms are evaluated in learning tasks defined within two environments.

DeepSea is an environment proposed as part of the Behaviour Suite (Bsuite) for RL (Osband et al., 2020), visualised in Figure 1. The environment targets the challenge of exploration and represents a $N \times N$ environment where the agent starts in the top left and has to reach a goal in the bottom right location. At each timestep, the agent moves one row down and can choose to either move left or right. For each state, both actions are randomly assigned to left and right movement. The agent observes the current location as a 2D one-hot encoding and receives a small negative reward for action right of $\frac{-0.01}{N}$ and 0 reward for moving left. Additionally, the agent receives a reward of $+1$ for reaching the goal and the episode ends after N timesteps. The difficulty of the exploration in DeepSea can be adjusted using N: the larger N, the harder it becomes for the agent to reach the goal location for optimal returns of 0.99. We evaluate all algorithms in the DeepSea task for $N \in \{10, 14, 20, 24, 30\}$.

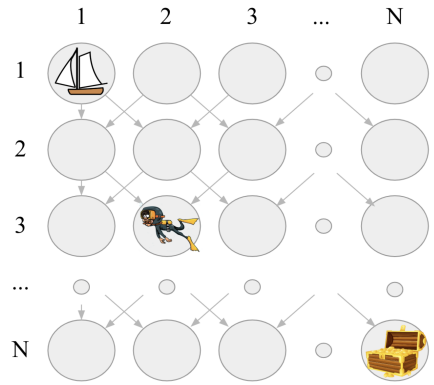


Figure 1: DeepSea environment, from Osband et al. (2020).

Figure 2: Hallway environment.

Hallway is a new environment proposed as part of this work to target exploration, visualised in Figure 2. In this navigation task, similar to DeepSea, an agent is located in a hallway starting at the left. A goal is located in the hallway and can be reached by moving N_l cells to the right. In contrast to DeepSea, the goal is not necessarily located at the right end of the hallway, but there might be further N_r empty cells to the right of the goal location. At each timestep, the agent can choose between three actions: move left, stay or move right. The agent receives a small negative reward of -0.01 for moving right or stay and 0 reward for moving left. Additionally, the episode ends after $2N_l$ steps and the agent receives large positive reward of $+1$ for reaching the goal for the first time and every time it stays

at the goal location for 10 steps. Therefore, the optimal behaviour is to move to the goal location and stay there for the remaining timesteps of the episode to collect further reward for staying at the goal. We hypothesise that Hallway tasks, in particular for $N_r > 0$, exploration through intrinsic rewards have to be carefully balanced because exploration incentives and extrinsic rewards in Hallway are not aligned. In contrast, receiving extrinsic reward in DeepSea requires the agent to reach states at the end of the environment, so intrinsic rewards for exploration strongly align with extrinsic rewards from the environment. We evaluate all algorithms in the Hallway environment with $N_l \in \{10, 20, 30\}$ and N_r either being 0 or equal to N_l.

4.4. Implementation details

Optimisation of A2C, PPO and DQN uses n-step returns for reduced bias of value estimates. On-policy interaction with the environment throughout all algorithms is completed using four parallel, synchronous environments (Mnih et al., 2016). On-policy algorithms are trained using batches of all four environments. Optimisation of on-policy baselines and the exploration policy for decoupled optimisation π_β uses an entropy regularisation term in the policy loss (Mnih et al., 2016).

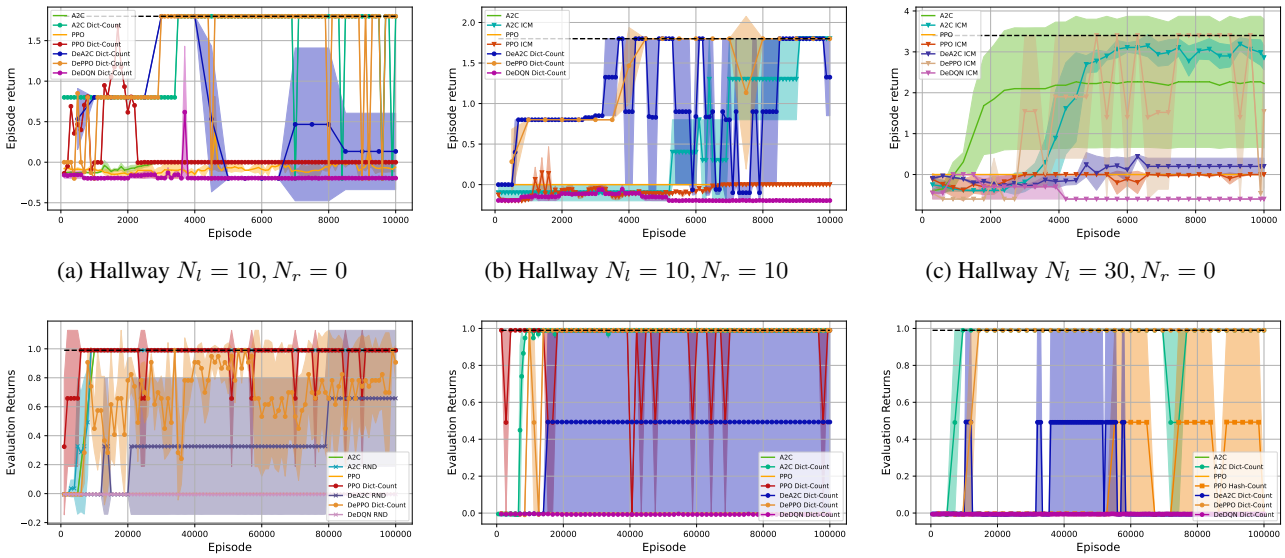
We train all approaches for 10,000 episodes on Hallway tasks and 100,000 episodes on DeepSea tasks. All algorithms are evaluated 100 times throughout training by taking their current parameters and applying their greedy policy on the task for 8 episodes. Reported results are averaged three random seeds with shading indicating a single standard deviation. Optimal returns are indicated using a dashed horizontal line. All baselines and DeRL algorithms use a weighting factor of $\lambda = 1$ for the combined reward signal unless stated otherwise.

5. Results

For all result plots, we show both A2C and PPO without any intrinsic rewards as well as with the single intrinsic reward which achieves the highest return with the algorithm for that task. Similarly, we visualise returns for all DeRL algorithms with the intrinsic rewards leading to highest returns for that task. For plots showing evaluation returns with all combinations of RL, DeRL algorithms and intrinsic rewards, see Appendix A.

Promising results: In multiple Hallway tasks, the exploitation policy of DeA2C and DePPO converges to the optimal policy, similar to our baselines, but is able to reach optimal returns earlier during training compared to A2C and PPO trained with intrinsic rewards. This can be observed for smaller Hallway tasks, in particular for $N_l = 10, N_r = 10$, visualised in Figure 3b. Similar results can be observed in DeepSea tasks, where DePPO achieves evaluation returns and sample efficiency comparable to the best baselines (see Figure 3e and Figure 3f). Generally, we find DePPO to exhibit more robust evaluation returns with less variance and fewer drops in returns compared to DeA2C and DeDQN across all Hallway tasks and several Hallway tasks. We believe training the exploitation policy π_e using PPO is more robust compared to A2C when trained with off-policy data from π_β due to the trust region optimisation applied by PPO (Schulman et al., 2015; 2017). This confines the policy change through a single optimisation by constraining divergence of behaviour and trained policy. In our case, DePPO uses a clipping approximation of the KL-divergence between π_β and π_e to enforce both policies to not diverge too far from each other. Such an optimisation constraint seems to assist training of π_e by stabilising the divergence of both policies.

Limitations and challenges: However, we also observe several remaining limitations and cases of failure for DeRL algorithms. DeDQN achieves significantly lower returns than both on-policy DeA2C and DePPO and exhibits no learning in most DeepSea (see Figure 3 bottom row and Figure 8a in Appendix A) and Hallway (see Figure 3 top row and Figure 6 in Appendix A) tasks despite its off-policy training. We believe distribution shift (Fujimoto et al., 2018) is causing instabilities when optimising the exploitation policy from data generated by π_β. While the same challenge is present in DeA2C and DePPO, IS weight correction appears to resolve some of these issues. We also observe that DeRL is less effective than baselines with and without intrinsic rewards in larger Hallway tasks with longer episode horizon. A2C without intrinsic rewards is able to solve some of these tasks with less variance and samples (see Figure 3c and Figures 6e and 6f in Appendix A). In these tasks, more reward can be accumulated because the agent can receive reward for staying at the goal location multiple times in contrast to tasks with $N_l = 10$. This appears to simplify exploration and intrinsic rewards mostly introduce noise to the optimisation and disrupt training. We believe that in these instances, exploration and exploitation policies often diverge which results in exploding or vanishing IS weights. Figure 4 visualises such exploding and vanishing IS weights for a single DeA2C run in the Hallway task with $N_l = N_r = 10$. These appear to correlate with some of the noticeable, frequent drops in returns throughout training. Even when applying Retrace(λ) (Munos et al., 2016) to clip IS weights, similar results could be observed. For larger DeepSea and Hallway tasks with longer episode horizons, DePPO exhibits similar instability as in harder Hallway tasks with significant variance and drops in performance. Generally, evaluation returns of DeRL are unstable in most tasks, showing significant drops in returns. While evaluation of baselines show similar drops, they appear less frequent.



(a) Hallway $N_l = 10, N_r = 0$ (b) Hallway $N_l = 10, N_r = 10$ (c) Hallway $N_l = 30, N_r = 0$

(d) DeepSea $N = 10$ (e) DeepSea $N = 14$ (f) DeepSea $N = 24$

Figure 3: Evaluation returns for A2C and PPO with and without intrinsic rewards and DeRL in three Hallway (top row) and DeepSea (bottom row) tasks. Shading indicates a single standard deviation.

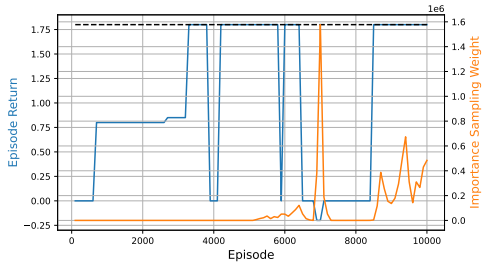


Figure 4: Evaluation returns and IS weights for DeA2C Dict-Count in Hallway ($N_l = N_r = 10$).

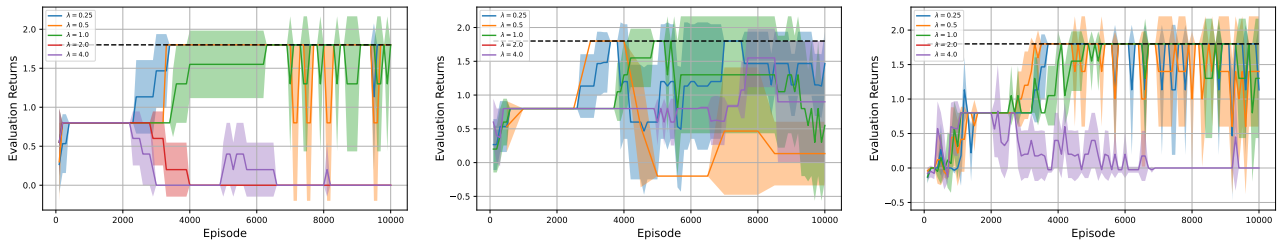
5.1. Varying intrinsic reward scale

In order to investigate the impact of DeRL on the robustness to varying scale of intrinsic rewards, we train all baselines and decoupled algorithms on the combined objective $r = r^e + \lambda r^i$ in two Hallway tasks. However, instead of using $\lambda = 1$ as in prior experiments, we evaluate for $\lambda \in \{0.25, 0.5, 1, 2, 4\}$ to analyse the robustness to varying scale of intrinsic reward rewards. Figure 5 shows evaluation returns for all values of λ for A2C, DeA2C and DePPO with the Dict-Count intrinsic reward in the Hallway task with $N_l = 10, N_r = 0$. For similar values of λ comparable learning curves can be observed for A2C and DePPO with larger weights $\lambda = 2$ and $\lambda = 4$ leading to overly dominating exploration and low evaluation returns. In contrast smaller weighting factors lead to convergence to the optimal performance, but such convergence is more stable for A2C than demonstrated by DePPO. DeA2C shows a very different picture with most scaling factors showing similar results which might indicate less sensitivity. However, evaluation returns for DeA2C exhibit high variance and do not allow conclusive analysis. While these experiments do not demonstrate significant differences in the sensitivity of DeRL to λ compared to typical RL with intrinsic rewards, these learning curves clearly indicate the importance of careful scaling. Even small deviations in λ can make the difference between learning the optimal policy and not learning at all. Further plots can be found in Appendix B.

6. Future Work

This work serves as an initial investigation into the possibility and effectiveness of decoupling exploration and exploitation for RL. However, there are three significant additions which we identified but did not address yet and require further investigation.

Analysing varying intrinsic reward decay: We already identified that the scale of intrinsic rewards has a significant impact on typical intrinsically motivated RL and DeRL training. We hypothesise that the speed of their decay has a similar impact on optimisation. If intrinsic rewards decay too quickly, exploration might be insufficient for RL optimisation to discover the optimal behaviour. If they decay too slowly, their non-stationary reward shaping might cause significant instabilities during the RL optimisation, similar to intrinsic rewards of large scale. Therefore, we propose



(a) A2C Dict-Count (b) DeA2C Dict-Count (c) DePPO Dict-Count

Figure 5: Evaluation returns of A2C, DeA2C and DePPO with Dict-Count intrinsic reward in Hallway $N_l = 10, N_r = 0$ for varying intrinsic reward weights λ. Shading indicates a single standard deviation.

to analyse the impact of varying speeds of decay for intrinsic rewards. For any intrinsic reward definition based on deep learning, the speed of decay can be varied through the learning rate of the intrinsic reward optimisation. For count-based intrinsic rewards, an additional factor can be added to the denominator of the intrinsic reward definition, shown in Equation (1), to change the speed of decay with increasing state visitation counts.

Minimising Distribution Shift: Training π_e using on-policy RL with IS correction still seems to suffer from significant variability. Based on our experiments with exploding IS weights (see Figure 4) and the improvement of DePPO over DeA2C, we believe that distribution shift of π_β and π_e is a cause for such variability. Therefore, we believe that the application of divergence constraints, proposed in the literature of offline RL (Levine et al., 2020), might be a valuable technique to minimise distribution shift. These auxiliary objectives are introduced to the optimisation and would enforce π_e and π_β to not diverge significantly by introducing a negative term $-\alpha d\left(\pi_e(\cdot|S_t), \pi_\beta(\cdot|S_t)\right)$ to the optimisation loss. This term is based on some some distance measure d between the distributions of policies π_β and π_e and some weighting hyperparameter α. Prominent distance measures are the maximum mean discrepancy (MMD) (Gretton et al., 2012), computed based on drawn samples from both policies, and the Kullback-Leibler divergence. Latter is commonly denoted KL-divergence and has been applied in offline RL (Jaques et al., 2019). Its regulariser can be written as

$$d_{KL}(\pi_e(\cdot|S_t), \pi_\beta(\cdot|S_t)) = \mathbb{E}_{A\sim\pi_e(\cdot|S)}\left[\log \pi_e(A|S_t) - \log \pi_\beta(A|S_t)\right] \quad (15)$$

Such divergence constraints have significant parallels to trust region optimisation (Schulman et al., 2015; 2017) as applied in DePPO for the optimisation of π_e and outlined in Section 5 which computes an approximation of the KL-divergence of π_β and π_e.

In DeRL, divergence constraints can be directly applied to the optimisation of either the exploration π_β or exploitation policy π_e. We can choose to keep π_β close to π_e and likewise can enforce π_e to stay close to π_β. We hypothesise, that adding such a constraining regularisation to the optimisation of π_β to enforce the exploration policy to stay close to the exploitation policy might be ideal as it does not interfere with the exploitation policy itself while still leveraging improved exploration of π_β. However, both directions as well as a combination of both constraints should be considered.

Only intrinsically rewarded exploration policy: The exploration policy π_β is solely trained to generate training data D for the optimisation of π_e. We hypothesise that it might be beneficial to optimise π_β using only intrinsic rewards. Such optimisation might cause further divergence of π_β and π_e but might be stable with additional divergence constrained, as outlined above.

7. Conclusion

In this work, we propose Decoupled RL (DeRL) which separates exploration and exploitation into two separate policies. The exploration policy is optimised with intrinsic rewards to incentivise exploration and the exploitation policy is trained using only extrinsic rewards on data generated experience of the exploration policy. We believe such decoupling is primarily beneficial when tuning the exploration is challenging or heavy exploration is required which would typically dominate any extrinsic rewards received by the environment. On-policy and off-policy versions of DeRL are formulated and evaluated in two sparse-reward environments. We observe that on-policy DeRL demonstrates improved sample efficiency in some tasks by reaching the optimal policy with fewer interactions with the environment. However, DeRL still seems to suffer from frequent drops in performance which correlate with divergence of the exploration and exploitation policies. We find that trust region optimisation of DePPO appears to benefit stability, but further research is proposed to address remaining challenges.

References

Andoni, A. and Indyk, P. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. *Communications of the ACM*, 51(1):117, 2008.

Barto, A. G. Intrinsic motivation and reinforcement learning. In *Intrinsically motivated learning in natural and artificial systems*, pp. 17–47. Springer, 2013.

Bellemare, M., Srinivasan, S., Ostrovski, G., Schaul, T., Saxton, D., and Munos, R. Unifying count-based exploration and intrinsic motivation. In *Advances in Neural Information Processing Systems*, pp. 1471–1479, 2016.

Bellemare, M. G., Naddaf, Y., Veness, J., and Bowling, M. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47(1):253–279, 2013.

Burda, Y., Edwards, H., Pathak, D., Storkey, A., Darrell, T., and Efros, A. A. Large-scale study of curiosity-driven learning. *arXiv preprint arXiv:1808.04355*, 2018a.

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

Charikar, M. S. Similarity estimation techniques from rounding algorithms. In *ACM symposium on Theory of computing*, pp. 380–388. ACM, 2002.

Chentanez, N., Barto, A. G., and Singh, S. P. Intrinsically motivated reinforcement learning. In *Advances in neural information processing systems*, pp. 1281–1288, 2005.

Degris, T., White, M., and Sutton, R. S. Off-policy actor-critic. In *International Conference on Machine Learning*, 2012.

Espeholt, L., Soyer, H., Munos, R., Simonyan, K., Mnih, V., Ward, T., Doron, Y., Firoiu, V., Harley, T., Dunning, I., et al. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. In *International Conference on Machine Learning*, pp. 1407–1416. PMLR, 2018.

Fujimoto, S., Van Hoof, H., and Meger, D. Addressing function approximation error in actor-critic methods. *arXiv preprint arXiv:1802.09477*, 2018.

Gretton, A., Borgwardt, K. M., Rasch, M. J., Schölkopf, B., and Smola, A. A kernel two-sample test. *The Journal of Machine Learning Research*, 13(1):723–773, 2012.

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning*, pp. 1861–1870. PMLR, 2018.

Jaques, N., Ghandeharioun, A., Shen, J. H., Ferguson, C., Lapedriza, A., Jones, N., Gu, S., and Picard, R. Way off-policy batch deep reinforcement learning of implicit human preferences in dialog. *arXiv preprint arXiv:1907.00456*, 2019.

Levine, S., Kumar, A., Tucker, G., and Fu, J. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

Lin, L.-J. *Reinforcement learning for robots using neural networks*. Carnegie Mellon University, 1992.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., and Hassabis, D. Human-level control through deep reinforcement learning. *Nature*, 518(7540): 529–533, 2015.

Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., and Kavukcuoglu, K. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pp. 1928–1937, 2016.

Munos, R., Stepleton, T., Harutyunyan, A., and Bellemare, M. G. Safe and efficient off-policy reinforcement learning. In *Advances in Neural Information Processing Systems*, 2016.

Osband, I., Doron, Y., Hessel, M., Aslanides, J., Sezener, E., Saraiva, A., McKinney, K., Lattimore, T., Szepesvari, C., Singh, S., et al. Behaviour suite for reinforcement learning. In *International Conference on Learning Representations*, 2020.

Ostrovski, G., Bellemare, M. G., van den Oord, A., and Munos, R. Count-based exploration with neural density models. In *International Conference on Machine Learning*, pp. 2721–2730. JMLR. org, 2017.

Oudeyer, P.-Y. and Kaplan, F. What is intrinsic motivation? a typology of computational approaches. *Frontiers in neurorobotics*, 1:6, 2009.

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

Oudeyer, P.-Y., Kaplan, F., et al. How can we define intrinsic motivation. In *Conference on Epigenetic Robotics*, volume 5, pp. 29–31, 2008.

Pathak, D., Agrawal, P., Efros, A. A., and Darrell, T. Curiosity-driven exploration by self-supervised prediction. In *International Conference on Machine Learning*, volume 2017, 2017.

Raileanu, R. and Rocktäschel, T. Ride: Rewarding impact-driven exploration for procedurally-generated environments. In *International Conference on Learning Representations*, 2020.

Rashid, T., Peng, B., Böhmer, W., and Whiteson, S. Optimistic exploration even with a pessimistic initialisation. In *International Conference on Learning Representations*, 2020.

Schmidhuber, J. Curious model-building control systems. In *Neural Networks, 1991. 1991 IEEE International Joint Conference on*, pp. 1458–1463. IEEE, 1991.

Schulman, J., Levine, S., Abbeel, P., Jordan, M., and Moritz, P. Trust region policy optimization. In *International conference on machine learning*, pp. 1889–1897. PMLR, 2015.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Silver, D., Lever, G., Heess, N., Degris, T., Wierstra, D., and Riedmiller, M. Deterministic policy gradient algorithms. In *International conference on machine learning*, pp. 387–395. PMLR, 2014.

Strehl, A. L. and Littman, M. L. An analysis of model-based interval estimation for markov decision processes. *Journal of Computer and System Sciences*, 74(8):1309–1331, 2008.

Taïga, A. A., Fedus, W., Machado, M. C., Courville, A., and Bellemare, M. G. Benchmarking bonus-based exploration methods on the arcade learning environment. *arXiv preprint arXiv:1908.02388*, 2019.

Tang, H., Houthooft, R., Foote, D., Stooke, A., Chen, O. X., Duan, Y., Schulman, J., DeTurck, F., and Abbeel, P. # exploration: A study of count-based exploration for deep reinforcement learning. In *Advances in neural information processing systems*, pp. 2753–2762, 2017.

van Hasselt, H., Guez, A., and Silver, D. Deep reinforcement learning with double q-learning. In *AAAI Conference on Artificial Intelligence*, AAAI'16, pp. 2094–2100. AAAI Press, 2016.

Watkins, C. J. C. H. *Learning from delayed rewards*. PhD thesis, King's College, Cambridge, 1989.

Whitney, W. F., Bloesch, M., Springenberg, J. T., Abdolmaleki, A., and Riedmiller, M. Rethinking exploration for sample-efficient policy learning. *arXiv preprint arXiv:2101.09458*, 2021.

A. Evaluation results

Figure 6 shows evaluation returns for A2C and PPO with and without the intrinsic rewards which have been found to reach highest returns and DeRL with its most effective intrinsic reward. Results are averaged across three random seeds and shading indicates a single standard deviation of returns.

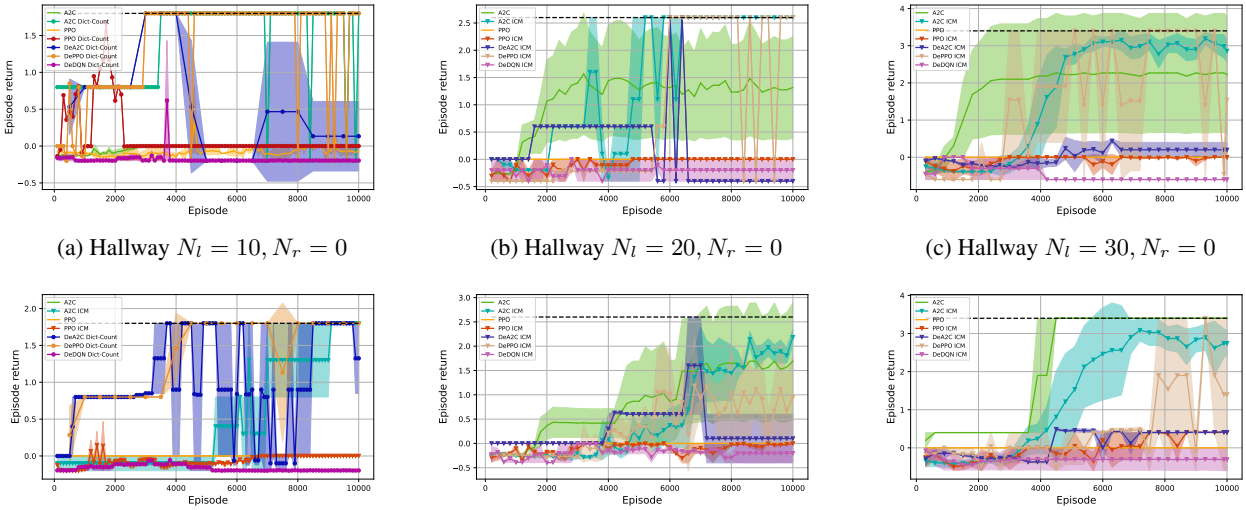
Following this overview over the best algorithms in each Hallway task, Figure 7 shows evaluation returns for all evaluated approaches including the combination of A2C and PPO with baselines with all five intrinsic reward definitions. Also, all DeRL evaluation results are visualised with Dict-Count and ICM intrinsic reward definitions.

Similarly, Figure 8 shows evaluation returns for all baselines, DeRL algorithms in each DeepSea task. A2C and PPO baselines were evaluated without intrinsic rewards and all five intrinsic rewards. DeRL algorithms were evaluated with Dict-Count and RND intrinsic rewards. Again, results are averaged across three random seeds and shading indicates a single standard deviation of returns.

B. Varying Intrinsic Reward Scale

Figure 9 and Figure 11 show the sensitivity of A2C, DeA2C, DePPO and DeDQN with respect to scaling factor λ of intrinsic Dict-Count and ICM rewards for the Hallway $(N_l = 10, N_r = 0)$ and Hallway $(N_l = 10, N_r = 10)$ tasks, respectively.

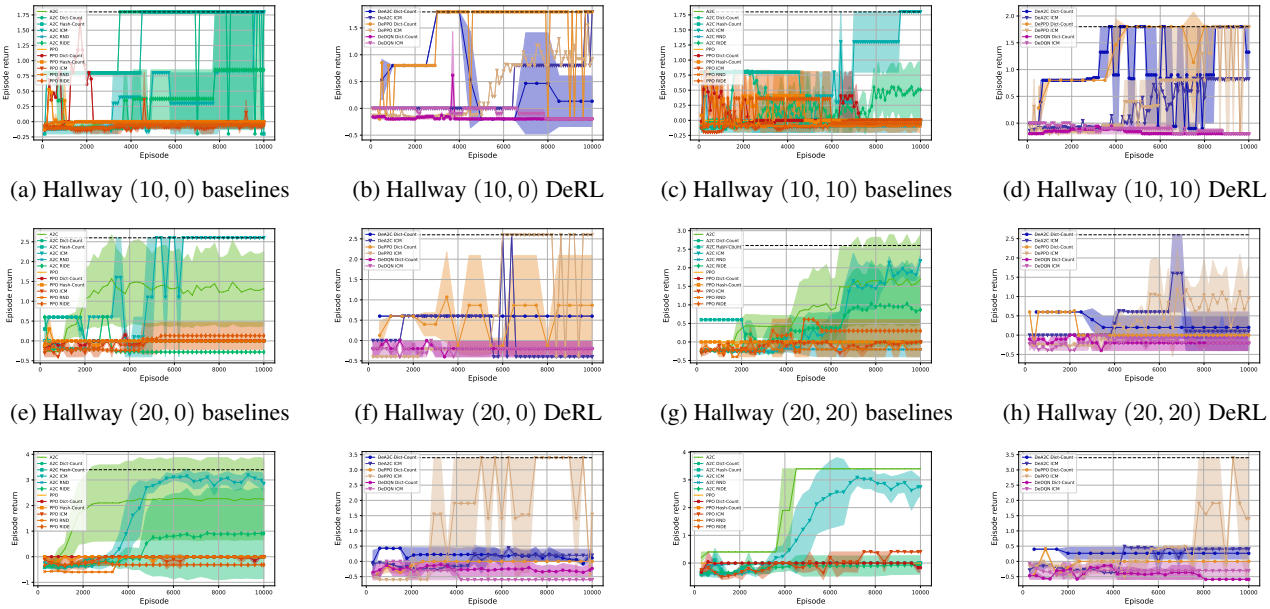
Sensitivity to λ for the PPO baseline with Dict-Count and ICM, as well as the sensitivity to λ for all remaining intrinsic reward baselines can be found in Figure 10 and Figure 12.



(a) Hallway $N_l = 10, N_r = 0$ (b) Hallway $N_l = 20, N_r = 0$ (c) Hallway $N_l = 30, N_r = 0$

(d) Hallway $N_l = 10, N_r = 10$ (e) Hallway $N_l = 20, N_r = 20$ (f) Hallway $N_l = 30, N_r = 30$

Figure 6: Hallway returns for A2C and PPO with and without best intrinsic rewards and DeRL. Shading indicates a single standard deviation.

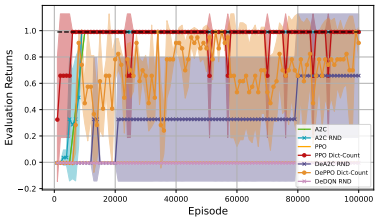


(a) Hallway $(10, 0)$ baselines (b) Hallway $(10, 0)$ DeRL (c) Hallway $(10, 10)$ baselines (d) Hallway $(10, 10)$ DeRL

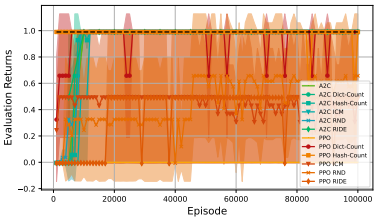
(e) Hallway $(20, 0)$ baselines (f) Hallway $(20, 0)$ DeRL (g) Hallway $(20, 20)$ baselines (h) Hallway $(20, 20)$ DeRL

(i) Hallway $(30, 0)$ baselines (j) Hallway $(30, 0)$ DeRL (k) Hallway $(30, 30)$ baselines (l) Hallway $(30, 30)$ DeRL

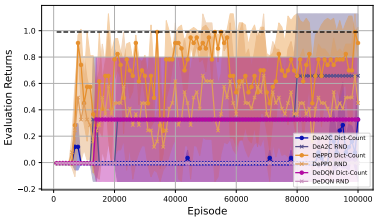
Figure 7: Hallway evaluation returns for all baselines and DeRL algorithms.



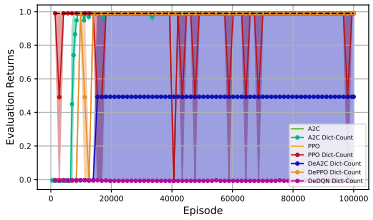
(a) DeepSea $N = 10$ best



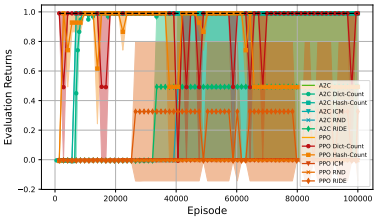
(b) DeepSea $N = 10$ baselines



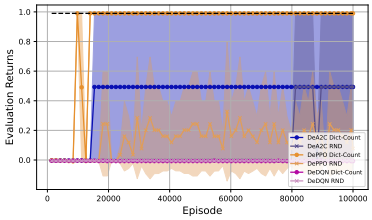
(c) DeepSea $N = 10$ DeRL



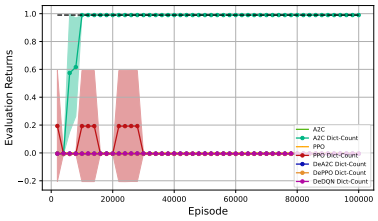
(d) DeepSea $N = 14$ best



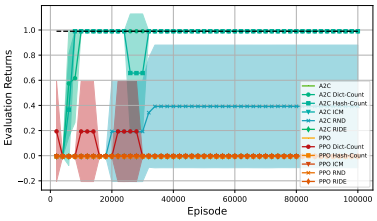
(e) DeepSea $N = 14$ baselines



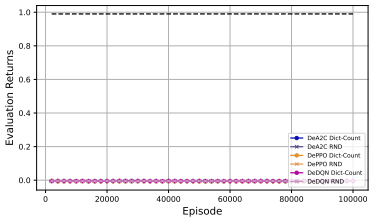
(f) DeepSea $N = 14$ DeRL



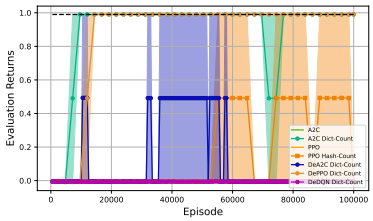
(g) DeepSea $N = 20$ best



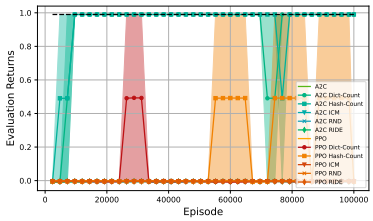
(h) DeepSea $N = 20$ baselines



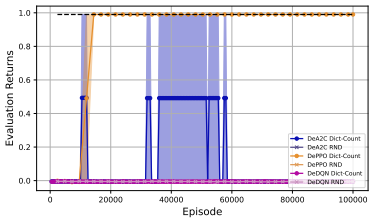
(i) DeepSea $N = 20$ DeRL



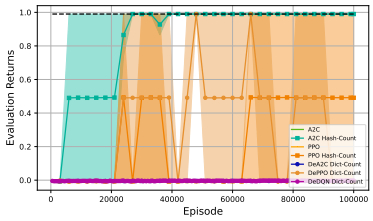
(j) DeepSea $N = 24$ best



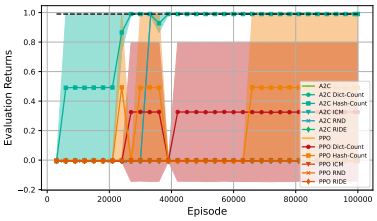
(k) DeepSea $N = 24$ baselines



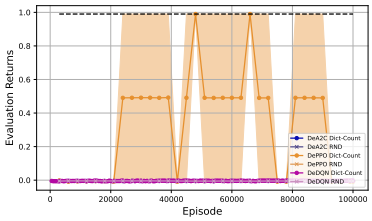
(l) DeepSea $N = 24$ DeRL



(m) DeepSea $N = 30$ best

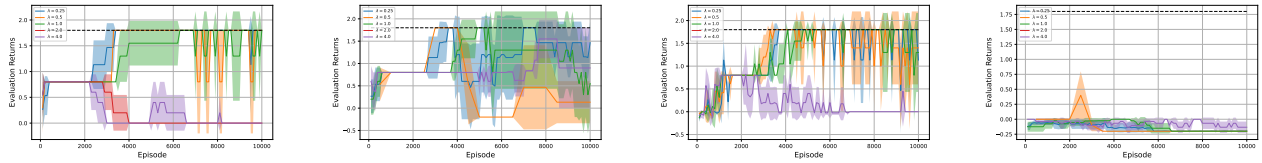


(n) DeepSea $N = 30$ baselines

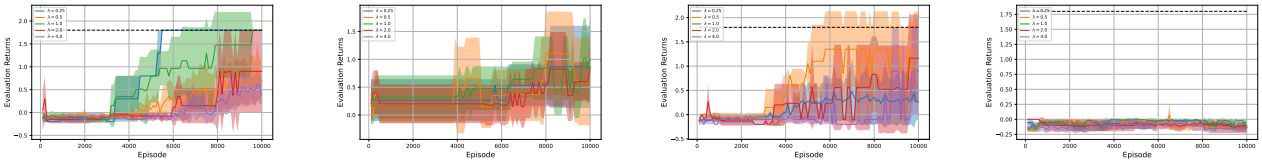


(o) DeepSea $N = 30$ DeRL

Figure 8: DeepSea evaluation returns for A2C and PPO with and without best intrinsic rewards and DeRL (first column), all baselines (second column) and all DeRL algorithms (third column) for all DeepSea tasks.

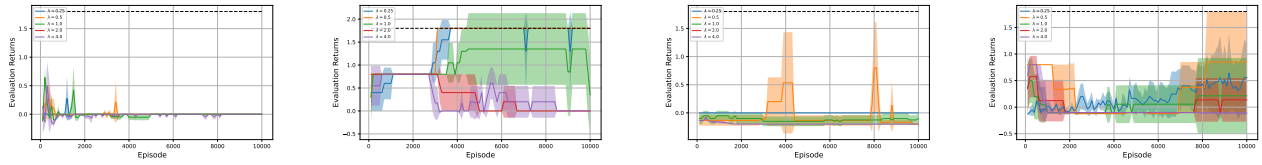


(a) A2C Dict-Count (b) DeA2C Dict-Count (c) DePPO Dict-Count (d) DeDQN Dict-Count

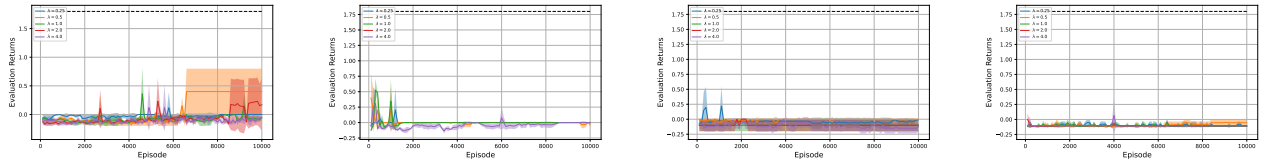


(e) A2C ICM (f) DeA2C ICM (g) DePPO ICM (h) DeDQN ICM

Figure 9: Hallway ($N_l = 10, N_r = 0$) evaluation returns of A2C, DeA2C, DePPO and DeDQN with Dict-Count and ICM intrinsic rewards for varying $\lambda \in \{0.25, 0.5, 1.0, 2.0, 4.0\}$

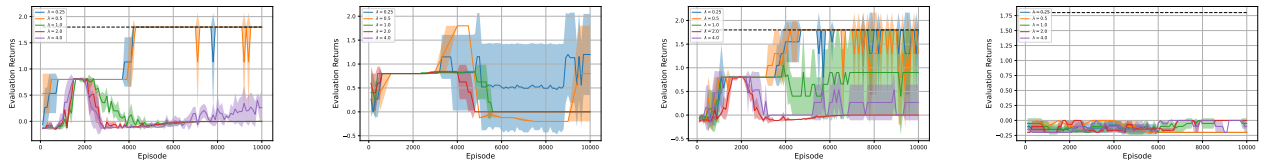


(a) PPO Dict-Count (b) A2C Hash-Count (c) A2C RND (d) A2C RIDE

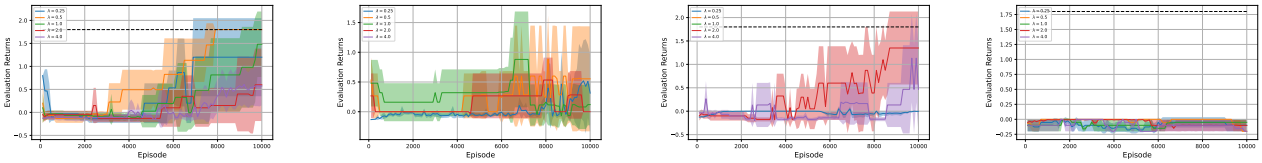


(e) PPO ICM (f) PPO Hash-Count (g) PPO RND (h) PPO RIDE

Figure 10: Hallway ($N_l = 10, N_r = 0$) evaluation returns of PPO with Dict-Count and ICM as well as A2C and PPO with remaining rewards Hash-Count, RND and RIDE for varying $\lambda \in \{0.25, 0.5, 1.0, 2.0, 4.0\}$

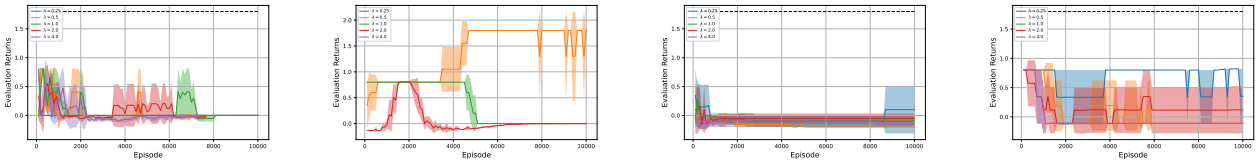


(a) A2C Dict-Count (b) DeA2C Dict-Count (c) DePPO Dict-Count (d) DeDQN Dict-Count

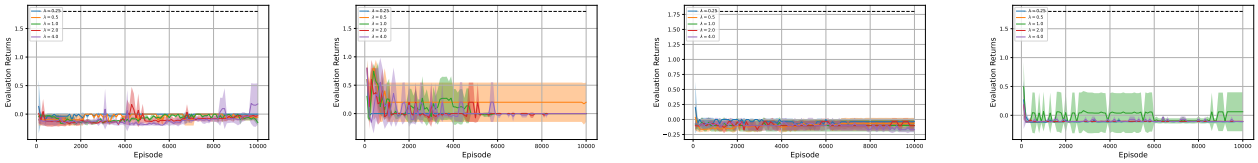


(e) A2C ICM (f) DeA2C ICM (g) DePPO ICM (h) DeDQN ICM

Figure 11: Hallway ($N_l = 10, N_r = 10$) evaluation returns of A2C, DeA2C, DePPO and DeDQN with Dict-Count and ICM intrinsic rewards for varying $\lambda \in \{0.25, 0.5, 1.0, 2.0, 4.0\}$



(a) PPO Dict-Count (b) A2C Hash-Count (c) A2C RND (d) A2C RIDE



(e) PPO ICM (f) PPO Hash-Count (g) PPO RND (h) PPO RIDE

Figure 12: Hallway ($N_l = 10, N_r = 10$) evaluation returns of PPO with Dict-Count and ICM as well as A2C and PPO with remaining rewards Hash-Count, RND and RIDE for varying $\lambda \in \{0.25, 0.5, 1.0, 2.0, 4.0\}$