Adaptive Intrinsic Motivation with Decision Awareness

Anonymous Authors[1]

Abstract

Intrinsic motivation is a simple but powerful method to encourage exploration, which is one of the fundamental challenges of reinforcement learning. However, we demonstrate that widely used intrinsic motivation methods are highly dependent on the ratio between the extrinsic and intrinsic rewards through extensive experiments on sparse reward MiniGrid tasks. To overcome the problem, we propose an intrinsic reward coefficient adaptation scheme that is equipped with intrinsic motivation awareness and adjusts the intrinsic reward coefficient online to maximize the extrinsic return. We demonstrate that our method, named Adaptive Intrinsic Motivation with Decision Awareness (AIMDA), operates stably in various challenging MiniGrid environments without algorithm-task-specific hyperparameter tuning.

1. Introduction

Considering that a reinforcement learning (RL) agent is trained using the data generated by itself, the importance of exploration in RL cannot be overemphasized. Exploration in RL has been studied extensively in various ways, by injecting noise (Lillicrap et al., 2016; Fortunato et al., 2018), by rewarding diversity (Eysenbach et al., 2019) and information gain (Houthooft et al., 2016), by empowerment (Gregor et al., 2017), by maximizing entropy of actions (Haarnoja et al., 2018), and by providing subgoals and curriculum (Florensa et al., 2018; Nair et al., 2018).

In addition to the exploration methods described, intrinsic motivation (IM) methods are the most widely studied, simple but powerful and prominent methods to encourage exploration (Bellemare et al., 2016; Pathak et al., 2017; Burda et al., 2019b; Raileanu & Rocktäschel, 2020; Seo et al., 2021). They train agents with the total reward $r^{\text{tot}} = r^{\text{ext}} + \beta r^{\text{int}}$, that is the weighted sum of the extrinsic reward r^{ext} (the true reward from the environment) and the intrinsic reward r^{int} (the auxiliary reward from IM). Therefore, these methods are particularly effective for hard-exploration tasks where the extrinsic reward is sparse.

One common weakness of conventional IM methods is that they are very sensitive to the choice of intrinsic reward coefficient (IRC) β. If the IRC is set too large, the agent will not pursue the extrinsic reward and converge to an undesired policy that maximizes the intrinsic return. If the IRC is too small, the agent will not be encouraged enough to explore the environment, which may lead to a suboptimal policy. The scale of the intrinsic reward varies greatly depending on the tasks, IM methods, and learning progress. Therefore, the performance, measured in terms of the extrinsic return, depends heavily on the choice of the hyperparameter β more than the type of IM method generating the intrinsic reward. In addition, the optimal IRC can be changed throughout the course of training, i.e., a large IRC is preferred in the beginning of training and a small IRC is preferred later (Seo et al., 2021). Refer to Figure 1, where the optimal β varies across tasks and algorithms. An optimal IRC for one task and algorithm may fail completely on another task and algorithm. This phenomenon of IRC dictating the policy is undesirable since the intrinsic reward should only be an auxiliary reward to guide exploration. A decision-aware RL agent should stably maximize the extrinsic return without being dictated by the scale of the auxiliary return.

As a makeshift for this problem, recent works evaluate independent runs for a sufficiently large range of IRCs, e.g., $\beta \in \{0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1.0\}$, then report the best extrinsic return for each task (Raileanu & Rocktäschel, 2020; Seo et al., 2021; Zhang et al., 2021; Parisi et al., 2021). However, this evaluation process is highly resource-inefficient and it is unrealistic to afford many trials with different coefficients in the real application stage. The optimal value of β may be outside the search space or between two values due to the finite precision (Figure 3). Also, this evaluation makes it difficult to determine the comparative advantage and robustness of the different IM methods across different tasks.

To overcome the problem, we propose Adaptive Intrinsic Motivation with Decision Awareness (AIMDA), where we equip the agent with awareness of the intrinsic motivation

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

Work in Progress. Under review by the International Conference on Machine Learning Workshop. Do not distribute.

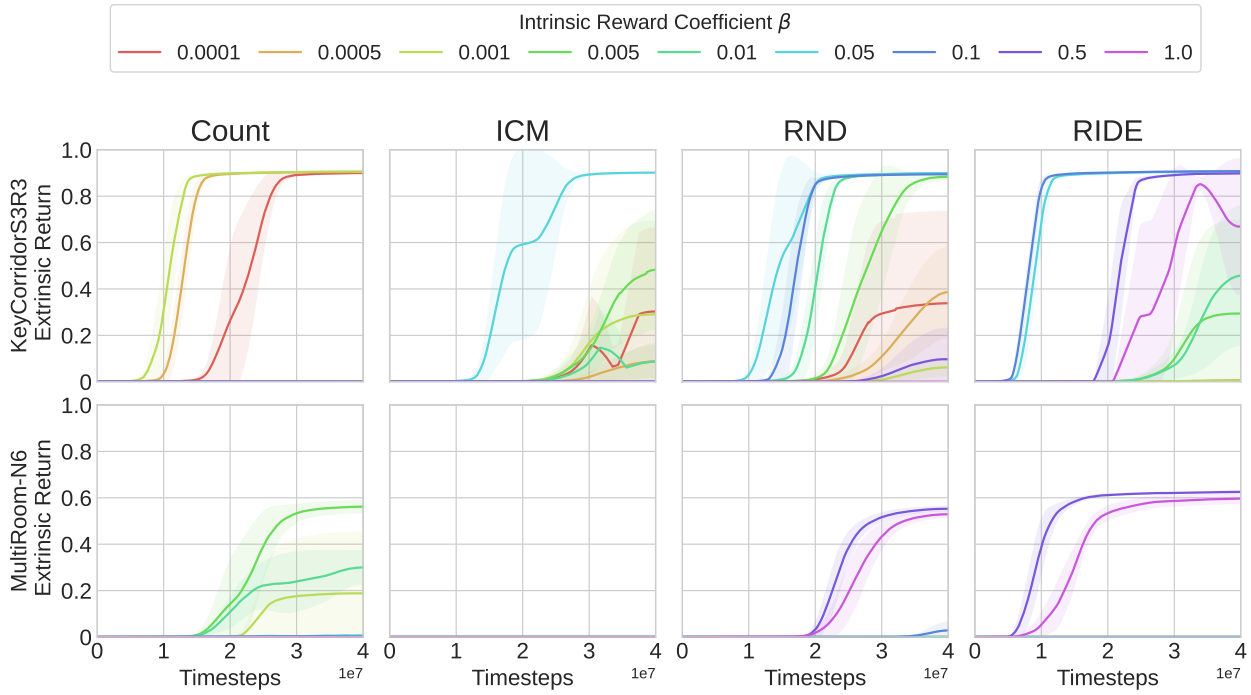

Figure 1. The extrinsic return of multiple IM methods (Count-based, ICM, RND, and RIDE) on two MiniGrid tasks (KeyCorridorS3R3 and MultiRoom-N6) with different IRCs: $\beta \in \{0.0001, 0.0005, 0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1.0\}$. The optimal IRC varies greatly depending on the task and algorithm. We report the mean and standard deviation over three random seeds. Refer to Appendix A.1 for implementation details of the experiments.

and allow the agent to adapt the scale of the intrinsic motivation itself to maximize the extrinsic return. We utilize the distributed RL framework (Mnih et al., 2016; Espeholt et al., 2018) to train multiple actors with different IRCs in parallel. We graft the decision-awareness onto the agent by appending the IRC to the state embedding. Then we use the concatenation as the input of value and policy networks. This intrinsic motivation awareness allows us to periodically replace the worst-performing coefficient with the mean of the two best performing coefficients, where the indicator of the performance is the extrinsic return.

In this work, we reveal the IRC dependency of IM methods with extensive experiments on various MiniGrid hard-exploration tasks (Chevalier-Boisvert et al., 2018), using widely-used IM methods (Bellemare et al., 2016; Pathak et al., 2017; Burda et al., 2019b; Raileanu & Rocktäschel, 2020; Seo et al., 2021) across a vast range of IRCs. Then we demonstrate the robustness and effectiveness of AIMDA that adaptively adjusts IRCs with decision awareness without additional training data, network, and algorithm-task-specific hyperparameter selection.

2. Background and Related Work

2.1. Intrinsic Motivation

Count-based exploration is one of the most well-known IM methods. They provide agents with high exploration bonus for states with lower visitation counts (Bellemare et al., 2016; Ostrovski et al., 2017; Martin et al., 2017; Tang et al., 2017; Parisi et al., 2021). For environments with high-dimensional states, where counting exact number of state visitation is infeasible, they use density models such as Context-Tree Switching (Bellemare et al., 2016) or PixelCNN (Ostrovski et al., 2017).

Another major family of IM methods is curiosity-driven IM, where the curiosity is normally defined as the error of the agent's prediction about the environment dynamics (Stadie et al., 2015). Intrinsic Curiosity Module (ICM) learns to predict the next state's latent embedding (Pathak et al., 2017; Burda et al., 2019a). RIDE uses episodic counts and the difference of two consecutive states' latent representation to create an exploration bonus (Raileanu & Rocktäschel, 2020).

Besides the aforementioned IM methods, many successful IM methods have been proposed recently. Random Network

Distillation (RND) uses prediction error between features of a randomly-initialized network and a learned network (Burda et al., 2019b). RE3 uses the k-nearest neighbor distance of a state in the feature space, where the feature space is obtained by passing the replay buffer through a randomly initialized encoder (Seo et al., 2021). Several studies have recently shown the success of using combinations of previously proposed IM methods. NovelD uses the difference of novelty measures of two consecutive states (Zhang et al., 2021). C-BET uses a combination of agent-centric and environment-centric visitation counts (Parisi et al., 2021).

2.2. Automated Reinforcement Learning

Although RL has shown great success in many domains, it is widely being accepted that most results rely on heavily tuned hyperparameters and the structure of the networks (Henderson et al., 2018; Engstrom et al., 2020; Andrychowicz et al., 2021). Following the success of Automated Machine Learning (AutoML) (Hutter et al., 2019), the field of Automated Reinforcement Learning (AutoRL) to automate design choices of RL has been receiving attention recently (Parker-Holder et al., 2022). AutoRL methods not only apply AutoML to RL naively, but they also handle the RL-specific problems, such as the non-stationarity of RL and diversity of environments. Many recent works on AutoRL have produced successful results by actively using the distributed RL framework (Mnih et al., 2016; Espeholt et al., 2018; Kapturowski et al., 2018; Badia et al., 2020; Mnih et al., 2020).

Blackbox online tuning methods adaptively select hyperparameters online. Some of them try to adapt the hyperparameter weights and schedules for temporal-difference methods (Sutton & Singh, 1994; Kearns & Singh, 2000; White & White, 2016; Paul et al., 2019). Some methods use bandits, where each arm is assigned to a set of hyperparameters such as the degree of stochasticity (Schaul et al., 2019), degree of exploration (Ball et al., 2020), and degree of optimism (Moskovitz et al., 2021). Pislar et al. (2022) uses a bandit to trigger a switch between exploration and exploitation modes. All these methods train the non-stationary bandit to maximize the extrinsic return.

Population-based methods train multiple agents in parallel to explore and exploit the hyperparameter space periodically. Most of them explore the hyperparameter space with random perturbation and exploit stronger parameters with higher performance (Jaderberg et al., 2017; Parker-Holder et al., 2020; Liu et al., 2019; Franke et al., 2021; Zhang et al., 2016). The strength of population-based methods is that they can learn the optimal schedule online with the same wall clock time as that of a vanilla baseline. However, they require a significant amount of memory and computation resources, because they train multiple independent networks in parallel.

Algorithm 1 AIMDA

Initialize IM type, number of actors N, IRC population $\boldsymbol{\beta} = \{\beta^{(1)}, \ldots, \beta^{(N)}\}$, total training steps t_{tot}, global timestep $t_{\text{global}} = 0$, adaptation period t_{adp}, replay buffer $\mathcal{B} = \mathbf{0}$, cumulative extrinsic return buffer $\mathcal{B}_{\text{ext}} = \text{zeros}(N)$
while $t_{\text{global}} < t_{\text{tot}}$ **do**
  //DECISION-AWARE EXPLORATION
  for actor $i = 1$ **to** N in **parallel do**
    $t \leftarrow 0$
    $\beta^{(i)} \leftarrow \boldsymbol{\beta}[i]$
    while $t < t_{\text{update}}$ **do**
      Collect a transition including IRC $\beta^{(i)}$
      $\tau_t^{(i)} = \left(o_t^{(i)}, a_t^{(i)}, o_{t+1}^{(i)}, \beta^{(i)}, r_t^{(i),\text{ext}} + \beta^{(i)} r_t^{(i),\text{int}}\right)$
      $\mathcal{B} \leftarrow \mathcal{B} \cup \{\tau_t^{(i)}\}$
      $\mathcal{B}_{\text{ext}}[i] \leftarrow \mathcal{B}_{\text{ext}}[i] + r_t^{(i),\text{ext}}$
      $t \leftarrow t + 1$
    end while
  end for
  $t_{\text{global}} \leftarrow t_{\text{global}} + N t_{\text{update}}$
  Update policy and value networks with transitions in $\mathcal{B}$
  Reset $\mathcal{B} \leftarrow \mathbf{0}$
  //ADAPTING IRCS VIA MIX-BEST-TWO
  if $t_{\text{global}} \% t_{\text{adp}} == 0$ **then**
    $i_{\text{min}} = \text{argmin}\, \mathcal{B}_{\text{ext}}$
    $i_{\text{max1}} = \text{argmax}\, \mathcal{B}_{\text{ext}}$, $i_{\text{max2}} = \text{arg2ndmax}\, \mathcal{B}_{\text{ext}}$
    $\boldsymbol{\beta}[i_{\text{min}}] \leftarrow (\boldsymbol{\beta}[i_{\text{max1}}] + \boldsymbol{\beta}[i_{\text{max2}}])/2$
    Reset $\mathcal{B}_{\text{ext}} \leftarrow \text{zeros}(N)$
  end if
end while

The closest work to ours is Agent57 (Mnih et al., 2020) that also trains multiple agents with a population of different IRCs. It trains a bandit to adaptively select the optimal IRC and discount factor. However, its weakness, as well as any other methods that use bandits (Schaul et al., 2019; Ball et al., 2020; Moskovitz et al., 2021), is that the scope of the search space is limited to the set of pre-defined arms. Agent57 uses 32 arms, each of which is assigned to the intrinsic reward coefficient of sigmoid function from 0 to 0.3. Therefore, it cannot adapt to situations requiring $\beta > 0.3$. It also has limited expressibility for β near 0.15 since most arms are populated near 0 and 0.3. Also due to ϵ-UCB exploration, Agent57 has to explore the same pre-defined set of IRCs until the end of training.

3. Method

The ultimate goal of our method, Adaptive Intrinsic Motivation with Decision Awareness (AIMDA), is to optimally

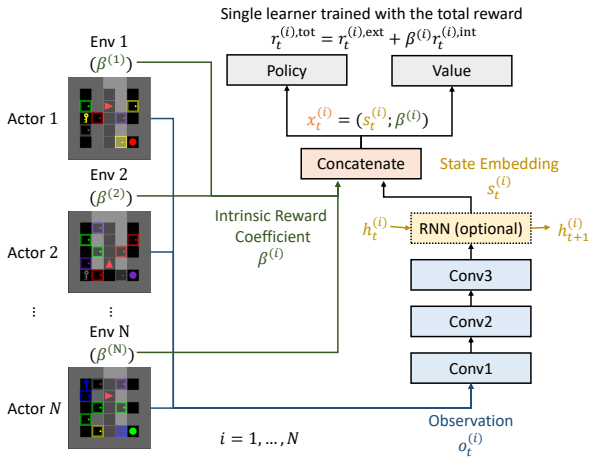

Figure 2. The decision awareness activated by the IRC input concatenated to the state embedding. Conventional IM methods use the same fixed IRCs for all actors throughout the course of training, therefore there is no need to input IRC.

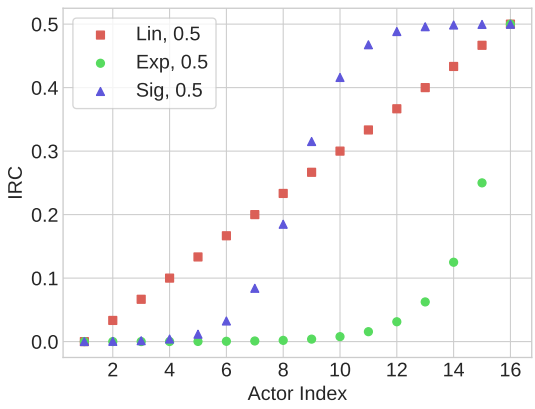

Figure 3. Examples of three types of Initialization schemes (linear, exponential and sigmoid) for the initial IRC population with $N = 16$ actors and $\beta_{\max} = 0.5$.

adapt the IRC online by training multiple actors in parallel with a population of various non-discretized IRCs. AIMDA is based on a multi-actor and single-learner framework (Mnih et al., 2016), where each actor i collects trajectories from its own environment with a distinct IRC, $\beta^{(i)}$. AIMDA allows all actors' IRCs to eventually converge to the IRC that attains the highest extrinsic return, without any additional network or training. We initialize the population of IRCs with various schemes (Section 3.3) then adapt the population to maximize the extrinsic return (Section 3.2), where the adaptation is enabled by the IRC awareness (Section 3.1).

3.1. Intrinsic Reward Coefficient Awareness

IRC awareness is a necessary prerequisite to enable online adaptation of the IRC because the behavior of the optimal policy depends on the total return (i.e., sum of the intrinsic and extrinsic returns) weighted by the IRC. We inject this awareness by appending $\beta^{(i)}$ to the state embedding, where the concatenation is the input of value and policy networks as in Figure 2. Given trajectories from multiple actors with different IRCs and the corresponding total rewards, a single learner can learn the optimal policy for various values of IRCs.

Because each actor is not bound to a fixed IRC, it is possible to freely change the assigned IRC online. While changing the IRCs freely, the computational burden for training does not increase much, because we do not allocate additional learners for each actor unlike Mnih et al. (2020). A single learner with the IRC input can distinguish trajectories with the input IRC.

3.2. Adaptation Scheme

We propose an online adaptation scheme of IRC to maximize the extrinsic return by utilizing the awareness introduced in Section 3.1. There are countless ways to adapt to the optimal values by reflecting the decision awareness (i.e., to maximize the extrinsic return). In the current form of this work, we demonstrate the *mix-best-two* scheme that replaces the worst IRC in terms of the accumulated extrinsic reward with the mean of the best two IRCs. Starting from the sufficiently large initial population of the IRCs, the IRCs can converge to a fixed point as in Figure 4a.

Note that the *mix-best-two* scheme allows the population of IRCs to converge to the algorithm-task-specific optimal IRC, being more exploitative than randomly selecting schemes (Badia et al., 2020). Also, the proposed scheme can express more diverse IRCs without requiring additional networks to be trained such as a non-stationary bandit (Mnih et al., 2020).

3.3. Initialization Scheme

An initialization scheme of the population of IRCs suitable for the *mix-best-two* scheme should be wide enough to possibly contain the optimal IRC. However, the decrease in precision caused by sizing the population too wide should also be avoided. The minimum value of the population can be set to 0 naturally (i.e., no intrinsic motivation), but the maximum value remains to be adjusted by the hyperparameter $\beta_{\max}$.

There is one more hyperparameter, which determines the distribution of the population. In this work we use the linear, exponential as well as the sigmoid initialization used for Agent57 (Mnih et al., 2020). Each initialization is formu-

lated for actor $i = 1, \ldots, N$ as follows.

1. Linear: $\beta^{(i)} = \beta_{\max} \frac{i-1}{N-1}$, focusing evenly from 0 to $\beta_{\max}$.
2. Exponential: $\beta^{(i)} = \beta_{\max} 2^{i-N}$, focusing near 0.
3. Sigmoid: $\beta^{(i)} = \beta_{\max} \left(1 + \exp\left(\frac{N}{2} \frac{N+1-2i}{N-1}\right)\right)^{-1}$, focusing near 0 and $\beta_{\max}$.

We denote each initialization in short as (distribution, $\beta_{\max}$), e.g., (Lin, 0.5).

Adaptive $\beta_{\max}$ Depending on the task and the type of IM method, the scale of the intrinsic return may be extremely smaller than the scale of the extrinsic return, requiring $\beta_{\max}$ to be large. If $\beta_{\max}$ is set too small, all IRCs will converge to $\beta_{\max}$ but can not become larger than $\beta_{\max}$ as in Figure 4b. If we have access to the upper bound (UB), or a sufficiently satisfactory goal of the given task's extrinsic return (e.g., the upper bound of all MiniGrid tasks' extrinsic return is 1.0), we may adaptively select $\beta_{\max}$ large enough according to the first period's mean intrinsic return. We initialize the IRCs of all actors to zero and fix them for the first period. During the first period, we do not use the intrinsic rewards for training, but only store them to measure the scale of the mean intrinsic return. Then we calculate the appropriate IRC β_{app} as follows.

$$\beta_{\text{app}} \underset{\tau \in \text{1st period}}{\text{MEAN}} \left[\sum_{\tau} r_t^{\text{int}} \right] = \text{UB} \left[\sum_{t=1}^{T} r_t^{\text{ext}} \right]. \quad (1)$$

After the first period, we initialize the IRC population with $\beta_{\max}$ sufficiently larger than β_{app}. Then we use the population to continue training with standard AIMDA. We denote this initilazation as (distribution, $K\times$) for $\beta_{\max} = K\beta_{\text{app}}$, e.g., (Lin, 2×).

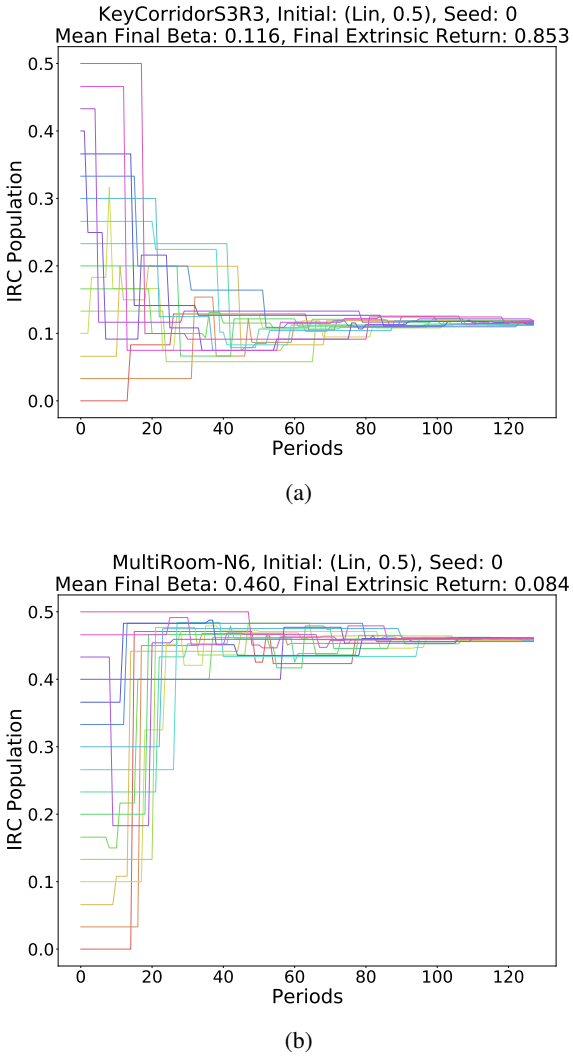

Figure 4. Adaptation of IRC population of RE3-AIMDA on KeyCorridorS3R3 and MultiRoom-N6 tasks, with an initialization (Lin, 0.5). There are 128 periods in total, where each period for this task corresponds to 78,125 frames (10M frames/128 periods).

4. Experiments

4.1. Experiment Setup

The current form of our work reports the performance of RE3-AIMDA based on RE3 (Seo et al., 2021), which is a recently proposed compute-efficient and well-reproduced IM method. Note that AIMDA can be combined with any other IM methods such as count-based exploration (Bellemare et al., 2016), ICM (Pathak et al., 2017), RND (Burda et al., 2019b), and RIDE (Raileanu & Rocktäschel, 2020). Our implementation is based on the open source implementation of RE3.[1] From the reference implementation, we keep all the training process and the hyperparameters, but only modify the following three parts: **(1)** The IRC is appended to the original state embedding (1 dim. + 64 dim. = 65 dim.); **(2)** We store each actor's sum of extrinsic return for the recent period and adjust the IRC with *mix-best-two* 128 times at regular intervals during training; **(3)** We use the first nearest neighbor distance instead of the log of the mean of the top five nearest neighbor distances, which turned out to perform better in general.

We evaluate RE3-AIMDA on MiniGrid environment (Chevalier-Boisvert et al., 2018), a widely-used, sparsely rewarded gridworld benchmark that requires challenging exploration (Raileanu & Rocktäschel, 2020; Parisi et al., 2021; Seo et al., 2021; Zhang et al., 2021). Among dozens

[1] https://github.com/younggyoseo/RE3

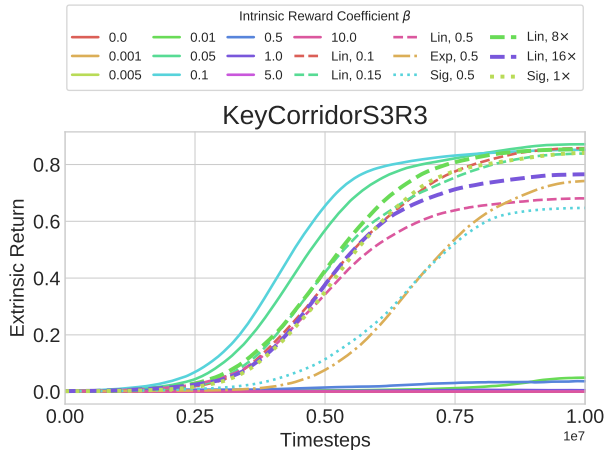

Figure 5. The extrinsic return of RE3 with fixed IRC and RE3-AIMDA on KeyCorridorS3R3, mean of 8 random seeds. RE3-AIMDA methods achieve competitive extrinsic return without the knowledge of optimal IRC.

of MiniGrid tasks, we use the pool of 11 tasks adopted from NovelD (Zhang et al., 2021). The authors of NovelD categorize MiniGrid tasks into three levels of difficulty: easy, medium, and hard (Appendix A.2). We train the agent for 10M, 20M, and 40M frames, with respect to the level of difficulty. All the tasks have only one rewarding state, which is the terminal state, with reward $1.0 - 0.9 \times (\text{steps to reach the goal})/(\text{task horizon})$.

We evaluate the baseline RE3 with various fixed IRCs as well as our proposed method with various initializations on 11 MiniGrid tasks. All experiments related to RE3 and RE3-AIMDA are run for 8 random seeds. Because there is a large amount of RE3-related experimental results, we report the individual learning curve as in Figure 5 for all tasks for all IRC configurations in Appendix B. We summarize the results using two metrics, both mean of 8 random seeds: **(1)** The final extrinsic return (FER) in Figure 6; **(2)** Normalized area under the learning curve (NAUC) for the entire course of training in Figure 8. For example, the final extrinsic returns of Figure 5 are reported in the 4-*th* row of Figure 6. The optimal return in all MiniGrid tasks is bounded between 0 and 1. The NAUC, area under the learning curve normalized by the total training timesteps, is introduced to distinguish faster convergence when both extrinsic returns are the same. If the FER is zero for a task, we call the method has *failed* to solve the task. Please refer to Appendix A for more implementation details.

4.2. RE3 with AIMDA on 11 MiniGrid Tasks

The first column of Figure 6 reports the FER of vanila A2C (i.e., RE3 with $\beta = 0$). The next 9 columns report the FER of RE3 baseline with fixed IRCs $\beta \in \{0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1.0, 5.0, 10.0\}$. They show a pattern by task similar to that shown in Figure 1, where MultiRoom tasks require larger IRCs than KeyCorridor and ObstructedMaze tasks. For MultiRoom tasks, $\beta = 5.0$ is optimal, but it fails KeyCorridor and Obstructed Maze tasks. For KeyCorridor and Obstructed Maze tasks, $\beta = 0.01$ or 0.05 is optimal, but they fail MultiRoom tasks.

We report the FERs of RE3-AIMDA with predefined $\beta_{\max}$ at 11-*th* to 16-*th* column of Figure 6. Each column reports FERs for different initialization of IRC population (distribution and $\beta_{\max}$). Exponential and sigmoid initialization methods fail three of KeyCorridor tasks due to the relative sparsity of IRCs around 0.05. On the other hand, we find that the linear initialization works more stably for more tasks. Especially $\beta_{\max} = 0.1$ and 0.15 fail only on MultiRoom-N12-S10 task. They also attain the highest FERs for ObstructedMaze tasks, although the absolute value is much lower than 1.0, due to the limitations of the baseline RE3 method. We find that setting $\beta_{\max}$ large, improves the FERs of MultiRoom tasks while setting it smaller improves the FERs of KeyCorridor and ObstructedMaze tasks.

RE3-AIMDA has to spend a significant amount of steps exploring to determine the best IRCs. Therefore RE3-AIMDA methods converge to the optimal IRCs more slowly than the RE3 baselines with optimal fixed IRCs as in Figure 5. Contrary to our expectation, the IRCs do not necessarily converge to 0 even at the end of the learning process. As in Figure 4, AIMDA prefers to maintain the IRC value similar to the optimal IRC for RE3 with fixed IRC.

4.3. Initializing $\beta_{\max}$ with the first period's measurements

From the experiments in Section 4.2, we find that $\beta_{\max}$ is a key hyperparameter of RE3-AIMDA. If the initial IRC population is too diverse with a limited number of actors, the amount of learning near the optimal IRC becomes insufficient. To overcome this problem, we evaluate the initialization scheme of $\beta_{\max}$ introduced in the last part of Section 3.3.

We report the measurements using the trajectories of an agent trained with zero IRC during the first period in Table 1. We find that the MultiRoom tasks have the lowest mean intrinsic return and the ObstructedMaze tasks have the largest mean intrinsic return. Therefore, it is natural to set $\beta_{\max}$ larger for the MultiRoom tasks and smaller for ObstructedMaze tasks as in Figure 6. Note that the mean intrinsic return (IR) is nearly proportional to the task horizon, because RE3 normalizes its intrinsic reward with a running estimate of the standard deviation, making the scale of the intrinsic reward similar across tasks.

By setting $\beta_{\max}$ large enough, good IRCs are stably included

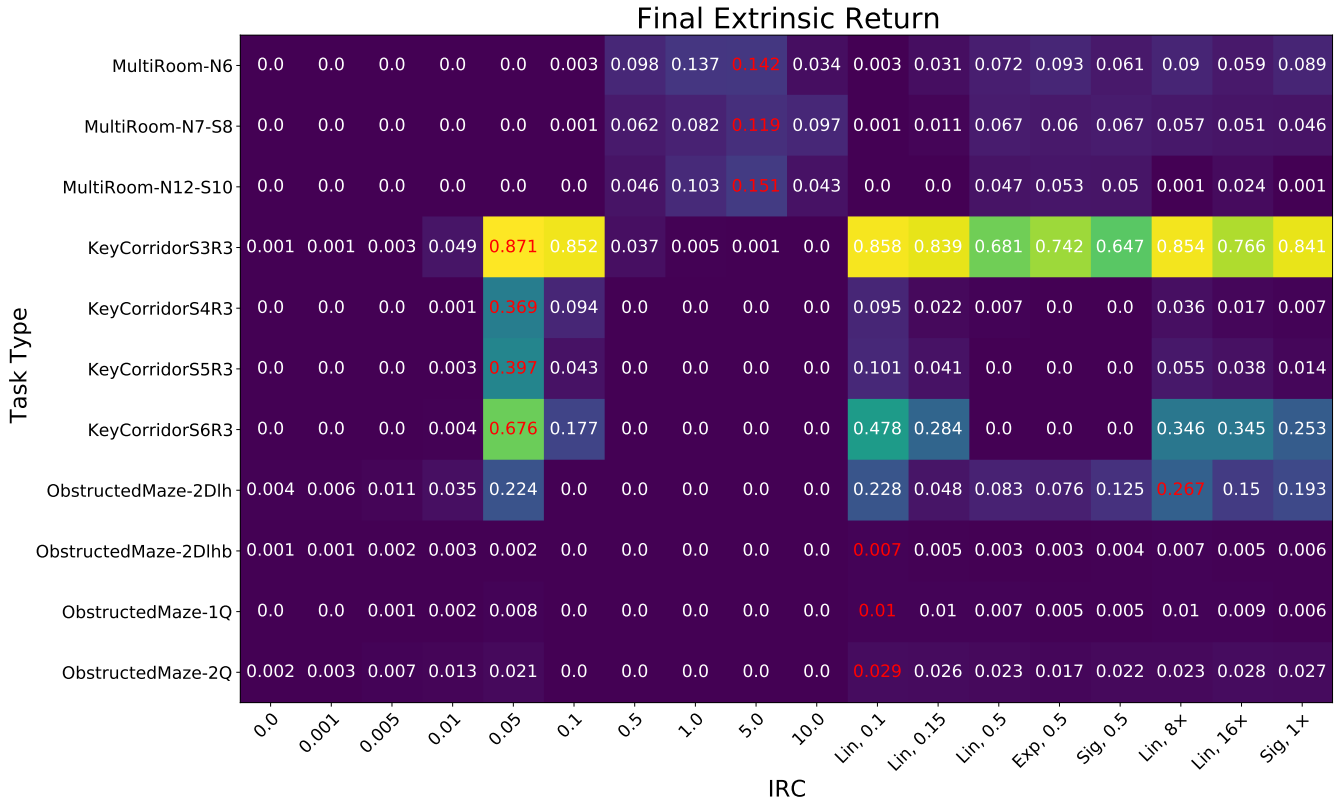

Figure 6. The final extrinsic returns of RE3 with fixed IRCs (column 2-10) and RE3 with AIMDA (column 11-18) on 11 MiniGrid tasks, mean of 8 random seeds. The largest value for each task is colored in red. Refer to the normalized area under curve in Appendix C to check the learning speed.

in the initial population. We find that RE3-AIMDA with $\beta_{\max} \geq \beta_{\text{app}}$ achieves non-zero returns for all tasks as in the last three columns of Figure 6. All fixed-IRC RE3 and other RE3-AIMDA methods with predefined $\beta_{\max}$ fail at least one task. Note that the difference between not getting a return at all and achieving even a small return is significant in the field of RL. In particular, in a sparse reward task such as MiniGrid, if even a small extrinsic reward is discovered, a steady improvement can be achieved based on this discovery.

5. Discussion and Future Work

Our experiment results showed the effectiveness of grafting decision awareness onto IM. However, what has been shown in this paper so far is only the beginning, and expect to see much more progress. Only a few combinations of decision awareness injection, initialization, and adaptation scheme is presented in this work. We expect great ideas from many researchers. Several directions for the future development and work-in-progress are presented as follows.

Emphasis on β-dependence of IM methods Currently we have reported the result on all 11 MiniGrid-tasks for only RE3 in Fig 6. For other IM methods, we have run experiments on a subset, MultiRoom-N6 and KeyCorridorS3R3 in Fig 1 yet, we are running experiments on the remaining tasks for these IM methods to emphasize the IRC dependence more significantly.

AIMDA on more IM methods Although our work points out the problems of various IM methods, at present, only experiments in which AIMDA is applied to RE3 have been carried out. We are working on applying AIMDA to other IM methods such as count-based exploration (Bellemare et al., 2016), ICM (Pathak et al., 2017), RND (Burda et al., 2019b), and RIDE (Raileanu & Rocktäschel, 2020). We expect AIMDA to be generally applicable to other IM methods for overall performance improvement.

Mixture of multiple IM methods NovelD (Zhang et al., 2021) and C-BET (Parisi et al., 2021) propose to use the difference of RND rewards and the sum of count-based rewards, respectively. As Rainbow (Hessel et al., 2018) achieves good performance by combining multiple success-

Table 1. **Column 3-4:** Measurements during the first period trained without intrinsic motivation (i.e., zero IRC). **Column 5-7:** Measurements during the last period trained with the best fixed IRC β^* that returns the best extrinsic return (Figure 6). IR and ER denote the RE3 intrinsic return and extrinsic return respectively. For the first period, the agent is only trained with ER. The first period corresponds to 78,125 steps, 156,250 steps, and 312,500 steps for easy, medium, and hard tasks, respectively. The upper bound of maximum extrinsic return is 1.0 for all MiniGrid tasks therefore we set $\beta_{\text{app}} = 1.0/\text{Mean IR}$.

		Trained with zero IRC First period			Trained with the best IRC Last Period	
	Task Horizon	Mean IR	Appropriate IRC	Best IRC	Mean Episode Length	Mean ER
Task	H	Mean IR	β_{app}	β^*	T^*	$1 - 0.9 \times T^*/H$
MultiRoom-N6	120	15.727	0.064	5.00	101.04±24.87	0.242
MultiRoom-N7-S8	140	18.666	0.054	5.00	122.31±27.09	0.214
MultiRoom-N12-S10	240	32.449	0.031	5.00	207.56±44.27	0.222
KeyCorridorS3R3	270	36.311	0.028	0.05	38.60±20.54	0.871
KeyCorridorS4R3	480	52.642	0.019	0.05	283.39±88.12	0.469
KeyCorridorS5R3	750	102.549	0.010	0.05	430.32±190.45	0.484
KeyCorridorS6R3	1080	127.481	0.008	0.05	370.49±310.19	0.691
ObstructedMaze-2Dlh	576	84.237	0.012	0.05	433.83±121.79	0.322
ObstructedMaze-2Dlhb	576	83.959	0.012	0.01	575.15±3.31	0.101
ObstructedMaze-1Q	720	58.963	0.017	0.05	716.11±15.07	0.105
ObstructedMaze-2Q	1584	222.238	0.004	0.05	1563.58±69.9	0.112

ful RL techniques, we may also consider a way to integrate multiple IM methods using AIMDA. If AIMDA can show overall improvements for general IM methods, we can use the weighted sum of multiple types of IM rewards, where the optimal weight is adaptively selected by the decision-aware agent.

Initial population of intrinsic reward coefficients In this work, we demonstrate linear, exponential, and sigmoid initializations with a predefined β_{max} or with an adaptive β_{max} based on the first period's intrinsic return. The adaptive selection of β_{max} has limitations in that the first period measurements may be noisy and the scale of the intrinsic return may vary throughout the learning process. The heuristic we proposed is just one of many possible initialization methods, and we expect to find a generally applicable initialization method better than our current form.

Adaptation scheme There are many plausible ways to adaptively adjust the IRC for the best extrinsic reward. We are trying multiple adaptation schemes in addition to the proposed method of mixing the best two to replace the worst one. For example, we may replace the worst coefficient with the best coefficient without mixing. We may model a random mutation to perturb the IRC as well as the crossover. We may shift the entire intrinsic reward population at the same time based on the performance. One thing to note is that the extrinsic returns are highly non-stationary, therefore we should avoid changing the coefficient too abruptly.

References

Andrychowicz, M., Raichuk, A., Stańczyk, P., Orsini, M., Girgin, S., Marinier, R., Hussenot, L., Geist, M., Pietquin, O., Michalski, M., Gelly, S., and Bachem, O. What matters for on-policy deep actor-critic methods? A large-scale study. In *International Conference on Learning Representations (ICLR)*, 2021.

Badia, A. P., Sprechmann, P., Vitvitskyi, A., Guo, D., Piot, B., Kapturowski, S., Tieleman, O., Arjovsky, M., Pritzel, A., Bolt, A., and Blundell, C. Never give up: Learning directed exploration strategies. In *International Conference on Learning Representations (ICLR)*, 2020.

Ball, P., Parker-Holder, J., Pacchiano, A., Choromanski, K., and Roberts, S. Ready policy one: World building through active learning. In *International Conference on Machine Learning (ICML)*, pp. 119:591–601, 2020.

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 (NeurIPS)*, pp. 29:1471–1479, 2016.

Burda, Y., Edwards, H., Pathak, D., Storkey, A., Darrell, T., and Efros, A. A. Large-scale study of curiosity-driven learning. In *International Conference on Learning Representations, (ICLR)*, 2019a.

Burda, Y., Edwards, H., Storkey, A., and Klimov, O. Exploration by random network distillation. In *International Conference on Learning Representations (ICLR)*, 2019b.

Chevalier-Boisvert, M., Willems, L., and Pal, S. Minimalistic gridworld environment for openai gym. https://github.com/maximecb/gym-minigrid, 2018.

Engstrom, L., Ilyas, A., Santurkar, S., Tsipras, D., Janoos, F., Rudolph, L., and Madry, A. Implementation matters in deep rl: A case study on ppo and trpo. In *International Conference on Learning Representations (ICLR)*, 2020.

Espeholt, L., Soyer, H., Munos, R., Simonyan, K., Mnih, V., Ward, T., Doron, Y., Firoiu, V., Harley, T., Dunning, I., Legg, S., and Kavukcuoglu, K. IMPALA: Scalable distributed deep-RL with importance weighted actor-learner architectures. In *International Conference on Machine Learning (ICML)*, pp. 80:1407–1416, 2018.

Eysenbach, B., Gupta, A., Ibarz, J., and Levine, S. Diversity is all you need: Learning skills without a reward function. In *International Conference on Learning Representations (ICLR)*, 2019.

Florensa, C., Held, D., Geng, X., and Abbeel, P. Automatic goal generation for reinforcement learning agents. In *International Conference on Machine Learning (ICML)*, pp. 80:1515–1528, 2018.

Fortunato, M., Azar, M. G., Piot, B., Menick, J., Osband, I., Graves, A., Mnih, V., Munos, R., Hassabis, D., Pietquin, O., Blundell, C., and Legg, S. Noisy networks for exploration. In *International Conference on Learning Representations (ICLR)*, 2018.

Franke, J. K., Köhler, G., Biedenkapp, A., and Hutter, F. Sample-efficient automated deep reinforcement learning. In *International Conference on Learning Representations (ICLR)*, 2021.

Gregor, K., Rezende, D. J., and Wierstrae, D. Variational intrinsic control. In *International Conference on Learning Representations (ICLR)*, 2017.

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 (ICML)*, pp. 80:1861–1870, 2018.

Henderson, P., Islam, R., Bachman, P., Pineau, J., Precup, D., and Meger, D. Deep reinforcement learning that matters. In *Association for the Advancement of Artificial Intelligence (AAAI)*, pp. 392:3207–3214, 2018.

Hessel, M., Modayil, J., van Hasselt, H., Schaul, T., Ostrovski, G., Dabney, W., Horgan, D., Piot, B., Azar, M., and Silver, D. Rainbow: Combining improvements in deep reinforcement learning. In *Association for the Advancement of Artificial Intelligence (AAAI)*, pp. 32:3215–3222, 2018.

Houthooft, R., Chen, X., Duan, Y., Schulman, J., Turck, F. D., and Abbeel, P. Vime: Variational information maximizing exploration. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 29:1109–1117, 2016.

Hutter, F., Vanschoren, J., and Kotthoff, L. *Automated Machine Learning - Methods, Systems, Challenges*. The Springer Series on Challenges in Machine Learning, Springer, 2019.

Jaderberg, M., Dalibard, V., Osindero, S., Czarnecki, W. M., Donahue, J., Razavi, A., Vinyals, O., Green, T., Dunning, I., Simonyan, K., Fernando, C., and Kavukcuoglu, K. Population based training of neural networks. *arXiv preprint arXiv: 1711-09846*, 2017.

Kapturowski, S., Ostrovski, G., Quan, J., Munos, R., and Dabney, W. Recurrent experience replay in distributed reinforcement learning. In *International Conference on Learning Representations (ICLR)*, 2018.

Kearns, M. and Singh, S. Bias-variance error bounds for temporal difference updates. In *Computational Learning Theory (COLT)*, pp. 142–147, 2000.

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. In *International Conference on Learning Representations (ICLR)*, 2016.

Liu, S., Lever, G., Merel, J., Tunyasuvunakool, S., Heess, N., and Graepel, T. Emergent coordination through competition. In *International Conference on Learning Representations (ICLR)*, 2019.

Martin, J., Sasikumar, S. N., Everitt, T., and Hutter, M. Count-based exploration in feature space for reinforcement learning. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 2471–2478, 2017.

Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T. P., Harley, T., Silver, D., and Kavukcuoglu, K. Asynchronous methods for deep reinforcement learning. In *International Conference on Machine Learning (ICML)*, pp. 48:1928–1937, 2016.

Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., and Kavukcuoglu, K. Agent57: Outperforming the Atari human benchmark. In *International Conference on Machine Learning (ICML)*, pp. 119:507–517, 2020.

Moskovitz, T., Parker-Holder, J., Pacchiano, A., Arbel, M., and Jordan, M. Tactical optimism and pessimism for deep reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 34:12849–12863, 2021.

Nair, A. V., Pong, V., Dalal, M., Bahl, S., Lin, S., and Levine, S. Visual reinforcement learning with imagined goals. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 31:9191–9200, 2018.

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, (ICML)*, pp. 70:2721–2730, 2017.

Parisi, S., Dean, V., Pathak, D., and Guptal, A. Interesting object, curious agent: Learning task-agnostic exploration. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 34:20516–20530, 2021.

Parker-Holder, J., Pacchiano, A., Choromanski, K. M., and Roberts, S. J. Effective diversity in population based reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 33:18050–18062, 2020.

Parker-Holder, J., Rajan, R., Song, X., Biedenkapp, A., Miao, Y., Eimer, T., Zhang, B., Nguyen, V., Calandra, R., Faust, A., Hutter, F., and Lindauer, M. Automated reinforcement learning (AutoRL): A survey and open problems. *arXiv preprint arXiv: 2201.03916*, 2022.

Pathak, D., Agrawal, P., Efros, A. A., and Darrell, T. Curiosity-driven exploration by self-supervised prediction. In *International Conference on Machine Learning, (ICML)*, pp. 70:2778–2787, 2017.

Paul, S., Kurin, V., and Whiteson, S. Fast efficient hyperparameter tuning for policy gradient methods. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 32:4618–4628, 2019.

Pislar, M., Szepesvari, D., Ostrovski, G., Borsa, D. L., and Schaul, T. When should agents explore? In *International Conference on Learning Representations (ICLR)*, 2022.

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

Schaul, T., Borsa, D., Ding, D., Szepesvari, D., Ostrovski, G., Dabney, W., and Osindero, S. Adapting behaviour for learning progress. *arXiv preprint arXiv: 1912.06910*, 2019.

Seo, Y., Chen, L., Shin, J., Lee, H., Abbeel, P., and Lee, K. State entropy maximization with random encoders for efficient exploration. In *International Conference on Machine Learning, (ICML)*, pp. 139:9443–9454, 2021.

Stadie, B. C., Levine, S., and Abbeel, P. Incentivizing exploration in reinforcement learning with deep predictive models. *arXiv preprint arXiv: 1507.00814*, 2015.

Sutton, R. and Singh, S. On step-size and bias in temporal-difference learning. In *Center for Systems Science, Yale University*, pp. 91–96, 1994.

Tang, H., Houthooft, R., Foote, D., Stooke, A., Chen, 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 (NeurIPS)*, pp. 30:2753–2762, 2017.

White, M. and White, A. A greedy approach to adapting the trace parameter for temporal difference learning. In *International Conference on Autonomous Agents & Multiagent (AAMAS)*, pp. 15:557–565, 2016.

Zhang, B., Rajan, R., Pineda, L., Lambert, N., Biedenkapp, A., Chua, K., Hutter, F., and Calandra;, R. On the importance of hyperparameter optimization for model-based reinforcement learning. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 130:4015–4023, 2016.

Zhang, T., Xu, H., Wang, X., Yi Wu, K. K., Gonzalez, J. E., and Tianl, Y. Noveld: A simple yet effective exploration criterion. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 34:25217–25230, 2021.

A. Implementation Details

A.1. Count-based, ICM, RND, and RIDE

We use the reference implementation of RIDE[2] (Raileanu & Rocktäschel, 2020), which also provides the implementations of other IM methods: count-based (Bellemare et al., 2016), ICM (Pathak et al., 2017), and RND (Burda et al., 2019b). They use IMPALA (Espeholt et al., 2018) with recurrence as their baseline RL algorithm. Following Raileanu & Rocktäschel (2020), we set the entropy coefficient for action to 0.0001 for ICM, RND, and Count on all environments. For RIDE, we use entropy coefficient of 0.0005 for KeyCorridorS3R3 and use 0.001 for MultiRoom-N6. We train 16 actors in parallel with 40 shared-memory buffers. We currently report experimental results on two tasks: KeyCorridor-S3R3 and MultiRoom-N6, but we are running further experiments on the remaining 9 tasks. We run experiments over 9 values of IRCs, $\beta \in \{0.0001, 0.0005, 0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1.0\}$. We train all configurations for 40M frames and 3 random seeds. We report the results with a moving average of window size 4M steps.

A.2. RE3 and RE3-AIMDA

The main experiments in Section 4 are based on the implementations of RE3.[3] The baseline RL algorithm is A2C (Mnih et al., 2016) without recurrence. We run experiments on 11 MiniGrid tasks with a different number of total steps respective to their level of difficulty (Zhang et al., 2021). We report the results with a moving average of window size equal to 0.1 times the total training steps.

1. Easy (10M steps): MultiRoom-N6, MultiRoom-N7-S8, MultiRoom-N12-S10, and KeyCorridorS3R3
2. Medium (20M steps): KeyCorridorS4R3, KeyCorridorS5R3, KeyCorridorS6R3, and MiniGrid-ObstructedMaze-2Dlh
3. Hard (40M steps): MiniGrid-ObstructedMaze-2Dlhb, ObstructedMaze-1Q, and ObstructedMaze-2Q

All experimental results on RE3 and RE3-AIMDA are reported as the mean of 8 random seeds. We evaluate the agent every 2048 steps. The normalized area under curve (NAUC) is calculated as the sum of the extrinsic returns of all evaluations divided by the total number of evaluations.

RE3 We run a search over 10 values of IRCs $\beta \in \{0.0, 0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1.0, 5.0, 10.0\}$. The agent is trained with only the extrinsic reward when $\beta = 0.0$.

RE3-AIMDA The raw observation in $(7 \times 7 \times 3)$ is passed through 3 constitutional layers and encoded in 64 dimensions. Then we append the raw value of the IRC to form the concatenation in 65 dimensions. Regardless of the total number of learning steps, we perform the *mix-best-two* at regular periods. However, we do not perform *mix-best-two* when all actors fail to gain a non-zero return. When there is a tie between the best IRCs or between the worst IRCs, we sample one uniformly at random among the candidates.

[2] https://github.com/facebookresearch/impact-driven-exploration
[3] https://github.com/younggyoseo/RE3

B. Learning Curves of RE3 on 11 MiniGrid Tasks

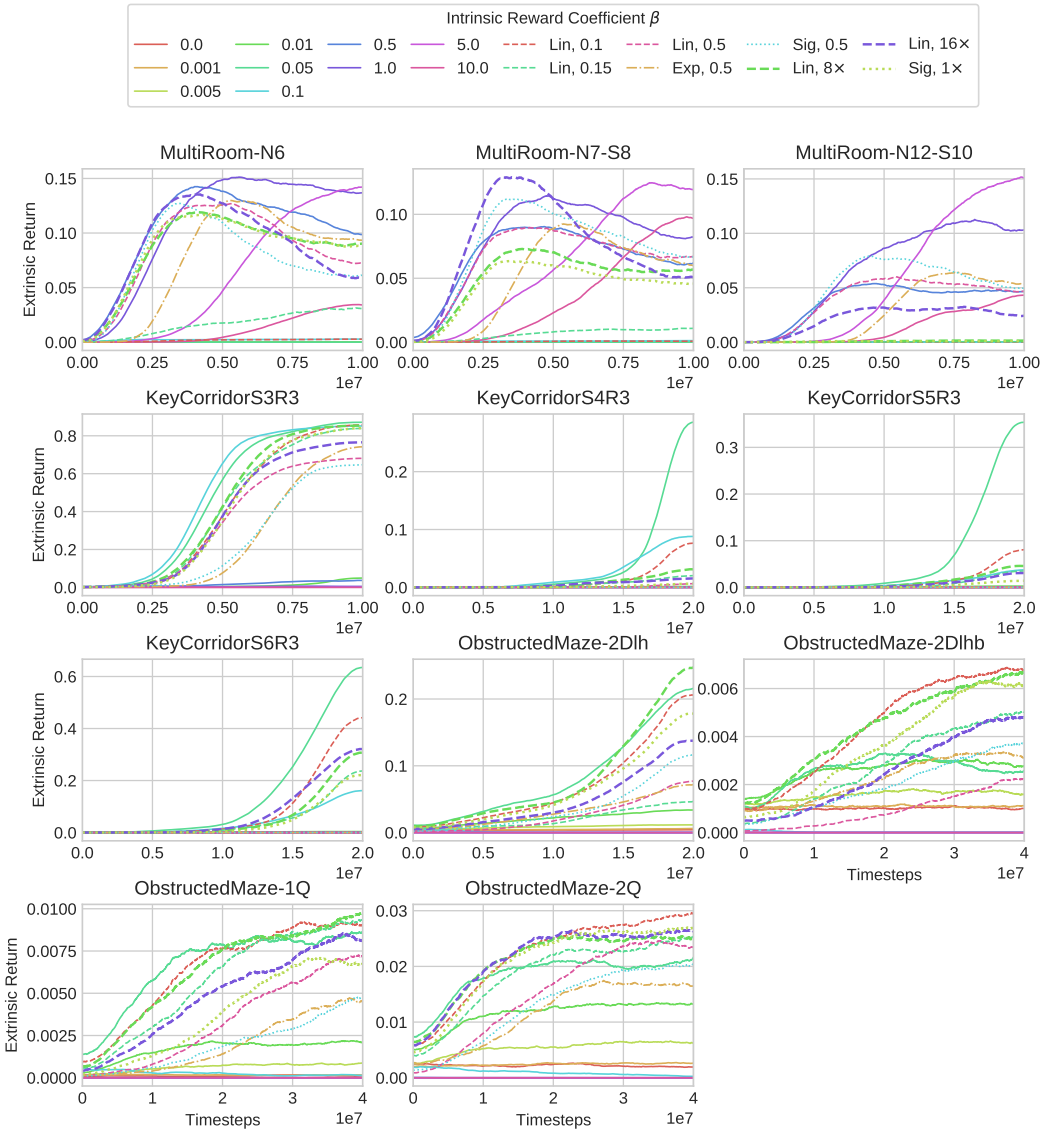

Figure 7. Learning curves of RE3 and RE3-AIMDA on 11MiniGrid tasks. These results are used to report the FER and NAUC summary in Figure 6 and Figure 8, respectively. Refer to Appendix A.2 for implementation details to obtain the experimental results.

C. Normalized Area Under Curve of RE3 and RE3-AIMDA.

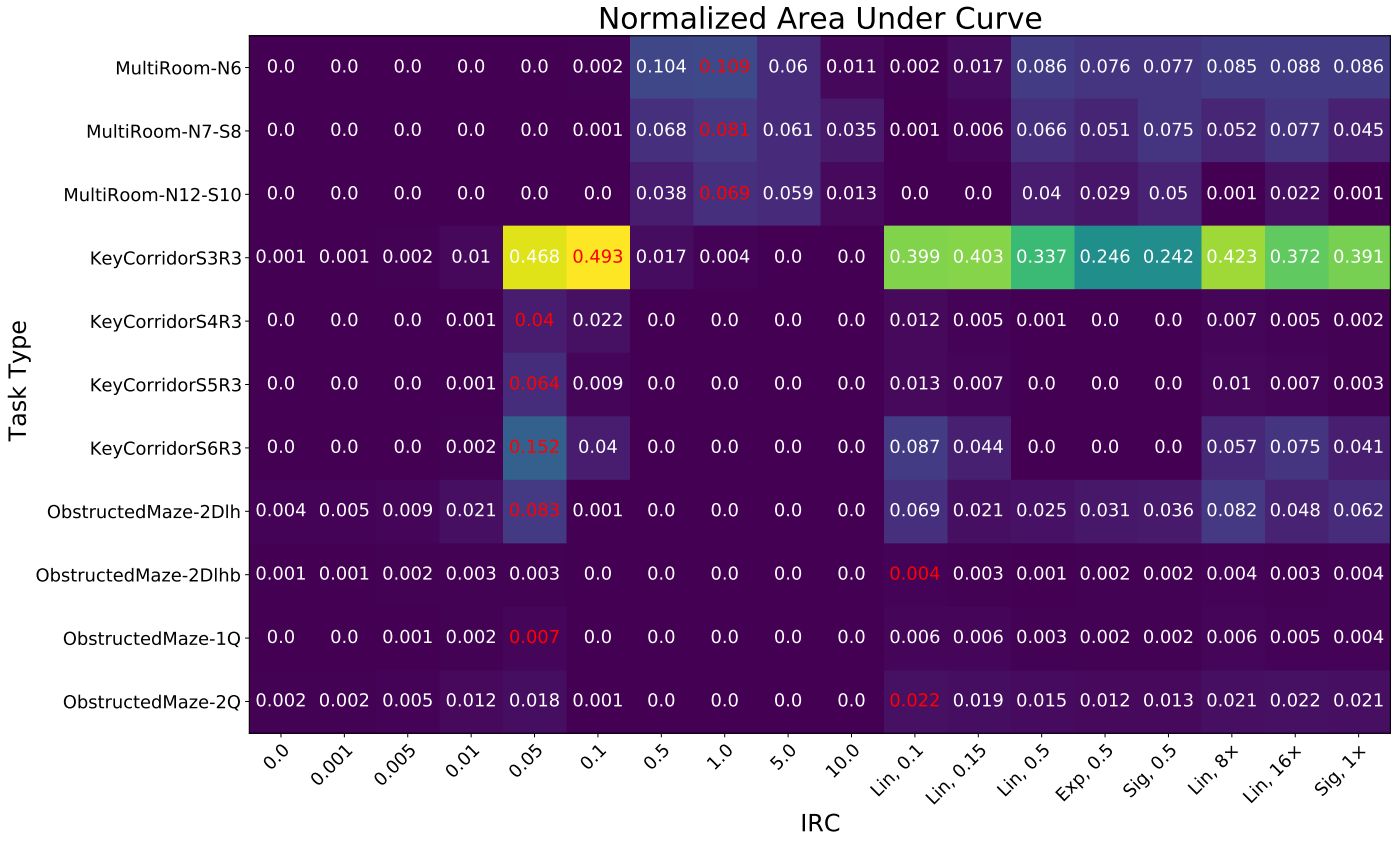

Figure 8. Normalized Area Under Curve of RE3 and RE3-AIMDA on 11 MiniGrid Tasks. Mean of 8 random seeds. The largest value for each task is colored in red.