I^{2}HRL: Interactive Influence-based Hierarchical Reinforcement Learning

Rundong Wang, **Runsheng Yu**, **Bo An** and **Zinovi Rabinovich**
School of Computer Science and Engineering, Nanyang Technological University, Singapore
{rundong001, runsheng.yu, boan, zinovi}@ntu.edu.sg

Abstract

Hierarchical reinforcement learning (HRL) is a promising approach to solve tasks with long time horizons and sparse rewards. It is often implemented as a high-level policy assigning subgoals to a low-level policy. However, it suffers the high-level non-stationarity problem since the low-level policy is constantly changing. The non-stationarity also leads to the data efficiency problem: policies need more data at non-stationary states to stabilize training. To address these issues, we propose a novel HRL method: *Interactive Influence-based Hierarchical Reinforcement Learning* (I^{2}HRL). First, inspired by agent modeling, we enable the interaction between the low-level and high-level policies, i.e., the low-level policy sends its policy representation to the high-level policy. The high-level policy makes decisions conditioned on the received low-level policy representation as well as the state of the environment. Second, we stabilize the training of the high-level policy via an information-theoretic regularization with minimal dependence on the changing low-level policy. Third, we propose the influence-based exploration to more frequently visit the non-stationary states where more transition data is needed. We experimentally validate the effectiveness of the proposed solution in several tasks in MuJoCo domains by demonstrating that our approach can significantly boost the learning performance and accelerate learning compared with state-of-the-art HRL methods.

1 Introduction

Reinforcement Learning (RL) methods have recently yielded a plethora of positive results, including playing games like Go [Silver *et al.*, 2016] and Atari [Mnih *et al.*, 2013], as well as controlling robots [Lillicrap *et al.*, 2015]. However, it is still challenging to learn policies in complex environments with large time horizons and sparse rewards. A promising method to address these issues is Hierarchical RL (HRL) that learns to operate at different temporal abstraction levels simultaneously. Recent end-to-end HRL methods, where the high-level policy periodically assigns subgoals for the low-level policy to pursue, have shown greatly improved performance in sparse reward problems.

Unfortunately, current HRL methods are subject to the non-stationarity problem [Nachum *et al.*, 2018; Levy *et al.*, 2019]. Namely, as the lower-level policy continues to change, a high-level action taken at the same state, but at different steps, may result in critically different state transitions and rewards. This negatively impacts policy exploration, since policies need more data at non-stationary states to stabilize training, i.e., an already visited non-stationary state may warrant further exploration. Yet, common exploration approaches in RL, such as count-based exploration [Bellemare *et al.*, 2016], re-focus agents on less-visited states, which makes them ill-suited to support HRL. Though some remedies to HRL's non-stationarity issue were proposed, e.g., off-policy experience correction [Nachum *et al.*, 2018], or pre-training and freezing the low-level policy [Eysenbach *et al.*, 2018], they require additional manual configuration. This either breeds more hyperparameters or breaks the end-to-end scheme entirely.

To address these issues, we develop a novel HRL approach named *Interactive Influence-based Hierarchical Reinforcement Learning* (I^{2}HRL). Our contributions are threefold. First, we introduce a feedback loop from the process of low-level policy learning to the high-level policy. The latter can now condition its decisions on the features of the low-level behaviour policy. Second, we propose an influence-based framework and introduce information-theoretic regularization to control the dependency of the high-level policy on the changes in the low-level behaviour. This stabilises the high-level policy training. Finally, we propose an influence-based exploration method for the high-level policy that improves sample efficiency. Intuitively, if a state, where the dependency of the high-level policy on the low-level behaviour is stronger, is a potential failure point due to the changing low-level policy and should be explored more.

We compare our method with state-of-the-art HRL algorithms on several continuous controlling tasks in the MuJoCo domain [Duan *et al.*, 2016]. Experimental results show that our method significantly outperforms existing algorithms. Bi-directional communication/interaction with the influence-based framework can accelerate the learning process, alleviate non-stationarity issues and improve data efficiency.

2 Related Work

HRL has been shown to be effective in dealing with long-horizon and sparse reward problems. Intuitively, HRL is (at least) a bi-level approach, where the high-level policy breaks down a problem into sub-tasks and learns to sequence them, while the low-level learns to resolve the sub-tasks efficiently. The specifics of the break-down, and how exactly the high-level communicates to the low-level, varies among methods. The signal sent to the low-level can thus be some discrete values for selection of options [Bacon *et al.*, 2017] or skills [Konidaris and Barto, 2009], or it can be continuous vectors to set a subgoal in the state space [Nachum *et al.*, 2018] or latent space [Vezhnevets *et al.*, 2017]. Generally, off-policy HRL algorithms [Levy *et al.*, 2019; Nachum *et al.*, 2018] are more efficient than on-policy algorithms [Bacon *et al.*, 2017; Vezhnevets *et al.*, 2017]. However, the off-policy scheme creates non-stationarity for the high-level policy, since the low-level policy is constantly changing. Various solutions to this issue were recently proposed, e.g., Hierarchical reinforcement learning with off-policy correction (HIRO) [Nachum *et al.*, 2018] uses joint training and an off-policy correction, which modifies old transitions into new ones that agree with the current low-level policy. Unfortunately, this means that the higher level may need to wait until the lower level policy converges before it can learn a meaningful policy itself. Hierarchical Actor-Critic (HAC) [Levy *et al.*, 2019] introduces hindsight action transitions, which simulate a transition function using the optimal lower-level policy hierarchy, to train the high-level policy. Also, these transitions need carefully designed domain specific rewards. Other methods [Florensa *et al.*, 2017; Heess *et al.*, 2016; Eysenbach *et al.*, 2018] break down the end-to-end manner by pre-training and fixing the low-level policy. The frozen low-level skills require a well-designed pre-training environment and cannot be adapted to all tasks. In contrast, we propose an efficient end-to-end solution by enabling bi-directional communication among HRL levels. Furthermore, by regularizing the dependency of the high-level on the low-level policy we stabilize our method against non-stationarity.

Now, one of HRL promised benefits is structured exploration, i.e., explore with sub-task policies rather than primitive actions. However, the exploration/sample efficiency still matters [Nachum *et al.*, 2019]. Majority of HRL methods directly use single-agent exploration methods at the low-level, such as $\epsilon-$greedy or intrinsic motivation [Kulkarni *et al.*, 2016; Rafati and Noelle, 2019]. Nonetheless, some works do focus on the high-level policy exploration, e.g., the diversity concept is often utilized to drive the variability in the high-level policy choices [Eysenbach *et al.*, 2018; Florensa *et al.*, 2017]. The prerequisite, or course, is that a diverse set of low-level skills exists, which also requires well-designed pre-training environments. Moreover, the diversity guidance does not consider the interaction between the two levels. In this paper, we propose to guide the high-level exploration by the low-level policy influence. Intuitively, in a state, where the high-level action choice is less influenced by the particulars of the low-level policy, the high-level transition is near-stationary, and needs less exploration data.

Overall, our method can be best intuited if HRL is interpreted as a form of multi-agent reinforcement learning (MARL). After all, agent modeling and communication are feasible solutions for addressing non-stationarity in MARL [Papoudakis *et al.*, 2019]. By modelling the intentions and policies of others, agents can stabilize their training process. Training stabilization is also achieved through communication, where agents exchange information about their observations, actions and intentions. In our methods, considering the nature of cooperation between two levels of HRL, the low-level policy directly sends its policy representation to the high-level policy. To our knowledge, no prior work has connected these multi-agent solutions to HRL.

3 Preliminaries

In this section, we present some fundamental background of the RL as well as HRL.

3.1 Reinforcement Learing

An RL problem is generally studied as a Markov decision process (MDP), defined by the tuple: $\mathrm{MDP} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, r, \gamma, T)$, where $\mathcal{S} \subseteq \mathbb{R}^n$ is an n-dimensional state space, $\mathcal{A} \subseteq \mathbb{R}^m$ an m-dimensional action space, $\mathcal{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow \mathbb{R}_+$ a transition probability function, $r : \mathcal{S} \rightarrow \mathbb{R}$ a bounded reward function, $\gamma \in (0, 1]$ a discount factor and T a time horizon. In MDP, an agent receives the current state $s_t \in \mathcal{S}$ from the environment and performs an action $a_t \in \mathcal{A}$. The agent's actions are often defined by a policy $\pi_\theta : S \rightarrow \mathcal{A}$ parameterized by θ. The objective of the agent is to learn an optimal policy: $\pi_{\theta^*} := \mathrm{argmax}_{\pi_\theta} \mathbb{E}_{\pi_\theta} \left[\sum_{i=0}^{T} \gamma^i r_{t+i} | s_t = s \right]$.

3.2 Hierarchical Reinforcement Learning

One drawback of the generic RL is its inability to effectively handle long-term credit assignment problems, particularly in the presence of sparse rewards. HRL proposes methods for decomposing complex tasks into simpler subproblems that can be more readily tackled by low-level action primitives. We follow the two-level goal-conditioned off-policy hierarchy presented in HIRO [Nachum *et al.*, 2018]. A high-level policy π_h computes a state-space subgoal $g_t \sim \pi_h(s_t)$ every k time steps (g_t is also written as a_h in the rest of paper). Then a low-level policy π_l takes as an input the current state s_t and the assigned subgoal g_t and is encouraged to perform an action $a_t \sim \pi_l(s_t, g_t)$ that satisfies its subgoal via a low-level intrinsic reward function $r_l(s_t, g_t, s_{t+1})$. Finally, the high-level policy receives cumulative rewards $r_h = \sum_{i=1}^{k} r(s_{t+i})$. The low-level reward function is set as: $r_l(s_t, g_t, s_{t+1}) = -\|s_t + g_t - s_{t+1}\|_2$, and the subgoal-transition function is set as $g_{t+1} = \pi_h(s_t)$, if $t \bmod k = 0$, or otherwise using a fixed goal transition function $h(s_t, g_t, s_{t+1}) = s_t + g_t - s_{t+1}$.

4 Interactive Influence-based HRL

In this section, we present our framework for learning hierarchical policies with bi-direction communication and the high-level exploration. First, we make use of the low-level policy modeling and pass it to the high-level policy. In addition, we minimize the influence of the low-level policy which

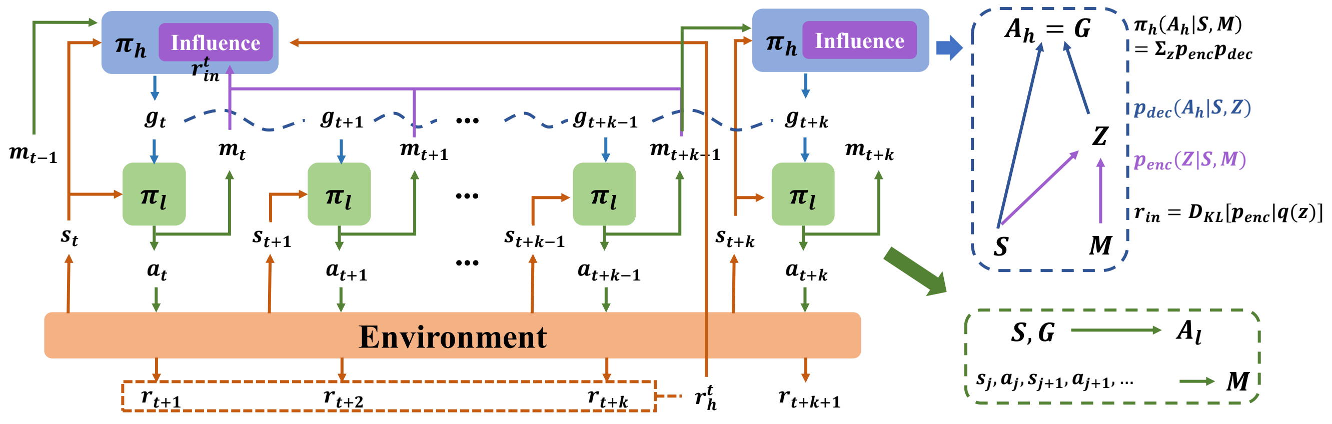


Figure 1: Overview of I^{2}HRL, which consists of the interaction between the low-level and high-level policies, as well as an influence-based framework. At the high-level decision step t, the high-level policy receives the worker's previous message m_{t-1} and state s_t and sends a goal $g_t = \pi_h(s_t, m_{t-1})$. During the high-level decision interval $\{t+1, t+2, \cdots, t+k-1\}$, the subgoal follows the transition function $g_{t+i} = h(s_{t+i-1}, g_{t+i-1}, s_{t+i})$. The messages $\{m_t, m_{t+1}, \cdots, m_{t+k-1}\}$ sent by the low-level policy are used to compute the high-level intrinsic rewards. The rewards for the high-level policy: $r_h^t = \sum_{j=1,2,\cdots,k}(r(s_{t+j}) + r_{in}(s_{t+j}, m_{t+j}))$

is quantified by the mutual information between the subgoals assigned by the high-level policy and the low-level policy representation. Lastly, we introduce the influence-based exploration with intrinsic rewards for the high-level policy.

We have two MDPs for the high-level policy and low-level policy respectively:

$$\text{MDP}_h = (\mathcal{S}, \mathcal{A}_h, \mathcal{P}_h, r_h, \gamma, T/k)$$
$$\text{MDP}_l = (\mathcal{S}, \mathcal{A}_l, \mathcal{P}_l, r_l, \gamma, k)$$

The non-stationarity emerges in the high-level policy transition. That is, at different steps, $\mathcal{P}_h$ outputs different probability and r_h outputs different rewards given the same transition. The cause of non-stationary transition functions is the change of the lower level policy. It is similar to the situation in the multi-agent reinforcement learning: the state transition function P and the reward function of each agent r_i depend on the actions of all agents. Each agent keeps changing to adapt to other agents, thus breaking the Markov assumption that governs most single-agent RL algorithms. It is natural to view HRL as cooperative two-agent reinforcement learning with such characteristics: (1) unshared rewards, (2) one-direction delayed communication and (3) fully observability.

4.1 Low-level Policy Modeling

Motivated by agent/opponent modeling, if the high-level policy can know or reason about the behaviors and the intentions of the low-level policy, the coordination efficiency will be improved and the training process of the agents might be stabilized. Due to the cooperation between the low-level and high-level policies, we propose the bi-direction communication where the low-level policy sends its policy representation to the high-level policy.

The low-level policy determines what to communicate. It is straight-forward to send the policy parameters $m = \theta_l$ as a complete representation of its own policy. However, it is well-known that the low-level policy network is often over-parameterized, which makes it hard to feed the parameters directly into the high-level policy. Also for agent modeling, one agent only needs to know the desires, beliefs, and intentions of others [Rabinowitz *et al.*, 2018]. Consequently, it is practical to encode the low-level policy $m = f^m(\pi_l(\cdot,\cdot))$, where the f^m is an encoder function as the representation module.

In this work, we learn a representation function that maps episodes from the low-level policy π_l to a real-valued vector embedding. In practice, we optimize the parameters θ of a function $f^m_{\theta_m} : \mathcal{E} \to \mathbb{R}^d$ where $\mathcal{E}$ denotes the space of successive state-action transitions $\langle s^t, a^t_h, a^t_l, s^{t+1}, a^{t+1}_h, a^{t+1}_l, \cdots, s^{t+c}, a^{t+c}_h, a^{t+c}_l\rangle$ of size c corresponding to the low-level policy and d is the dimension of the embedding. We propose a principle for learning a good representation of a policy: *predictive representation*. The representation should be accurate for predicting the low-level policy actions given states.

For satisfying the principle, we utilize an imitation function via supervised learning. Supervised learning does not require direct access to the reward signal, making it an attractive task for reward-directed representation learning. Intuitively, the imitation function attempts to mimic the low-level policy based on the historical behaviours. Concretely, we utilize the representation function $f^m_{\theta_m} : \mathcal{E} \to \mathbb{R}^d$, as well as an imitation function $f_\phi : \mathcal{S} \times \mathcal{A} \times \mathbb{R}^d \to [0,1]$ where ϕ are parameters of this imitation function that maps the low-level policy observation and embedding to a distribution over the agent's actions. We propose the following negative cross entropy objective to maximize with respect to ϕ and θ_m:

$$\mathop{\mathbb{E}}_{\substack{e_1 \sim D_l \backslash e_2 \\ \langle s, a_h, a_l\rangle \in e_2 \sim D_l}} \left[\log f_\phi\left(a_l | s, a_h, f^m_{\theta_m}(e_1)\right)\right] \quad (1)$$

where D_l is the replay buffer of the low-level policy. For the low-level policy, the objective function samples two distinct trajectories e_1 and e_2 (e_1 needs to be the history of e_2). The state-action pairs from e_1 are used to learn an embedding $f^m_{\theta_m}(e_1)$ that conditions the imitation function network trained on state-action pairs from e_2.

The imitation function is smaller than the low-level policy because it takes the low-level policy information as input. Also, it is trained more frequently than the low-level policy.

4.2 High-level Policy: Influence-based Framework

One may argue that the source of non-stationarity moves from the low-level policy to the low-level representation. That is, the high-level policy has the low-level policy representation but an out-of-date or inaccurate representation. We claim that the training process of the high-level policy is more stable with a representation than without a representation. Also, we introduce an influence-based framework for the high-level policy to extract useful information from the representation.

We expect the high-level policy to make decision with the most useful information and minimal influence of the low-level policy. In such situation, even if the low-level representation may be imprecise, the high-level policy can still make good decisions.

Consequently, we measure the influence via a conditional mutual information and regularize/minimize it for the high-level policy:

$$\begin{aligned} I(A_h; M|S) &= \sum_s \sum_m \sum_{a_h} p(a_h, m, s) \log \frac{p(s)p(a_h, m, s)}{p(a_h, s)p(m, s)} \\ &= \sum_s p(s) \sum_m p(m) \sum_{a_h} \pi(a_h|s, m) \log \frac{\pi_h(a_h|s, m)}{\pi_{h0}(a_h|s)} \\ &= \mathbb{E}_{\pi_h, \pi_l} \Big[D_{\mathrm{KL}} \left[\pi_h(a_h|s, m) | \pi_{h0}(a_h|s) \right] \Big] \end{aligned} \tag{2}$$

where A_h is the high-level action; M is the low-level policy representation; S is the state; $\pi_h(a_h|s, m)$ is the high-level policy. $\mathbb{E}_{\pi_h, \pi_l}$ is an abbreviation of $\mathbb{E}_{s,m,a_h\rangle \sim \pi_h, \pi_l}$ which denotes an expectation over the trajectories $\langle s, m, a_h \rangle$ generated by π_h, π_l. D_{KL} is the Kullback-Leibler divergence. $\pi_{h0}(a_h|s) = \sum_m p(m)\pi_h(a_h|s, m)$ is a *default* high-level policy without the low-level policy's influence, although the high-level policy never actually follows the *default* policy. $I(a_h; m|s) = 0$ if and only if a_h and m are independent given the state s. This mutual information can also be seen as a measure of stationarity. When the mutual information is small, whatever the low-level policy is, the high-level policy can always expect that a certain assignment of itself can lead to a high reward. The high-level policy's optimization objective is:

$$\begin{aligned} J(\theta_h) = \mathbb{E}_{\pi^h_{\theta_h}, \pi^l} \Big[& \sum_{i=0,k,\cdots} \gamma^i r_h^{t+i} \\ & - \beta D_{\mathrm{KL}} \left[\pi_h(a_h|s, m) | \pi_{h0}(a_h|s) \right] \Big] \end{aligned} \tag{3}$$

We now focus on the second term because the optimization of the cumulative rewards term can be solved by standard RL algorithms. $\pi_h(a_h|s, m)$ is a deterministic policy, so we cannot directly compute the mutual information term. So we introduce an internal variable Z. Due to the data processing inequality (DPI) [Cover and Thomas, 2012], $I(Z; M|S) \geq I(A_h; M|S)$. Therefore, minimizing $I(Z; M|S)$ also minimizes $I(A_h; M|S)$. We parameterize the policy $\pi_h(a_h|s, m)$ using an encoder $p_{\mathrm{enc}}(z|s, m)$ and a decoder $p_{\mathrm{dec}}(a_h|s, z)$ such that $\pi_h(a_h|s, m) = \sum_z p_{\mathrm{enc}}(z|s, m) p_{\mathrm{dec}}(a_h|s, z)$. Thus, we instead maximize this lower bound on $J(\theta_h)$:

$$\begin{aligned} J(\theta_h) &\geq \mathbb{E}_{\pi^h_{\theta_h}, \pi^l} \Big[\sum_{i=0,k,\cdots} \gamma^i r_h^{t+i} \Big] - \beta I(Z; M|S) \\ &= \mathbb{E}_{\pi^h_{\theta_h}, \pi^l} \Big[\sum_{i=0,k,\cdots} \gamma^i r_h^{t+i} - \beta D_{\mathrm{KL}} \left[p_{\mathrm{enc}}(z|s, m) | p(z|s) \right] \Big] \end{aligned} \tag{4}$$

where $p(z|s) = \sum_m p(m) p_{\mathrm{enc}}(z|s, m)$ is the marginalized encoding. In practice, performing this marginalization over the representation may often be intractable since there may be a continuous distribution of representation.

Inspired by the information bottleneck principle and variation information bottleneck [Tishby and Zaslavsky, 2015; Alemi *et al.*, 2016], we use a Gaussian approximation $q(z|s)$ of the marginalized encoding $p(z|s)$. Since $D_{KL}[p(z|s)||q(z|s)] \geq 0$, that is, $\sum_z p(z|s) \log p(z|s) \geq \sum_z p(z|s) \log q(z|s)$, instead, we get its variational upper bound:

$$\begin{aligned} I(Z; M|S) &= \mathbb{E}_{\pi^h_{\theta_h}, \pi^l} \left[D_{\mathrm{KL}}[p_{\mathrm{enc}}(z|s, m) | p(z|s)] \right] \\ &= \sum_s p(s) \sum_m p(m) \sum_z p_{\mathrm{enc}}(z|s, m) \log p_{\mathrm{enc}}(z|s, m) \\ &\quad - \sum_s p(s) \sum_z p(z|s) \log p(z|s) \\ &\leq \sum_s p(s) \sum_m p(m) \sum_z p(z|m, s) \log \frac{p_{\mathrm{enc}}(z|s, m)}{q(z|s)} \\ &= \mathbb{E}_{\pi^h_{\theta_h}, \pi^l} \left[D_{\mathrm{KL}}[p_{\mathrm{enc}}(z|s, m) | q(z|s)] \right] \end{aligned} \tag{5}$$

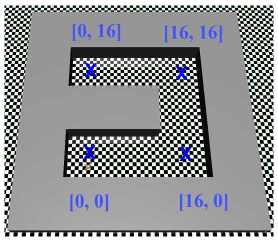
This provides a lower bound of the objective in Eq. (3) $\tilde{J}(\theta_h) \leq J(\theta_h)$ that we maximize:

$$\tilde{J}(\theta_h) = \mathbb{E}_{\pi^h_{\theta_h}, \pi^l} \Big[\sum_{i=0,k,\cdots} \gamma^i r_h^{t+i} - \beta D_{\mathrm{KL}} \left[p_{\mathrm{enc}}(z|s, m) | q(z|s) \right] \Big] \tag{6}$$

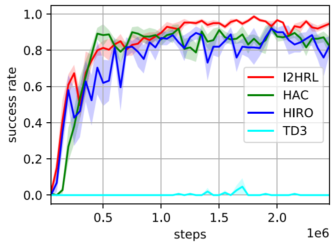
4.3 Influence-based Adaptive Exploration

Exploration in HRL heavily relies on the high level policy exploration [Nachum *et al.*, 2019]. The non-stationarity of high-level transition can lead to the inability of traditional exploration methods. During the training process of HRL, the non-stationary states instead of unvisited states need more data to stabilize the low-level policy. We can roughly split the HRL learning process into two phases with a vague boundary: (1) non-stationary phase; (2) near-stationary phase. In the first phase, i.e., at the beginning of the learning process, the low-level policy is always changing. Every state should be explored. In the second phase, the low-level policy is near-optimal. Most states are stationary for the high-level policy, thus the agent should pay more attention on the states which still cause non-stationarity. Process between these two phases can be seen as a mixture of both.

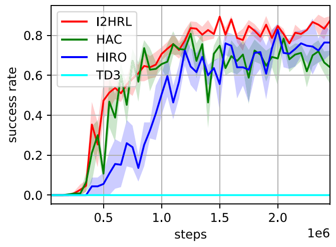
Consequently, we propose an influence-based reward for high level exploration, which is quantified by the KL divergence $D_{\mathrm{KL}} \left[p_{\mathrm{enc}}(z|s, m) | q(z) \right]$. Intuitively, if the KL divergence is small, the agent visits a state which is less easily influenced by the low-level policy. It means that the high-level transition is near-stationary, i.e., less data is needed here



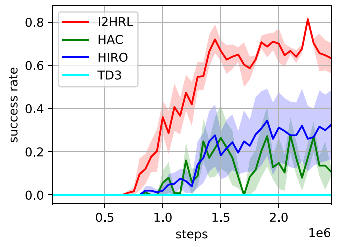
(a) AntMaze Env.



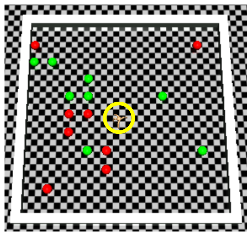
(b) Success Rate in AntMaze [16,0]

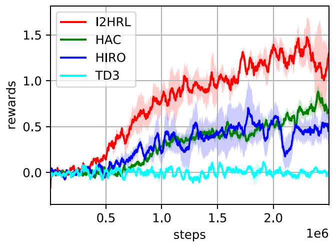


(c) Success Rate in AntMaze [16,16]

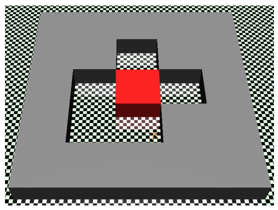


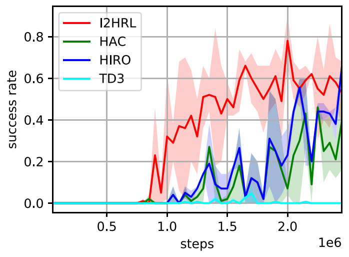
(d) Success Rate in AntMaze [0,16]


(e) AntGather Env.



(f) Evaluation Reward in AntGather


(g) AntPush Env.



(h) Evaluation Reward in AntGather

Figure 2: Evaluation results in the AntMaze, AntGather, and AntPush.

other than other states. When the KL divergence is large, a non-trivial influence of low-level policy is forced on the high-level policy. It means that the high-level policy needs more data to stabilize training.

During training, we utilize regularization mentioned in the previous section to train the high-level policy every k steps. Also, we calculate the KL divergence as the intrinsic reward every step with p_{enc} and $q(z)$ fixed:

$$r^t_{in} = D_{\text{KL}}\left[p_{\text{enc}}(z^t|s^t, m^{t-1})|q(z^t)\right] \tag{7}$$

Eventually, instead of optimizing the Eq. (6), we derive the final objective for the high-level policy that we maximize:

$$\tilde{J}'(\theta_h) = \mathbb{E}_{\pi^h_{\theta_h}}\Big[\sum_{i=0,k,\cdots}\gamma^i_h(r^{t+i}_h + \beta_r\sum_{j=0,1,\cdots,k} r^j_{in}) \\ - \beta D_{\text{KL}}\left[p_{\text{enc}}(z|s,m)|q(z|s)\right]\Big] \tag{8}$$

where β_r is a factor.

Even though the r_{in} and D_{KL} seem confusing since they are in the same form, we notice that r_{in}, as the reward signal, will give credits to both $p_{\text{enc}}(z|s,m)$ as well as $p_{\text{dec}}(a_h|s,z)$, while D_{KL} is only used to update p_{enc}. Moreover, they are actually for adaptive exploration [Kim *et al.*, 2019]. During the first phase, visited states at the early training steps can lead to similar high intrinsic rewards, thus contributing little to making different intrinsic rewards among states. Hence, the high-level policy explores a wide range of the state space. Meanwhile, the KL term makes effort to force the p_{enc} to decrease the influence of the low-level policy at these visited states, so that in the future, the intrinsic rewards here will be smaller and the states will be less visited. During the second phase, most states are stationary thus lead to similar low intrinsic rewards. However, once some states leads to non-stationarity and cause high intrinsic rewards, the overall policy will visit them more. It results in a narrow range of exploration with a focus on the non-stationary states.

5 Experiments

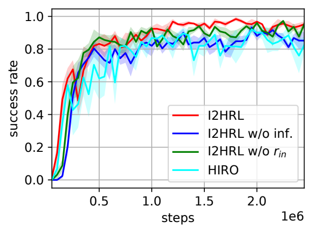
We design the experiments to answer the following questions: (1) How does I^{2}HRL compare against other end-to-end HRL algorithms? (2) Can influence-based framework stabilize the high-level policy training? (3) Is the proposed high-level exploration efficient under the non-stationrity?

5.1 Environmental Settings

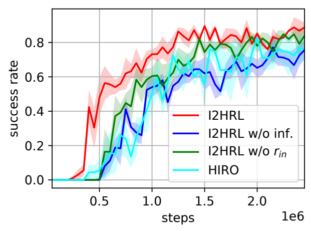
We evaluate and analyze our methods in the benchmarking hierarchical tasks [Duan *et al.*, 2016]. These environments were all simulated using the MujoCo physics engine for model-based control. The tasks are as follows:

Ant Gather. A quadrupedal ant is placed in a 20×20 space with 8 apples and 8 bombs. The agent receives a reward of $+1$ or collecting an apple and -1 for collecting a bomb; all other actions yield a reward of 0. Results are reported over the average of the past 100 episodes.

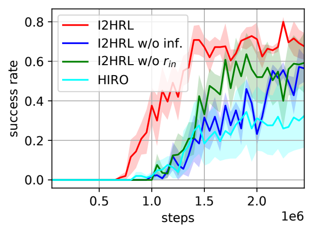
Ant Maze. The ant is placed in a U-shaped corridor and initialized at position $(0, 0)$. It is assigned an (x, y) goal-position that it is expected to reach (this (x, y) goal-position is only included in the state of the high-level policy). The agent is rewarded with its negative L_2 distance from current position to this goal position.



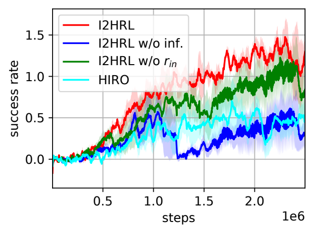
(a) AntMaze [16,0]



(b) AntMaze [16,16]



(c) AntMaze [0,16]



(d) AntGather

Figure 3: Ablation study for the proposed interaction, the influence-based framework and rewards. I^{2}HRL represents our method with Eq. (8); I^{2}HRL w/o inf represents that the internal variable z is removed, and the concatenation of the low-level policy representation with the state is feed into the high-level policy; I^{2}HRL w/o r_{in} represents the Eq. (6). HIRO is as a baseline.

Ant Push. Th ant would move forward, unknowingly pushing the movable block until it blocks its path to the target. To successfully reach the target, the ant must first move to the left around the block and then push the block right, clearing the path towards the target location

We choose three methods as baselines:

- Twin Delayed Deep Deterministic Policy Gradient (TD3) [Fujimoto *et al.*, 2018]: a state-of-the-art flat RL algorithm in continuous control domain to validate the need for hierarchical models to solve these tasks.
- HIRO [Nachum *et al.*, 2018]: a state-of-the-art HRL algorithm which utilizes the off-policy correction to address the non-stationarity problem.
- HAC [Levy *et al.*, 2019]: a state-of-the-art HRL algorithm which introduces hindsight action transitions to address the non-stationarity problem.

Results are reported over 5 random seeds of the simulator and the network initialization, and the time horizon is set to 500 steps. Both levels of I^{2}HRL utilize TD3. The low-level and high-level critic updates every single step and every 10 steps respectively. The low-level and high-level actor updates every 2 steps and every 20 steps respectively. We use Adam optimizer with learning rate of $3e-4$ for actor and critic of both levels of policies. We set the high-level policy decision interval k and the length of trajectories for low-level policy represent c as 10. Discount $\gamma = 0.99$, replay buffer size is $200,000$ for both levels of policies. The method-specific hyper-parameters (β and β_r) are fine-tuned for each tasks. All methods are well fine-tuned.

5.2 Results

Comparison with Baselines. In the Ant Gather, the results are the average external rewards of window size of 100. In the Ant Maze, the results are reported as success rates which are evaluated every 50,000 steps with the goal-position (16, 0), (16, 16), and (0, 16) respectively. The results, shown in Fig. 2 and 2f, demonstrate the training performance from scratch with I^{2}HRL and other baselines. In the Fig. 2b, all HRL methods reach a similar performance because it is easy for the ant to reach a goal which is directly in front of the ant. For the goal-position (16, 16) and (0, 16), agent is supposed to learn to change its direction. In the Fig. 2c, we can see that I^{2}HRL as well as HAC can reach the (16,16) earlier than HIRO, however, I^{2}HRL exceeds baselines by about 10% success rate when converged. In the Fig. 2d, which shows the success rate of algorithms in the most difficult task for ant, I^{2}HRL has a better performance than baselines on both the final performance and the convergence. In the AntGather and AntPush, I^{2}HRL also has a higher average rewards than other baselines. TD3 is non-hierarchical and not aimed for long-horizon sparse reward problems. Additionally, TD3 uses the Gaussian noise on the actions to explore, which serve as baseline exploration strategy. The success rates of TD3 in all maze tasks and the rewards in the gather task are almost zero. It shows the inability of flat RL algorithms on such tasks.

Ablations. To answer the question (2), we remove the internal variable z as well as the intrinsic reward, that is, directly feed the concatenation of the low-level policy representation and the state into the high-level policy. To answer the question (3), we only drop out the intrinsic reward r_{in}. In the Fig. 3, we notice that I^{2}HRL without influence-based framework has similar performance as compared with baseline. It demonstrates that the low-level policy representation can help the high-level policy alleviate the non-stationarity problem. However, we can also see that there are some drops and rebounds in both AntMaze and AntGather, like at the 2 million steps in Fig. (3c) and at 1.2 million steps in Fig. (3d). While I^{2}HRL without r_{in} does not have these drops and has better performance. As we mentioned in section 4.2, the source of non-stationarity moves from the low-level policy to the low-level representation. When the low-level policy representation is not sufficient or accurate to represent the low-level policy, the high-level policy makes bad decisions with bad representations. As the training process going, the representation becomes stable, thus the high-level policy starts to correct its understanding of the low-level representation. Also in the Fig. (3c), I^{2}HRL without r_{in} reaches a higher success rate from 1.5 million steps to 2 million steps than I^{2}HRL without the influence-based framework. It demonstrates the effectiveness of the proposed influence-based framework. The efficiency of proposed influence-based exploration is also shown in the Fig. (3). I^{2}HRL has better performance than I^{2}HRL without r_{in}, especially earlier to get the goal position and reach a higher success rate. It shows that r_{in} improves data efficiency, thus accelerating the learning process.

5.3 Visualization

Visualization of the low-level policy representation. We visualize the low-level policy representations in terms of different phases. We get the low-level policy representation (8 dimensions) every 100k steps over 3 million steps. Then we use t-SNE [Maaten and Hinton, 2008] to visualize the representations as shown in Fig. 4. At the beginning of the training process, the representation is sparse. While when the low-level policy is near-optimal, the representation is more dense.

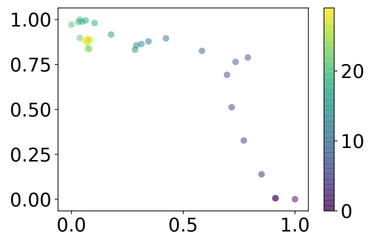
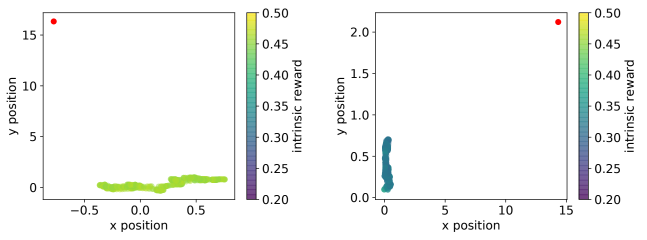
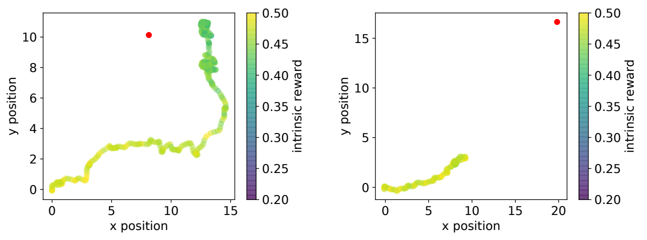


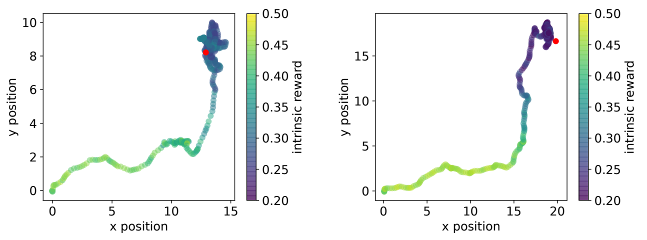
Figure 4: Visualization of the low-level policy representation.



(a) initial phase (b) initial phase



(c) non-stationary phase (d) non-stationary phase



(e) near-stationary phase (f) near-stationary phase

Figure 5: Visualization of the intrinsic reward. Axes represent the position of the ant. The red point is the goal position in AntMaze. Colormap shows the magnitude of the intrinsic rewards. (a)(b) The initial phase is at 0 step. (c)(d) The non-stationary phase is at 1 million steps. (e)(f) The near-stationary phase is at 3 million steps.

Visualization of non-stationary states. To backup our claims in Section 4.3, we visualize the intrinsic rewards in AntMaze with random target positions in terms of the different phases. Concretely, we split the training process into three phases: initial phase, non-stationary phase, and near-stationary phase. Since our method is end-to-end training, the intrinsic rewards are inaccurate in the initial phase. Consequently we pass the parameters of p_{enc} of the near-stationary phase to the parameters of the initial phase. At the beginning of the training process, the ant only navigates its surroundings with relatively high intrinsic rewards. When the low-level policy is near-optimal, the intrinsic rewards around goal positions are lower than rewards in the non-stationary phase.

As stated in Section 4.3, if the intrinsic reward is small, the agent visits a state which is less easily influenced by the low-level policy. It means that the high-level transition is near-stationary, i.e., less data is needed here other than other states. The result shows that once the ant learns how to approach a goal, the nearer it gets to the goal, the less data is needed. Because the task is goal-oriented, the ant actually needs to explore its surroundings when it does not know where to go, i.e. far away from its goals. For example, a well-trained adventurer is walking in a desert, looking for an oasis. When he is far away from the oasis, he will walk around and explore to determine a direction. When he is approaching to the oasis, he will rush his goal without hesitation.

6 Conclusion and Future Work

In this work, we focus on the non-stationarity problem in the end-to-end HRL. We propose the Interactive Influence-based HRL (I^{2}HRL). Concretely, we enable the interaction between the high-level policy and the low-level policy. We propose the influence-based framework to address the non-stationarity. This framework provides an influence-based adaptive exploration to help the agent to explore the more non-stationary states. I^{2}HRL outperforms state-of-the-art HRL baselines and accelerate the learning process.

I^{2}HRL splits the HRL into two parts: policy representation and communication. For the first one, more principles can be proposed to learn a good representation, such as predicting the low-level policy values (value-based), or predicting the state transitions and rewards (model-based). Various agent/opponent modeling methods can be future works for learning the low-level policy representation. For the second part, the literature of multi-agent communication can also improve the cooperation between the two levels in HRL, such as centralized training or value decomposition.

Acknowledgements

This research is supported AISG-RP-2019-0013, NSOE-TSS2019-01, NTU SUG “Choice Manipulation and Security Games” and NTU. We gratefully acknowledge the support of NVAITC (NVIDIA AI Tech Center) for our research.

References

[Alemi *et al.*, 2016] Alexander A Alemi, Ian Fischer, Joshua V Dillon, and Kevin Murphy. Deep variational information bottleneck. *arXiv preprint arXiv:1612.00410*, 2016.

[Bacon *et al.*, 2017] Pierre-Luc Bacon, Jean Harb, and Doina Precup. The option-critic architecture. In *AAAI Conference on Artificial Intelligence*, pages 1726–1734, 2017.

[Bellemare *et al.*, 2016] Marc Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Remi Munos. Unifying count-based exploration and intrinsic motivation. In *Advances in Neural Information Processing Systems*, pages 1471–1479, 2016.

[Cover and Thomas, 2012] Thomas M Cover and Joy A Thomas. *Elements of Information Theory*. John Wiley & Sons, 2012.

[Duan *et al.*, 2016] Yan Duan, Xi Chen, Rein Houthooft, John Schulman, and Pieter Abbeel. Benchmarking deep reinforcement learning for continuous control. In *International Conference on Machine Learning*, pages 1329–1338, 2016.

[Eysenbach *et al.*, 2018] Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. Diversity is all you need: Learning skills without a reward function. *arXiv preprint arXiv:1802.06070*, 2018.

[Florensa *et al.*, 2017] Carlos Florensa, Yan Duan, and Pieter Abbeel. Stochastic neural networks for hierarchical reinforcement learning. *arXiv preprint arXiv:1704.03012*, 2017.

[Fujimoto *et al.*, 2018] Scott Fujimoto, Herke van Hoof, and David Meger. Addressing function approximation error in actor-critic methods. *arXiv preprint arXiv:1802.09477*, 2018.

[Heess *et al.*, 2016] Nicolas Heess, Greg Wayne, Yuval Tassa, Timothy Lillicrap, Martin Riedmiller, and David Silver. Learning and transfer of modulated locomotor controllers. *arXiv preprint arXiv:1610.05182*, 2016.

[Kim *et al.*, 2019] Youngjin Kim, Wontae Nam, Hyunwoo Kim, Ji-Hoon Kim, and Gunhee Kim. Curiosity-bottleneck: Exploration by distilling task-specific novelty. In *International Conference on Machine Learning*, pages 3379–3388, 2019.

[Konidaris and Barto, 2009] George Konidaris and Andrew Barto. Efficient skill learning using abstraction selection. In *International Joint Conference on Artificial Intelligence*, pages 1107–1112, 2009.

[Kulkarni *et al.*, 2016] Tejas D Kulkarni, Karthik Narasimhan, Ardavan Saeedi, and Josh Tenenbaum. Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. In *Advances in Neural Information Processing Systems*, pages 3675–3683, 2016.

[Levy *et al.*, 2019] Andrew Levy, Robert Platt, and Kate Saenko. Hierarchical reinforcement learning with hindsight. In *International Conference on Learning Representations*, 2019. https://openreview.net/forum?id=ryzECoAcY7.

[Lillicrap *et al.*, 2015] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

[Maaten and Hinton, 2008] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(Nov):2579–2605, 2008.

[Mnih *et al.*, 2013] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[Nachum *et al.*, 2018] Ofir Nachum, Shixiang Shane Gu, Honglak Lee, and Sergey Levine. Data-efficient hierarchical reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 3303–3313, 2018.

[Nachum *et al.*, 2019] Ofir Nachum, Haoran Tang, Xingyu Lu, Shixiang Gu, Honglak Lee, and Sergey Levine. Why does hierarchy (sometimes) work so well in reinforcement learning? *arXiv preprint arXiv:1909.10618*, 2019.

[Papoudakis *et al.*, 2019] Georgios Papoudakis, Filippos Christianos, Arrasy Rahman, and Stefano V Albrecht. Dealing with non-stationarity in multi-agent deep reinforcement learning. *arXiv preprint arXiv:1906.04737*, 2019.

[Rabinowitz *et al.*, 2018] Neil C Rabinowitz, Frank Perbet, H Francis Song, Chiyuan Zhang, SM Eslami, and Matthew Botvinick. Machine theory of mind. *arXiv preprint arXiv:1802.07740*, 2018.

[Rafati and Noelle, 2019] Jacob Rafati and David C Noelle. Efficient exploration through intrinsic motivation learning for unsupervised subgoal discovery in model-free hierarchical reinforcement learning. *arXiv preprint arXiv:1911.10164*, 2019.

[Silver *et al.*, 2016] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484, 2016.

[Tishby and Zaslavsky, 2015] Naftali Tishby and Noga Zaslavsky. Deep learning and the information bottleneck principle. In *2015 IEEE Information Theory Workshop (ITW)*, pages 1–5, 2015.

[Vezhnevets *et al.*, 2017] Alexander Sasha Vezhnevets, Simon Osindero, Tom Schaul, Nicolas Heess, Max Jaderberg, David Silver, and Koray Kavukcuoglu. Feudal networks for hierarchical reinforcement learning. In *International Conference on Machine Learning*, pages 3540–3549, 2017.