State-Conditioned Adversarial Subgoal Generation

Vivienne Huiling Wang^{1,2}, **Joni Pajarinen**^{2}, **Tinghuai Wang**^{3}, **Joni-Kristian Kämäräinen**^{1}

^{1} Computing Sciences, Tampere University, Finland
^{2} Department of Electrical Engineering and Automation, Aalto University, Finland
^{3} Huawei Helsinki Research Center, Finland
{huiling.wang,joni.kamarainen}@tuni.fi, joni.pajarinen@aalto.fi, tinghuaiwang@huawei.com

Abstract

Hierarchical reinforcement learning (HRL) proposes to solve difficult tasks by performing decision-making and control at successively higher levels of temporal abstraction. However, off-policy HRL often suffers from the problem of a non-stationary high-level policy since the low-level policy is constantly changing. In this paper, we propose a novel HRL approach for mitigating the non-stationarity by adversarially enforcing the high-level policy to generate subgoals compatible with the current instantiation of the low-level policy. In practice, the adversarial learning is implemented by training a simple state conditioned discriminator network concurrently with the high-level policy which determines the compatibility level of subgoals. Comparison to state-of-the-art algorithms shows that our approach improves both learning efficiency and performance in challenging continuous control tasks.

Introduction

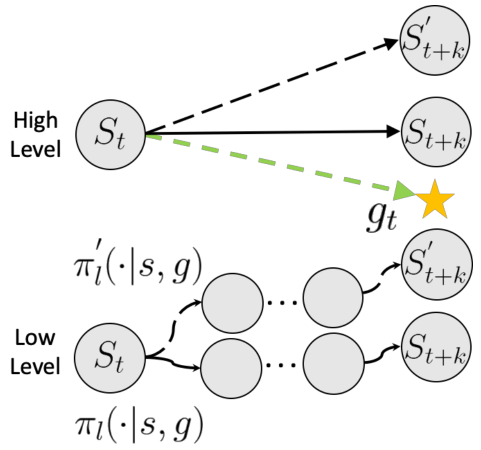
Hierarchical reinforcement learning (HRL), in which hierarchical policies learn to perform decision-making at successively higher levels of temporal and behavioral abstraction, has long held the promise to tackle complex problems with long-term credit assignment and sparse rewards. Among the prevailing HRL paradigms, the goal-conditioned HRL frameworks (Dayan and Hinton 1992; Schmidhuber and Wahnsiedler 1993; Kulkarni et al. 2016; Vezhnevets et al. 2017; Nachum et al. 2018; Levy et al. 2019; Zhang et al. 2020; Li et al. 2021) have achieved remarkable success. In goal-conditioned HRL, a high-level policy breaks the original task into a series of subgoals that a low-level policy is incentivized to reach. The effectiveness and efficiency of goal-conditioned HRL relies on reasonable and semantically meaningful subgoals providing a strong supervision signal to the low-level policy.

Nonetheless, off-policy training of a hierarchy of policies remains a key challenge due to the non-stationary state transitions induced by the hierarchical structure. Specifically, the same high-level action taken under the same state in the past may result in significantly different low-level state transitions due to the constantly changing low-level policy which renders the experience invalid for training. When all policies within the hierarchy are trained simultaneously, the high-level transition will constantly change as long as the low-level policy continues to be updated. However, learning hierarchical policies in parallel is still feasible as long as the high-level policy is able to efficiently adapt itself to the updated versions of low-level policy, and the hierarchical policy stabilizes once the low-level policy has converged to an optimal or near optimal policy. HIRO (Nachum et al. 2018) and HAC (Levy et al. 2019) have made attempts to address this problem by *relabeling* an experience in the past with a high-level action, *i.e.*, subgoal, to maximize the probability of the past lower-level actions. In other words, the high-level action which induced a low-level behavior in the past may be *relabeled* to a subgoal which is likely to induce the same low-level behavior with the current low-level policy. However, the relabeling approach does not facilitate efficient training of the high-level policy to comply with the update of low-level policy, which consistently generates incompatible subgoals and deteriorates the non-stationarity issue. Such unfit state transitions in off-policy training lead to improper learning of the high-level value function, therefore negatively affecting high-level policy exploration.

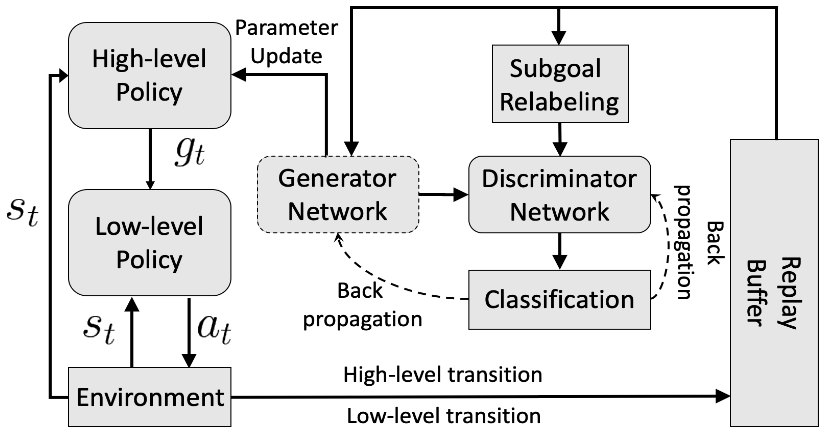
In this paper, we present a novel approach for mitigating the non-stationarity in goal-conditioned HRL. We aim to significantly improve the high-level policy's knowledge of the low-level's ability, thus improving the overall learning efficiency and stability. Concretely, we introduce an adversarial learning paradigm for HRL which enforces the high-level policy to learn to generate subgoals compatible with the current instantiation of the low-level policy. This is motivated by the assumption that the relabeled subgoals are sampled from a distribution which is asymptotically approximating an optimal high-level policy under stationary data distribution. Consequently the increasing divergence between the distribution of current subgoals and relabeled subgoals is the key indication of the non-stationarity. This suggests a conjecture that once this distribution divergence is mitigated the high-level policy naturally achieves stationarity.

To this end, we propose a discriminator network to distinguish a generated subgoal that may not be reachable by the low level policy from a relabeled subgoal that we know is reachable by the low level policy. The high-level policy plays the role of the generator network that learns to generate subgoals following a distribution compatible with the current low level policy.

The proposed adversarial learning thus reduces the shift and consequently the divergence in data distribution from relabeled experience to the current high-level policy behaviour and encourages the high level policy to generate reasonable



(a) The non-stationarity problem



(b) Overview of SAGA

Figure 1: (a) The emergence of the non-stationarity problem: since the low-level policy has changed from $\pi_l(\cdot|s,g)$ to $\pi_l^{'}(\cdot|s,g)$, a subgoal g_t generated by a certain high-level policy in the past may not yield the same low-level behavior given the current low-level policy and consequently renders the experience invalid for training. (b) Overview of SAGA: the high-level policy generates high-level actions *i.e.*, subgoals every k time steps to guide the low-level policy which interacts with the environment. Off-policy adversarial learning is performed for high-level policy to improve its stability and sample efficiency with relabeled subgoals.

subgoals. Fitting to state transitions with compatible high-level actions effectively improves the accuracy of the high-level value function and enhances its subsequent exploration underpinning a stationary hierarchical model.

Preliminaries

In reinforcement learning, the interaction between agent and environment is modeled as a Markov Decision Process (MDP) $M =< \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma >$, where $\mathcal{S}$ is a state space, $\mathcal{A}$ is an action set, $\mathcal{P}: \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0,1]$ is a state transition function, $\mathcal{R}: \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is a reward function, and $\gamma \in [0,1)$ is a discount factor. A stochastic policy $\pi(a|s)$ maps a given state s to a probability distribution over actions $\pi: \mathcal{S} \to \mathcal{A}$. The objective of the agent is to maximize the expected cumulative discounted reward $\mathbb{E}_\pi[\sum_{t=0}^{\infty} \gamma^t r_t]$, where r_t is the obtained reward at the discrete time step t.

Two-Layer HRL Framework: We adopt a continuous control RL setting, modeled as a finite-horizon, goal-conditioned MDP $M =< \mathcal{S}, \mathcal{G}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma >$, where $\mathcal{G}$ is a goal set. We consider a HRL framework comprising two hierarchies following (Nachum et al. 2018) with a high-level policy $\pi_h(g|s)$ and a low-level policy $\pi_l(a|s,g)$. High-level policy operates at a coarser layer and generates a high-level action, *i.e.*, subgoal $g_t \sim \pi_h(\cdot|s_t) \in \mathcal{G}$, every k timesteps when $t \equiv 0 \pmod{k}$. A pre-defined goal transition function $g_t = f(g_{t-1}, s_{t-1}, s_t)$ is utilized when $t \not\equiv 0 \pmod{k}$. The high level modulates the behavior of the low-level policy by intrinsic rewards for reaching these subgoals. Following prior work (Andrychowicz et al. 2017; Nachum et al. 2018; Zhang et al. 2020), the goal set $\mathcal{G}$ corresponds to a subset of state space, *i.e.*, $\mathcal{G} \subset \mathcal{S}$, and the goal transition function is defined as $f(g_{t-1}, s_{t-1}, s_t) = s_{t-1} + g_{t-1} - s_t$. The high-level policy aims to maximize the extrinsic reward r_{kt}^h defined as:

$$r_t^h = \sum_{i=t}^{t+k-1} r_i^{\text{env}}, t = 0, 1, 2, \cdots \tag{1}$$

where r_i^{env} is the reward from the environment.

The low-level policy aims to maximize the intrinsic reward provided by the high-level policy. It takes the high-level action or subgoal g as input, and interacts with the environment every timestep by taking an action $a_t \sim \pi_l(\cdot|s_t, g_t) \in \mathcal{A}$. To encourage the low-level policy to reach the subgoal g, an intrinsic reward function measuring the subgoal-reaching performance is adopted $r_t^l = -||s_t + g_t - s_{t+1}||_2$.

The above goal-conditioned HRL framework allows the low-level policy to receive learning signals even before achieving a certain goal-reaching capability and enables concurrent end-to-end training of the high-level and low-level policies. However, off-policy training of the above HRL framework suffers from the non-stationarity problem of the high-level policy as mentioned in Section 1. HIRO (Nachum et al. 2018) proposes to relabel the high-level transition $(s_t, g_t, \sum_{i=t}^{t+k-1} r_i^{\text{env}}, s_{t+k})$ with a different subgoal $\tilde{g}_t$ to make the actual observed low-level action sequence more likely to have happened with respect to the current low-level policy by maximizing $\pi_l(a_{t:t+k-1}|s_{t:t+k-1}, \tilde{g}_{t:t+k-1})$.

State-Conditioned Adversarial Subgoal Generation

In this section, we present our *State-conditioned Adversarial subGoal generAtion for hierarchical learning* (SAGA), an adversarial learning approach guiding the high-level policy generating more reachable subgoals for low-level policy. The non-stationarity in the previous HRL methods leads to unstable and inefficient high-level policy training. In this section, we introduce our adversarial learning approach to significantly improve the sample efficiency and overall performance of off-policy training of the high-level policy.

Adversarial Learning of High-Level Policy: SAGA integrates adversarial learning and policy training in a two-player game similarly to Generative Adversarial Networks (GANs) (Goodfellow et al. 2014), which primarily comprises a subgoal generator network $G(\mathbf{s};\theta_g):\mathbf{s}\rightarrow\mathbf{g}$ and a subgoal discriminator network $D(\mathbf{g}|\mathbf{s};\theta_d)\rightarrow\{0,1\}$. As opposed to the generator defined in GAN which samples from a noise distribution, our subgoal generator network $G(\mathbf{s};\theta_g)$ maps from state space to subgoal space; our subgoal discriminator network is conditioned on state $\mathbf{s}$ reminiscent of conditional GAN (Mirza and Osindero 2014) where, in contrast, both its generator and discriminator are conditioned on class labels. In order to mitigate the non-stationary issue, we aim to reduce the divergence between the data distribution of the relabeled experience and the current high-level policy behaviour, with the assumption that subgoals in the relabeled experience are "optimal" in learning a stationary hierarchical model. To this end, the subgoal discriminator tries to distinguish the generated subgoals from relabeled subgoals of the replay buffer. In practice, we let the subgoal generator $G(\mathbf{s};\theta_g)$ be a surrogate of the high-level actor network.

Although our approach is applicable to general actor-critic based HRL algorithms, we adopt the TD3 (Fujimoto, Hoof, and Meger 2018) algorithm for each level in the HRL structure following HIRO (Nachum et al. 2018) and HRAC (Zhang et al. 2020). Thus the first objective of the subgoal generator is to maximize the expected return induced by a deterministic policy:

$$J_{\text{dpg}}=\mathbb{E}_{\mathbf{s}\sim\mathcal{D}}[Q_h(\mathbf{s},\mathbf{g})|_{\mathbf{g}=G(\mathbf{s};\theta_g)}] \tag{2}$$

where $\mathcal{D}$ is the replay buffer with the high level action relabeled similarly to HIRO, *i.e.*, relabeling g_t of the high level transition $(s_t,g_t,\sum_{i=t}^{t+k-1}r_i^{\text{env}},s_{t+k})$ with $\tilde{g}_t$ to maximize the probability of incurred low-level action sequence $\pi_l(a_{t:t+k-1}|s_{t:t+k-1},\tilde{g}_{t:t+k-1})$, which is approximated by maximizing the log probability

$$\log\pi_l(a_{t:t+k-1}|s_{t:t+k-1},\tilde{g}_{t:t+k-1})\propto -\frac{1}{2}\sum_{i=t}^{t+k-1}||a_i-\pi_l(s_i,\tilde{g}_i)||_2^2+\text{const.} \tag{3}$$

In order to learn the distribution of the subgoal generator $G(\mathbf{s};\theta_g)$ over the relabeled subgoal $\tilde{g}$ through adversarial learning, we define a subgoal discriminator network $D(\mathbf{g}|\mathbf{s};\theta_d)$ which outputs the probability that subgoal g is an "optimal" subgoal, *i.e.*, the relabeled subgoals rather than a subgoal sampled from the generator's distribution $G(\mathbf{s})$. That is, we train $D(\mathbf{g}|\mathbf{s};\theta_d)$ to maximize the probability of distinguishing the data distribution of "optimal" and "sub-optimal" subgoals. Simultaneously we train $G(\mathbf{s};\theta_g)$ to minimize the probability that a generated subgoal is classified as a "sub-optimal" subgoal by the discriminator network, that is, we minimize $\log(1-D(G(\mathbf{s})|\mathbf{s}))$:

$$J_{\text{adv}}=\min_G\max_D V(D,G)=\mathbb{E}_{\mathbf{s},\mathbf{g}\sim\mathcal{D}}[\log D(\mathbf{g}|\mathbf{s})]+\mathbb{E}_{\mathbf{s}\sim\mathcal{D}}[\log(1-D(G(\mathbf{s})|\mathbf{s}))]. \tag{4}$$

Combining terms defined in Eq. (2) and Eq. (4), the high-level actor *i.e.*, subgoal generator $G(\mathbf{s};\theta_g)$ is learned by performing gradient update on parameter θ_g

$$\nabla_{\theta_g}J=\mathbb{E}_{\mathbf{s}\sim\mathcal{D}}[\nabla_{\theta_g}G(\mathbf{s})\,\nabla_{\mathbf{g}}Q_h(\mathbf{s},\mathbf{g})|_{\mathbf{g}=G(\mathbf{s})}]-\alpha_{\text{adv}}\mathbb{E}_{\mathbf{s}\sim\mathcal{D}}[\nabla_{\theta_g}\log(1-D(G(\mathbf{s})|\mathbf{s})], \tag{5}$$

where α_{adv} is a hyperparameter to weigh the adversarial loss.

The subgoal discriminator is learned by updating θ_d with gradient

$$\nabla_{\theta_d}J_{\text{adv}}=\mathbb{E}_{\mathbf{s},\mathbf{g}\sim\mathcal{D}}[\nabla_{\theta_d}[\log D(\mathbf{g}|\mathbf{s})+\log(1-D(G(\mathbf{s})|\mathbf{s})]].$$

Note that SAGA is not enforcing the high level to generate the exact relabeled actions, but rather encourages the high-level action to follow a similar data distribution with the one which is compatible with the current instantiation of the low-level policy regardless of its effectiveness. Consequently, the approach attempts to stabilize the hierarchical learning with minimum risk of hurting exploration.

Related Work

HRL (Dayan and Hinton 1992; Schmidhuber and Wahnsiedler 1993; Kulkarni et al. 2016; Vezhnevets et al. 2017; Nachum et al. 2018; Levy et al. 2019; Zhang et al. 2020; Li et al. 2021) has long held the promise to tackle long-term credit assignment and sparse reward problems, where the high-level policy decomposes the task into subtasks whilst the low-level policy learns how to efficiently solve these subtasks. The specific way of this decomposition, *i.e.*, how exactly the high level communicates with the low level, varies in different approaches. Various forms of signals from the high level have been proposed, ranging from using discrete value for option (Bacon, Harb, and Precup 2017; Fox et al. 2017; Gregor, Rezende, and Wierstra 2017) or skill (Konidaris and Barto 2009; Eysenbach et al. 2019; Sharma et al. 2020; Bagaria and Konidaris 2019) selection, to forming a vector within a learned embedding space as subgoal (Vezhnevets et al. 2017; Li et al. 2021). However, majority of these approaches are unable to benefit from advances of off-policy model-free RL.

Improving the learning efficiency of HRL through off-policy training has attracted a considerable amount of research efforts in recent years. However, besides instability, off-policy training also poses the non-stationary problem which is characteristic to HRL. In (Nachum et al. 2018), they proposed an off-policy method which relabels past experiences to reduce the impact in training with invalid high-level state transitions due to non-stationarity. Employing hindsight techniques (Andrychowicz et al. 2017), (Levy et al. 2019) proposed to train multi-level policies in parallel while penalizing the high-level for generating subgoals which are not reachable in the low level. In (Zhang et al. 2020) the large subgoal space issue was addressed by restricting the high-level action space from the whole subgoal space using an adjacency constraint. In (Wang et al. 2020) high-level policy decision making is conditioned on the received low-level policy representation as well as the state of the environment to improve stationarity. Another solution is to add a slowness objective to effectively learn the subgoal representation so that the low-level reward function varies in a stationary way (Li et al. 2021).

The general topic of goal generation in RL has also been studied (Florensa et al. 2018; Nair et al. 2018; Ren et al. 2019;

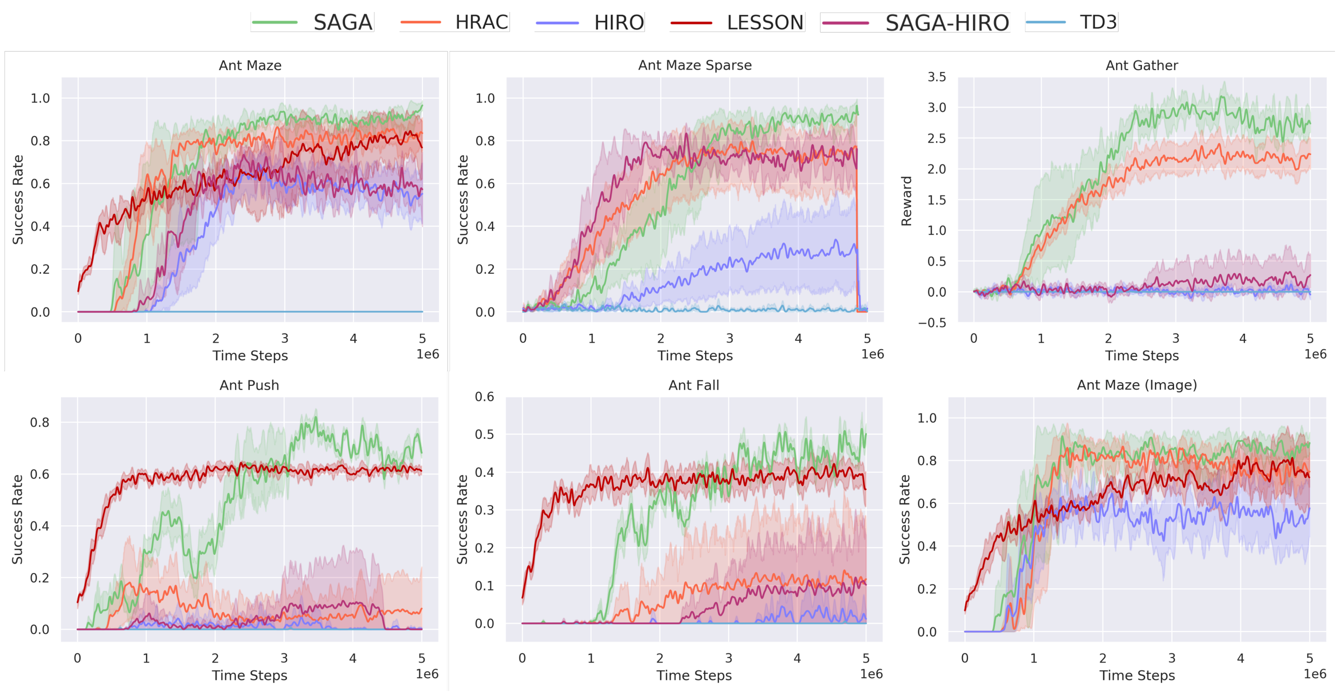


Figure 2: Learning curves of SAGA and baselines on all environments. Each curve and its shaded region represent average episode reward (for Ant Gather) or average success rate (for the rest; see the appendix) and 95% confidence interval respectively, averaged over 10 independent trials. We find that SAGA performs well across all tasks. It is worth noting that SAGA learns rapidly; on the complex navigation tasks it normally requires only less than three million environment steps to achieve good performance.

Campero et al. 2021). GoalGAN (Florensa et al. 2018) uses a standard GAN to produce tasks at the appropriate level of difficulty for training the policy. While GoalGAN is similar in spirit with the proposed SAGA to some extent, there are several key differences apart from if it is a hierarchical policy or not. GoalGAN is using a standalone generator that does not condition on the observation; its GAN and policy are two modules that are independently and sequentially trained. In contrast, SAGA's generator is a surrogate of the original actor network and SAGA directly updates its policy through the incurred adversarial loss and policy loss concurrently. Other related works are, for example, (Nair et al. 2018) that proposes to combine unsupervised representation learning and reinforcement learning of goal-conditioned policies. A framework to generate hindsight goals which are easy for an agent to achieve in the short term is proposed in (Ren et al. 2019). In a recent work (Campero et al. 2021) a framework where a teacher network learns to propose increasingly challenging yet achievable goals is proposed; the teacher is positively rewarded if the student achieves the goal with suitable effort, but penalized if the student either cannot achieve the goal, or can do so too easily. The foremost difference from SAGA is that these methods are developed for flat architectures and therefore cannot successfully solve tasks which require complex high-level decision making.

Experiments

This section evaluates and compares our method against standard RL and prior HRL methods in challenging environments which require a combination of locomotion and object manipulation. We also ablate the various components to understand their importance. Our experiments are designed to answer the following questions:

1. Can SAGA improve the sample efficiency and performance of goal-conditioned HRL across various long-horizon continuous control tasks?
2. Can SAGA outperform alternative adversarial learning approaches in the goal-conditioned HRL framework?

Environment Setup

We consider the following five environments for our analysis:

1. **Ant Maze**: A '⊃'-shaped maze poses a challenging navigation task for a quadruped-Ant. The ant needs to reach a target position starting from a random position in a maze with dense rewards. A variant (labeled 'Image') with low-resolution image observations is also adopted; the observation is formed by zeroing out the x, y coordinates and appending a $5\times5\times3$ top-down view of the environment, as described in (Nachum et al. 2019).
2. **Ant Maze Sparse**: From a random start position, the ant needs to reach a target position in a maze with sparse rewards.
3. **Ant Gather**: Starting from a fixed position, the ant collects green apples and avoids red bombs.
4. **Ant Push**: A challenging environment which requires both task and motion planning. The ant needs to move to

	Ant Maze	Ant Maze Sparse	Ant Gather	Ant Push	Ant Fall	Ant Maze (Image)
SAGA	**0.93±0.01**	**0.92±0.01**	**2.72±0.07**	**0.72±0.02**	**0.47± 0.02**	**0.87± 0.02**
HRAC	0.83±0.03	0.75±0.08	2.19±0.34	0.06±0.06	0.11 ±0.07	0.76 ±0.03
HIRO	0.54±0.06	0.29±0.10	0.02±0.02	0.00±0.00	0.01± 0.01	0.53± 0.06
LESSON	0.81±0.04	-	-	0.62± 0.02	0.38± 0.01	0.74± 0.06
TD3	0.0±0.0	0.01±0.0	0.0±0.0	0.0±0.0	0.0±0.0	-
w/o state for D	0.91±0.02	0.88±0.01	2.62±0.06	0.58±0.03	0.39± 0.03	-
w/o state for D/G	0.0±0.0	0.19±0.01	0.0±0.0	0.0±0.0	0.0±0.0	-
SAGA-HIRO	0.58±0.03	0.72±0.04	0.18±0.15	0.0±0.0	0.11± 0.07	-

Table 1: Final performance of the policy obtained after 5M steps of training, averaged over 10 randomly seeded trials with standard error.

the left of the maze so that it can move up and right to push the block out of the way for reaching the target.

5. **Ant Fall**: This environment extends the navigation to three dimensions. The ant starts on a raised platform with the target located directly in front of it but separated by a chasm which it cannot cross by itself. The ant needs to push the block forward, fill the gap, walk across and move to the left in order to reach the target.

Implementation

For the hierarchical policy network, we employ the same architecture as HRAC (Zhang et al. 2020) which adopts TD3 (Fujimoto, Hoof, and Meger 2018) as the underlying algorithm for training both the high-level and low-level policy. Specifically, we adopt two networks comprising three fully-connected layers with ReLU nonlinearities as the actor and critic networks of both low-level and high-level TD3 networks. The size of the hidden layers of both actor and critic is 300. The output of the high-level actor is activated using the tanh function and scaled according to the size of the environments.

The subgoal generator network has the identical architecture as the high-level actor. For the subgoal discriminator network, we use a network consisting of 3 fully-connected layers (size of 300, 300 and 1 respectively) with Leaky-ReLU (negative slope 0.2) nonlinearities and sigmoid function in all tasks. Adam optimizer is used for all networks.

Comparative Analysis

To test the performance of SAGA, we compare against the following baseline methods:

1. **HIRO** (Nachum et al. 2018): an off-policy goal-conditioned HRL algorithm proposes to address the non-stationarity issue by relabeling high-level actions.
2. **HRAC** (Zhang et al. 2020): a state-of-the-art off-policy goal-conditioned HRL algorithm introduces an adjacency network to restrict the high-level action space to a k-step adjacent region of the current state[1]
3. **LESSON** (Li et al. 2021): a state-of-the-art off-policy goal-conditioned HRL algorithm learns the subgoal representation by posing a slowness objective.
4. **TD3** (Fujimoto, Hoof, and Meger 2018): a state-of-the-art flat RL algorithm we compare to validate the need for hierarchical policies.

For fair comparison, all the HRL baselines use the same hierarchical structure and environment configuration as SAGA. Table 1 shows the final performance of the policy after training, and all baselines are significantly outperformed by SAGA. The learning curves of SAGA and baselines across all tasks are plotted in Fig. 2. In the gather task *i.e.*, Ant Gather, SAGA achieves considerably better performance and consistently exceeds all baselines in gather task *i.e.*, Ant Gather, and all navigation tasks *i.e.*, Ant Maze, Ant Maze (Image), Ant Maze Sparse, Ant Push and Ant Fall[2], in terms of sample efficiency and asymptotic performance. SAGA shows consistently better training efficiency benefiting from the improved learning stability of the hierarchical policies. This suggests that the proposed adversarial learning approach effectively enforces the high-level policy to generate subgoals compatible with the current instantiation of the low-level policy during training which in turn significantly mitigates the non-stationarity issue of off-policy training in HRL. It is also observed that flat RL algorithm TD3 does not learn in the complex environments used in the experiments which further validate the need for hierarchical policies.

Ablative Analysis

We also compare SAGA with several variants to investigate the importance of various design choices of SAGA. As we employ HRAC as the base model for SAGA, we introduce a variant of SAGA which is employing HIRO as base model to understand the generalization of our proposed approach. Fig. 2 also shows the learning curves of SAGA-HIRO and HIRO. In the Ant Maze task with dense rewards, SAGA-HIRO achieves slightly better performance with HIRO, while SAGA-HIRO exceeds HIRO in other tasks. Table 1 shows the consistent quantitative improvements by introducing the proposed approach. We note that HIRO hardly learns in Ant Push and learns poorly in Ant Fall by using the standard RL training based on the relabeled high-level actions, whereas SAGA-HIRO enforces the policy to learn in a more sample efficient manner. The empirical evaluation confirms that our algorithm is a principled and generic approach which can be applied in existing goal-conditioned HRL methods to effectively address the non-stationarity issue.

[1]We use HRAC's official implementation https://github.com/trzhang0116/HRAC to evaluate both HRAC and HIRO since HRAC is built on HIRO which provides fair comparisons using the same HRL implementation.

[2]We use LESSON's official implementation https://github.com/SiyuanLee/LESSON which includes environments Ant Maze, Ant Push and Ant Fall and adapt its task settings to be the same with other baselines for fair comparisons

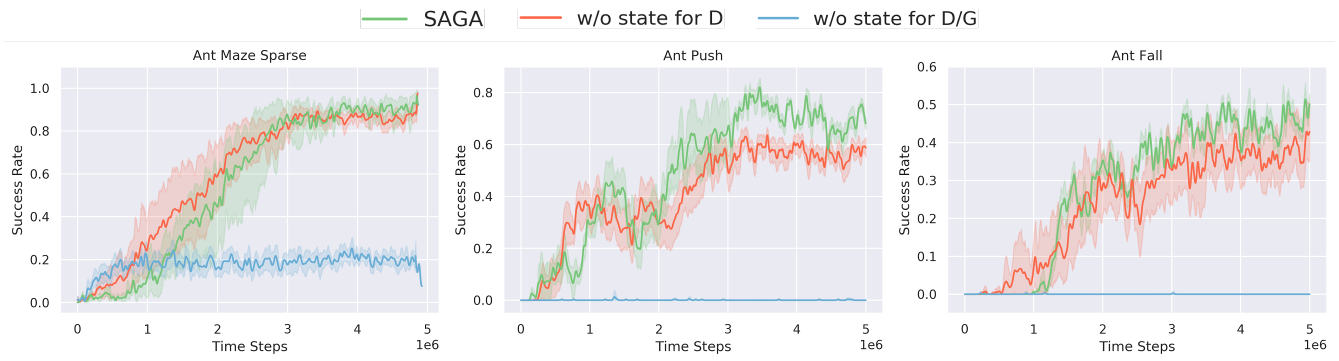


Figure 3: Ablation studies of adversarial learning approaches, averaged over 10 independent trials.

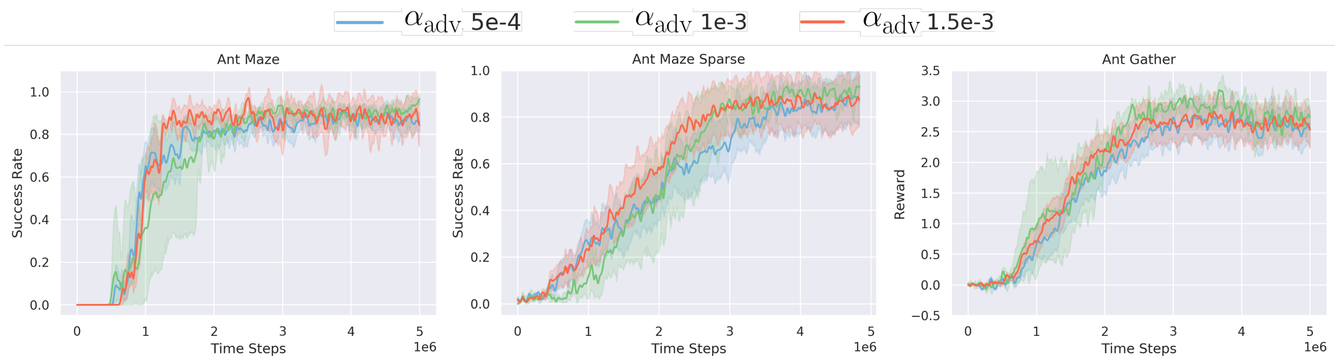


Figure 4: Learning curves with different coefficient of adversarial loss α_{adv}, averaged over 5 independent trials.

In the ablation studies of alternative adversarial learning approaches in the goal-conditioned HRL framework, we introduce two variants:

1. **w/o state for D**: a variant that uses a discriminator network which is not conditioned on state;
2. **w/o state for D/G**: a variant that uses common generator and discriminator networks which neither are conditioned on state, *i.e.*, the generator network takes as input a random noise sampled from Normal distribution and then trains the high-level critic using the generated subgoals, similar to (Florensa et al. 2018) in flat RL case.

As shown in Table 1 and Fig. 3, the advantage of the discriminator network conditioning on states is more pronounced in challenging tasks Ant Push and Ant Fall, where subgoal distributions may heavily depend on the states. In other words, state-conditioned discriminator network is able to account for the dynamic elements in the environment and subsequently enables subgoals that specifically interact with those elements. In the second variant, *i.e.*, a vanilla GAN setting, the generator network is no longer a surrogate of the high-level actor network. The assumption that both generator and discriminator are not depending on states leads to slow learning of generator network and consequently the deteriorating non-stationarity issue for the hierarchical policies.

Analysis of Hyperparameter Selection

We empirically study the effect of different coefficients of adversarial loss α_{adv}. Fig. 4 shows that SAGA with three coefficients of adversarial loss 0.0005, 0.001 and 0.0015 shows asymptotically similar results and generally $\alpha_{\text{adv}} = 0.001$ gives better performance across three tasks; we use $\alpha_{\text{adv}} = 0.001$ for all the tasks presented in the paper. In general, larger α_{adv} implies that the learning prioritizes the adaptation of the generated subgoals to follow the distribution of relabeled subgoals which speeds up its learning process harnessing the more accurate learning signals provided. However, the learning process may slow down in case of very large α_{adv} since the learning of the high-level value function will be slowed down and its subsequent exploration might be affected. As a contrary, with smaller α_{adv} the standard RL training is more pronounced; SAGA eventually degenerates to original baseline when its value is considerably small.

Analysis of Generated Subgoals

We visualize the generated subgoals of SAGA and the reached subgoals *i.e.*, the final state of k-step low-level roll-out in Fig. 5. The subgoals generated by SAGA are generally reachable and match the low-level trajectories, since they are distributed in the close vicinity of the actually reached subgoals. This is further confirmed by the measure of distance between generated subgoals and the reached subgoals *i.e.*, the final state of k-step low-level roll-out in Table 2. This suggests that SAGA generates the most reachable subgoals and may give the most effective learning signal for the low level to improve sample efficiency compared with HRAC and HIRO.

We also visually compare the generated subgoals of SAGA,

	Ant Maze	Ant Maze Sparse	Ant Gather	Ant Push	Ant Fall
SAGA	**2.79±0.08**	**2.13±0.26**	**1.65±0.52**	**2.62±0.46**	**2.49± 0.23**
HRAC	3.41±0.31	4.38±0.84	3.59±0.91	4.95±1.11	4.10 ±1.03
HIRO	10.74±1.05	14.14±0.0	13.71±1.15	9.38±2.85	11.4± 1.69

Table 2: The distance between generated subgoals and the reached subgoals *i.e*., the final state of k-step low-level roll-out, averaged over 10 randomly seeded trials with standard error.

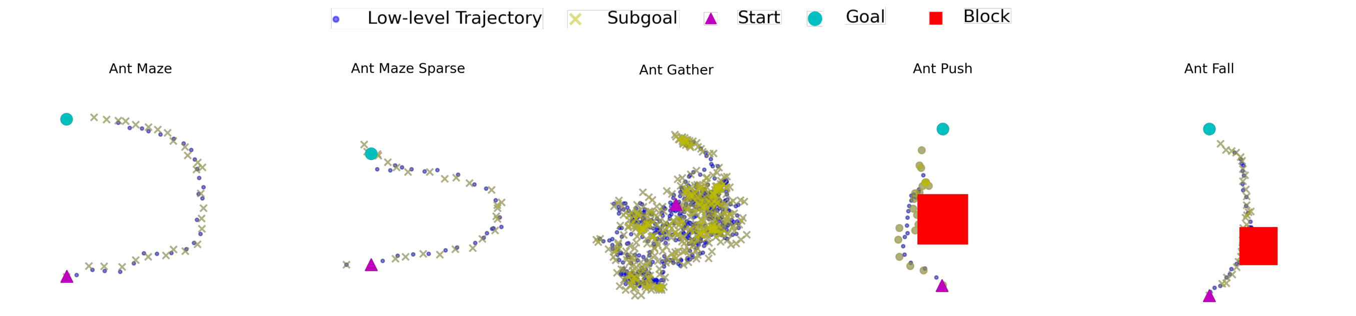


Figure 5: Visualization of generated subgoals by SAGA and the reached subgoals *i.e*., the final state of k-step low-level roll-out, in one of the randomly seeded trials. The generated subgoals are reachable and generally match the low-level trajectories.

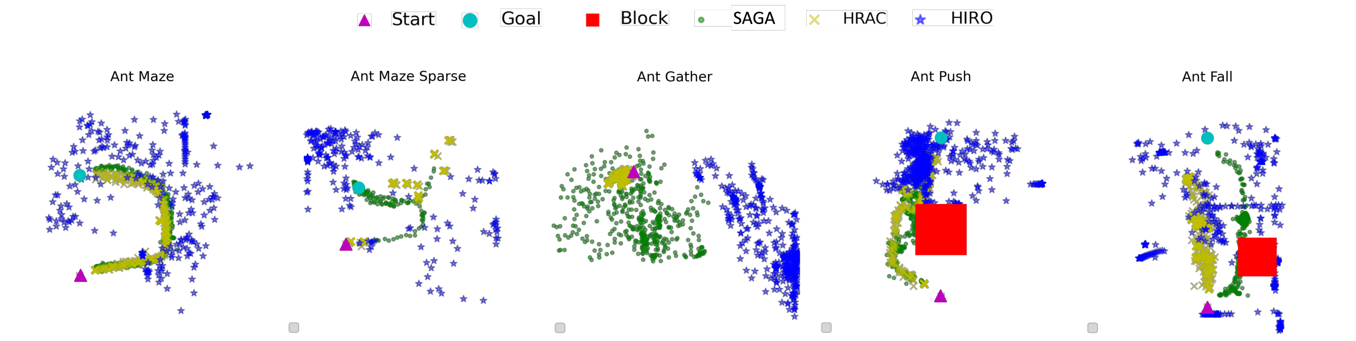


Figure 6: Visualization of generated subgoals. Subgoals generated by SAGA are generally matching the low-level trajectories or planed motions, which indicates that SAGA can generate reasonable subgoals for the low level to achieve and also guide the optimization to jump out of the local optimum of the ant to move directly towards the target in complex tasks such as Ant Push and Ant Fall. In contrast, subgoals generated by HRAC and HIRO frequently get stuck to local minimum and fail to guide the agent to accomplish the final task. Subgoals generated by LESSON lie in a learned subgoal representation space and may not be visualized along with other methods.

HIRO and HRAC in Fig. 6. The subgoals generated by SAGA generally match the planned motions in Ant Maze, Ant Maze Sparse and Ant Gather. Notably, SAGA generates subgoals to guide the optimization to jump out of the local optimum of the ant to move directly towards the target in Ant Push and Ant Fall. In detail, under the hierarchical policy trained by SAGA, in Ant Push the ant first moves to the left, then pushes the block to the right and finally reaches the target; guided by subgoals generated by SAGA, in Ant Fall, the ant first moves to the right to push the block forward which fills the gap and then walks across and moves to the left in order to finally reach the target. HRAC can also generate relatively reasonable subgoals to reach based on affinity constraints for the low level, however these subgoals frequently get stuck in a local optimum of moving directly to the target as illustrated in Ant Push and Ant Fall. HIRO, on the contrary, fails to generate achievable subgoals and cannot guide the agent to achieve its final target.

Conclusion

We proposed a novel adversarially guided subgoal generation framework for goal-conditioned HRL to mitigate the issue of non-stationarity in off-policy training. The learning of high-level policy is formulated as a two-player game where the subgoal generator endeavours to generate subgoals compatible with the current instantiation of low-level policy while the proposed discriminator network tries to distinguish the generated subgoals from the relabled subgoals. Empirical studies show that the proposed adversarial learning is capable of reducing the shifts in data distribution from relabeled experience to the current high-level policy behaviour and consequently improving the overall learning efficiency and stability.

References

Andrychowicz, M.; Crow, D.; Ray, A.; Schneider, J.; Fong, R.; Welinder, P.; McGrew, B.; Tobin, J.; Abbeel, P.; and Zaremba, W. 2017. Hindsight Experience Replay. In *Advances in Neural Information Processing Systems*, 5048–5058.

Bacon, P.-L.; Harb, J.; and Precup, D. 2017. The option-critic architecture. In *The AAAI Conference on Artificial Intelligence*, volume 31.

Bagaria, A.; and Konidaris, G. 2019. Option discovery using deep skill chaining. In *International Conference on Learning Representations*.

Campero, A.; Raileanu, R.; Küttler, H.; Tenenbaum, J. B.; Rocktäschel, T.; and Grefenstette, E. 2021. Learning with AMIGo: Adversarially Motivated Intrinsic Goals. In *International Conference on Learning Representations*.

Dayan, P.; and Hinton, G. E. 1992. Feudal Reinforcement Learning. In *Advances in Neural Information Processing Systems*, 271–278.

Eysenbach, B.; Gupta, A.; Ibarz, J.; and Levine, S. 2019. Diversity is All You Need: Learning Skills without a Reward Function. In *International Conference on Learning Representations (Poster)*.

Florensa, C.; Held, D.; Geng, X.; and Abbeel, P. 2018. Automatic Goal Generation for Reinforcement Learning Agents. In *International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, 1514–1523. PMLR.

Fox, R.; Krishnan, S.; Stoica, I.; and Goldberg, K. 2017. Multi-Level Discovery of Deep Options. *CoRR*, abs/1703.08294.

Fujimoto, S.; Hoof, H.; and Meger, D. 2018. Addressing function approximation error in actor-critic methods. In *International Conference on Machine Learning*, 1587–1596. PMLR.

Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; and Bengio, Y. 2014. Generative adversarial nets. *Advances in neural information processing systems*, 27.

Gregor, K.; Rezende, D. J.; and Wierstra, D. 2017. Variational Intrinsic Control. In *International Conference on Learning Representations (Workshop)*.

Konidaris, G.; and Barto, A. 2009. Efficient skill learning using abstraction selection. In *International Joint Conference on Artificial Intelligence*.

Kulkarni, T. D.; Narasimhan, K.; Saeedi, A.; and Tenenbaum, J. 2016. Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. *Advances in neural information processing systems*, 29: 3675–3683.

Levy, A.; Konidaris, G. D.; Jr., R. P.; and Saenko, K. 2019. Learning Multi-Level Hierarchies with Hindsight. In *International Conference on Learning Representations*.

Li, S.; Zheng, L.; Wang, J.; and Zhang, C. 2021. Learning Subgoal Representations with Slow Dynamics. In *International Conference on Learning Representations*.

Mirza, M.; and Osindero, S. 2014. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*.

Nachum, O.; Gu, S.; Lee, H.; and Levine, S. 2018. Data-Efficient Hierarchical Reinforcement Learning. In *Advances in Neural Information Processing Systems*, 3307–3317.

Nachum, O.; Gu, S.; Lee, H.; and Levine, S. 2019. Near-Optimal Representation Learning for Hierarchical Reinforcement Learning. In *International Conference on Learning Representations*.

Nair, A.; Pong, V.; Dalal, M.; Bahl, S.; Lin, S.; and Levine, S. 2018. Visual Reinforcement Learning with Imagined Goals. In *Advances in Neural Information Processing Systems*, 9209–9220.

Ren, Z.; Dong, K.; Zhou, Y.; Liu, Q.; and Peng, J. 2019. Exploration via Hindsight Goal Generation. In *Advances in Neural Information Processing Systems*, 13464–13474.

Schmidhuber, J.; and Wahnsiedler, R. 1993. Planning simple trajectories using neural subgoal generators. In *From Animals to Animats 2: Proceedings of the Second International Conference on Simulation of Adaptive Behavior*, volume 2, 196. MIT Press.

Sharma, A.; Gu, S.; Levine, S.; Kumar, V.; and Hausman, K. 2020. Dynamics-Aware Unsupervised Discovery of Skills. In *International Conference on Learning Representations*.

Vezhnevets, A. S.; Osindero, S.; Schaul, T.; Heess, N.; Jaderberg, M.; Silver, D.; and Kavukcuoglu, K. 2017. Feudal networks for hierarchical reinforcement learning. In *International Conference on Machine Learning*, 3540–3549. PMLR.

Wang, R.; Yu, R.; An, B.; and Rabinovich, Z. 2020. I2HRL: Interactive Influence-based Hierarchical Reinforcement Learning. In *International Joint Conference on Artificial Intelligence*, 3131–3138.

Zhang, T.; Guo, S.; Tan, T.; Hu, X.; and Chen, F. 2020. Generating Adjacency-Constrained Subgoals in Hierarchical Reinforcement Learning. In *Advances in Neural Information Processing Systems*.