Machine Learning (2024) 113:2527–2557
https://doi.org/10.1007/s10994-023-06503-w

Goal exploration augmentation via pre-trained skills for sparse-reward long-horizon goal-conditioned reinforcement learning

Lisheng Wu[1] · Ke Chen[1]

Received: 30 January 2023 / Revised: 13 November 2023 / Accepted: 16 December 2023 /
Published online: 5 February 2024

Abstract
Reinforcement learning often struggles to accomplish a sparse-reward long-horizon task in a complex environment. Goal-conditioned reinforcement learning (GCRL) has been employed to tackle this difficult problem via a curriculum of easy-to-reach sub-goals. In GCRL, exploring novel sub-goals is essential for the agent to ultimately find the pathway to the desired goal. How to explore novel sub-goals efficiently is one of the most challenging issues in GCRL. Several goal exploration methods have been proposed to address this issue but still struggle to find the desired goals efficiently. In this paper, we propose a novel learning objective by optimizing the entropy of both achieved and new goals to be explored for more efficient goal exploration in sub-goal selection based GCRL. To optimize this objective, we first explore and exploit the frequently occurring goal-transition patterns mined in the environments similar to the current task to compose skills via skill learning. Then, the pre-trained skills are applied in goal exploration with theoretical justification. Evaluation on a variety of spare-reward long-horizon benchmark tasks suggests that incorporating our method into several state-of-the-art GCRL baselines significantly boosts their exploration efficiency while improving or maintaining their performance.

Keywords Goal-conditioned reinforcement learning (GCRL) · Exploration · Sub-goal selection · Skill learning · Long-horizon and sparse-reward tasks

Editors: Fabio Vitale, Tania Cerquitelli, Marcello Restelli, Charalampos Tsourakakis.

✉ Ke Chen
Ke.Chen@manchester.ac.uk

Lisheng Wu
Lisheng.Wu@manchester.ac.uk

[1] Department of Computer Science, University of Manchester, Manchester M13 9PL, UK

1 Introduction

Reinforcement learning (RL) has successfully solved some complex problems, e.g., board games (Silver et al., 2017), protein prediction Jumper et al. (2021) and robotic locomotion tasks (Levine et al., 2016), where rewards as supervision signals play a crucial role in the learning process. Generally, it is possible to solve most if not all tasks via RL as long as the rewards are designed properly (Silver et al., 2021). In contrast to non-trivial reward design principles, setting valuable rewards only for states that reach the desired goals is easier and can generalize across different tasks. Those tasks therefore can be easily framed as goal-conditioned reinforcement learning (GCRL) problems to target at reaching the desired goals. However, the simple reward design also makes it extremely hard for RL to learn how to reach the goals as it is hard for the agent to explore them to obtain valuable rewards for learning. The problems have become more severe in long-horizon tasks where the goals are only reachable beyond a long-horizon. Thus, under the sparse-reward design, how to explore the goals efficiently in long-horizon tasks remains a key problem for the wider applications of RL.

In sparse-reward long-horizon GCRL tasks, instead of directly targeting at the desired goals, the agent often learns to reach an implicit curriculum of sub-goals that are easier to reach and help the agent to discover the pathway to the desired goals. Following the curriculum, the agent gradually expands its reachable sub-goals to cover the desired goals. In the process, the efficiency of exploring new sub-goals for the agent to learn is essential for discovering the desired goals efficiently. Several strategies have been proposed to explore new subgoals efficiently (Florensa et al., 2018; Pong et al., 2020; Pitis et al., 2020; Mendonca et al., 2021; Liu et al., 2022). However, there still exists a large gap to the level of efficiency required by wider RL applications.

The efficient exploration of human beings often establishes on various patterns in the interactions with the environment. Even a baby would master how to explore the room more efficiently via crawling, a kind of behavior patterns that enables the baby to move to nearby positions. We hypothesize that a key component for efficient goal exploration is to utilize the behavior patterns of the agent transitioned to goals nearby, like the baby crawling. However, existing GCRL strategies do not take such kind of patterns into consideration. In our work, we learn such kind of behavior patterns in the form of skills (Florensa et al., 2017; Eysenbach et al., 2018) that are pre-trained on the environments of the properties shared by downstream tasks. Each skill corresponds to an individual policy for the agent to conduct specific behavior patterns. The agent is trained in the pre-training environments to visit a set of different nearby goals following each skill and those skills are transferred to downstream tasks for more efficient exploration. From the viewpoint of exploration, we are interested in behavior patterns that visit goals as widely as possible as it tends to discover more novel goals. Thus, we propose a maximum entropy objective on the distribution of achieved goals induced by following those skills.

Our main contributions are summarized as follows: (1) We propose a maximum entropy goal exploration method, *goal exploration augmentation via pre-trained skills* (GEAPS), to augment exploration in GCRL. (2) We introduce the entropy of goals in skill learning, which stabilizes skill learning and helps the agent gain more efficiency in goal exploration on challenging downstream tasks. Furthermore, we conduct a theoretical analysis of this entropy-based skill learning method. (3) We provide theoretical analyses for the benefits of utilizing pre-trained skills and the effectiveness achieved through our exploration strategy under specific conditions. (4) We demonstrate that incorporating our GEAPS algorithm into the state-of-the-art GCRL methods boosts their exploration efficiency for several sparse-reward long-horizon benchmark tasks.

2 Related work

Exploration for New Goals. Using uniformly sampled actions, like ϵ-greedy algorithm, and introducing noises to policy actions are common strategies for exploration in RL. However, they are not sufficient to solve sparse-reward and long-horizon tasks. Different goal exploration methods have been proposed to accomplish those challenging tasks. A class of methods focus on a sub-goal selection strategy that helps with better goal exploration. Skew-Fit (Pong et al., 2020) samples sub-goals from a skewed distribution that is approximately uniform over historical achieved goals, and OMEGA (Pitis et al., 2020) selects sub-goals by maximizing the entropy of achieved goals from low-density regions. Goal GAN (Florensa et al., 2018) and the AMIGO (Campero et al., 2020) select sub-goals of intermediate difficulties that prevent the agent from getting trapped in too easy tasks and avoiding too difficult ones. By and large, however, such methods still rely on uniformly sampled actions and action noise to find new goals while pursuing sub-goals, which restricts the goal exploration to neighboring states along the trajectory to the sub-goal. To overcome this limitation, Pitis et al. (2020), Hoang et al. (2021) and Hartikainen et al. (2020) additionally explore goals via random actions after reaching the specific sub-goal, which gives the agent larger freedom to explore beyond the sub-goal. Nevertheless, random actions do not involve any learned knowledge about tasks other than the action space, which restricts them from exploring a wide range of goals. In contrast, we involve behavior patterns transitioned to nearby goals in the form of skills pre-trained in similar tasks. The pre-trained skills enable transition to nearby goals quicker so that a wider range of goals can be explored within the same time steps. As a model-based method, LEXA (Mendonca et al., 2021) trains an exploration policy in a world model of the environment to discover novel goals and perform exploration via the trained exploration policy in the environment. However, a notable lack of experiences around the novel goals makes the simulated dynamics inaccurate around them. The inaccurate simulated dynamics also prevent the exploration policy from exploring a wider range of novel goals. As a model-free method, our method does not rely on the exact dynamics around the novel goals. Instead, we explore new goals with the behavior patterns transitioned to nearby goals to increase the chance for the agent to reach the nearby goals faster than those methods without any knowledge of goal transition.

Skill Learning. To learn the behavioral patterns transitioned to nearby goals, we perform skill learning in pre-training environments with each skill learning to reach a different set of goals. To achieve this, a well-known idea is to maximize the mutual information between skills and the goals that are going to be visited, which can be expressed as follows:

$$I(\mathcal{G};\mathcal{Z}) = H(\mathcal{Z}) - H(\mathcal{Z}|\mathcal{G}) \tag{1a}$$

$$= H(\mathcal{G}) - H(\mathcal{G}|\mathcal{Z}) \tag{1b}$$

where $\mathcal{G}$ is the goal space and $\mathcal{Z}$ denotes the latent space of the skill policy where each skill is represented by the skill policy conditioned on an individual latent vector. As the state itself can be considered as a goal, we would review the related works below in terms

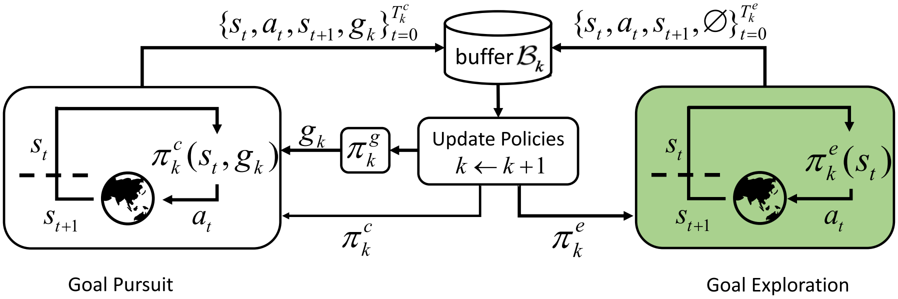

Fig. 1 A generic goal-conditioned reinforcement learning (GCRL) framework for long-horizon and sparse-reward tasks

of goals for simplicity. With Eq. 1a, SNN4HRL (Florensa et al., 2017) and DIYAN (Eysenbach et al., 2018) learn skills by fixing the distribution of latent vectors and minimizing the conditional entropy $H(\mathcal{Z}|\mathcal{G})$. DADS (Sharma et al., 2019) estimates $H(\mathcal{G})$ and $H(\mathcal{G}|\mathcal{Z})$ with the help of a skill-dynamics model and learns the skills via Eq. 1b. However, their learned skills can cover only a small portion of reachable goals due to the fact that mutual information may have many optima and covering more goals does not necessarily contribute to higher mutual information. EDL (Campos et al., 2020) explores the goal space at first, then encode those goals into discrete latent vectors $\mathcal{Z}$ via a trained VQ-VAE (Van Den Oord et al., 2017), and finally learn each skill from the rewards based on the likelihoods of the achieved goals that are predicted by the VQ-VAE decoder. Though the skills learned via EDL can reach goals further away, they are not optimized to reach all reachable goals. As the pre-training environments do not reveal the exact structures of downstream tasks, some behavioral patterns transitioned to nearby goals may not work out as expected. Thus, we expect the learned behavioral patterns to support as many transitions to nearby goals as possible so that they can be more robust to different situations in downstream tasks. To achieve this, we introduce an alternative objective for skill learning based on mutual information maximization. The maximized entropy of goals ensures that skills can reach a wider range of nearby goals and avoid bad local optima in goal covering. as demonstrated in our experiments reported in Sect. 5.3.4.

3 Preliminary

While traditional reinforcement learning is often modeled as a *Markov decision process* (MDP), GCRL augments the MDP with a goal state to form *goal-augmented* MDP (GA-MDP) (Schaul et al., 2015). A GA-MDP $\mathcal{M}^{\mathcal{G}}$ is denoted by a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{G}, r, \gamma, \phi, p_{dg}, T)$ where $\mathcal{S}, \mathcal{A}, \gamma, T$ are state space, action space, discount factor and the horizon, respectively. $\mathcal{T}$: $\mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0, 1]$ is the transition function, $\mathcal{G}$ is the goal space, p_{dg} is the desired goal distribution and $\phi : \mathcal{S} \rightarrow \mathcal{G}$ is a tractable mapping function that maps a state to its corresponding achieved goal. The reward function $r : \mathcal{S} \times \mathcal{G} \times \mathcal{A} \rightarrow \mathbb{R}$ provides the learning signals for the agent, but valuable rewards can only be obtained when the agent reaches the desired goals in the sparse-reward setting. GCRL requires the agent to learn a policy $\pi : \mathcal{S} \times \mathcal{G} \times \mathcal{A} \rightarrow [0, 1]$ to maximize the expected cumulative return:

$$J(\pi) = \mathbb{E}_{\substack{g \sim p_{dg}, a_t \sim \pi(\cdot|s_t, g) \\ s_{t+1} \sim \mathcal{T}(\cdot|s_t, a_t)}} [\sum_{t=1}^{T} \gamma^t r(s_t, a_t, g)].$$

In GCRL[1], the agent makes actions either in pursuit of a goal or trying to explore more goals. As depicted in Fig. 1, we divide the entire interaction process in an iteration into *goal pursuit* and *goal exploration*, depending on whether the decision policies are conditioned on goals. During policy training in the kth iteration, let π^g_k, π^c_k and π^e_k denote the policy deciding a behavioral goal for an agent to pursue, the goal-conditioned policy for the agent to achieve a goal and the exploration policy for the agent to explore new goals, respectively. Before goal pursuit, a goal g_k will be sampled from $\pi^g_k(\mathcal{G})$, $g_k \sim \pi^g_k(\mathcal{G})$. During goal pursuit, the agent takes an action $a_t \sim \pi^c_k(s_t, g_k)$ at each time step t until $T^c_k \leq T$. For clarity, T^c_k refers to the number of steps required to reach the goal g_k at iteration k in goal pursuit. In the goal exploration process, the agent takes actions $a_t \sim \pi^e_k(s_t)$ until $T^e_k \leq T$. To make the best use of interaction steps, we perform goal exploration subsequently after the agent achieves the goal during goal-pursuit (Pitis et al., 2020; Hoang et al., 2021; Hartikainen et al., 2020) instead of conducting goal exploration separately. Thus, the total steps in iteration k is $T = T^c_k + T^e_k$, meaning that the number of steps taken for goal exploration, T^e_k, depend on T^c_k in iteration k. The data collected from both goal pursuit and exploration are stored in the replay buffer $\mathcal{B}_k$ at iteration k. In the $(k+1)$th iteration, $\pi^g_k, \pi^c_k, \pi^e_k$ would be updated to $\pi^g_{k+1}, \pi^c_{k+1}, \pi^e_{k+1}$, respectively, based on the training data in the current replay buffer $\mathcal{B}_k$. Then, the updated policies will be used in the new round of data collection. Furthermore, we denote the achieved goals as the set $\mathcal{G}_{\mathcal{B}} : \{\phi(s)|s \in \mathcal{B}_k\}$, their distribution in the goal space $\mathcal{G}$ as $p_{ag,k}(\mathcal{G})$ and their entropy as $H_{ag,k}(\mathcal{G})$ at iteration k. To simplify the presentation, we shall drop off the explicit iteration index, k, from the subscript of the above notation in the rest of the paper.

4 Method

In this section, we propose a new learning objective for goal exploration, then present skill learning via the goal-transition patterns to optimize our learning objective, which leads to our GEAPS algorithm.

4.1 Learning objective for goal exploration

Unlike the previous works reviewed in Sect. 2, we focus on the goal exploration associated with goal-independent behavior. As it is hard to directly explore desired goals in long-horizon and sparse-reward tasks, a well-known learning objective is to maximize the entropy of historical achieved goal $H(\mathcal{G})$. OMEGA (Pitis et al., 2020) has shown how to optimize the entropy of already achieved goals, $H_{ag}(\mathcal{G})$, in the goal pursuit process. We make a step forward by analyzing how to further optimize $H_{ag}(\mathcal{G})$ via goal exploration immediately after

[1] Some GCRL methods may not have all the components in the generic framework shown in Fig. 1.

goal pursuit in each trial. Let $p_e(\mathcal{G})$ and $H_e(\mathcal{G})$ denote the distribution of goals encountered in goal exploration and its entropy, respectively. In the goal exploration process starting with the initial state s_0[2] and going through T^e transitions, we have

$$p_e(g|s_0) = \frac{1}{T^e} \sum_{i=0}^{T^e-1} \prod_{t=0}^{i} p(s_{t+1}|s_t, a_t)\pi^e(a_t|s_t)\mathbb{1}(\phi(s_{t+1}) = g). \tag{2}$$

After goal exploration (c.f. Fig. 1), the updated distribution of achieved goals $p'_{ag}(\mathcal{G})$ is a weighted mixture distribution of $p_{ag}(\mathcal{G})$ and $p_e(\mathcal{G})$ as follows:

$$p'_{ag}(\mathcal{G}) = c\, p_{ag}(\mathcal{G}) + (1-c)p_e(\mathcal{G}),$$

where $c = \frac{|\mathcal{B}|+T^c}{|\mathcal{B}|+T^c+T^e}$ and $|\mathcal{B}|$ is the size of the current replay buffer.

To develop our learning objective for goal exploration augmentation, we formulate a proposition as follows:

Proposition 1 *Let $H'_{ag}(\mathcal{G})$ represent the updated entropy of achieved goals following the goal exploration. This entropy is bounded from below by the sum of the weighted entropies of the original achieved goals and the goals encountered during goal exploration, namely, $c\, H_{ag}(\mathcal{G})$ and $(1-c)\, H_e(\mathcal{G})$. That is,*

$$H'_{ag}(\mathcal{G}) \geq cH_{ag}(\mathcal{G}) + (1-c)H_e(\mathcal{G}). \tag{3}$$

The proof of Proposition 1 can be found in Appendix 1. According to Eq. 3, an increase in $H_{ag}(\mathcal{G})$ and $H_e(\mathcal{G})$ elevates the lower bound of the resulting entropy $H'_{ag}(\mathcal{G})$. As the OMEGA (Pitis et al., 2020) asserts, $H_{ag}(\mathcal{G})$ can be maximized by selecting low-density goals as sub-goals. However, optimizing $H_e(\mathcal{G})$ is challenging due to the agent's limited understanding of new sub-goal dynamics, which may necessitate arbitrary exploration.

Despite unknown dynamics, we observe that overlapping elements may exist between the agent's transition mechanisms and a pre-training environment. These shared features form goal-transition patterns, beneficial for exploring unfamiliar goals. When all goal-transition patterns are available in a new sub-goal, the generic entropy of explored goals $H_e(\mathcal{G})$ is denoted by $\hat{H}_e(\mathcal{G})$. To optimize $\hat{H}_e(\mathcal{G})$, an exploration policy must aim to visit as many goals as feasible within a given time frame, while avoiding revisits and maintaining stochasticity. Backed by theoretical justification presented in Sect. 4.5.1, we suggest developing an exploration policy based on an array of stochastic pre-trained skills. Each skill targets a maximum set of sub-goals, leading to a maximized $\hat{H}_e(\mathcal{G})$. Although this assumption may not apply during actual exploration, $\hat{H}_e(\mathcal{G})$ still acts as an upper bound of $H_e(\mathcal{G})$ even though missing goal-transition patterns lead to failed transitions. Hence, enhancing $\hat{H}_e(\mathcal{G})$ could significantly improve the agent's exploration efficiency.

4.2 Skill acquisition

For a given environment, optimizing our learning objective in Eq. 3 leads to the maximum entropy of goals to be explored in goal exploration (c.f. Fig. 1). However, the exact dynamics

[2] To simplify the notation, we designate the state, s_{T^c}, reached by the goal pursuit after T^c transitions as the initial state, s_0, that triggers the goal exploration process (see Sect. 3 and Fig. 1 for clarity).

around the current state is often unknown, hence it is infeasible to directly maximize the entropy of goals to be explored via $p_e(\mathcal{G})$ in Eq. 2. Fortunately, this issue can be addressed with the auxiliary information named goal-transition patterns. A goal transition always has a starting goal g_s and an end goal g_e but goal transitions of the same g_s and g_e may involve different intermediate states. Here, we define a *goal-transition pattern* as a goal transition process that can transit across different states with actions but preserves the same properties independent of g_s and g_e in the goal space $\mathcal{G}$. It is analogous to image recognition where an object's identity is independent of its location in the image. Exploring with a goal-transition pattern from a state tends to make the changes specified by the pattern via goal-independent actions in $\mathcal{G}$. Goal-transition patterns enable planning in $\mathcal{G}$ to avoid the canceling-out effect of different actions used for goal exploration. Composing a set of frequently occurring inherent goal-transition patterns, named *skills*, in a manner that maximizes the entropy of goals to be explored enables an agent to expand its achieved goal space more efficiently for better goal covering. Such skills can be learned via another policy as described below.

Although we cannot find all the frequently occurring goal-transition patterns without traversing the entire environment, we observe that there are many goal-transition patterns in common that can be mined from similar environments via pre-training. A pre-training environment should share both the same agent space $\mathcal{S}^{agent}$ (Florensa et al., 2017; Konidaris & Barto, 2007) and the same goal space $\mathcal{G}$ with the current task. The agent space $\mathcal{S}^{agent}$ is simply a shared subspace of the state space $\mathcal{S}$ and semantically the same across a collection of relevant tasks. $\mathcal{S}^{agent}$ generally does not convey goal information since the transition dynamics in their goal spaces often differ on the pre-training tasks. In our work, $\mathcal{S}^{agent}$ needs to be independent of the goal space $\mathcal{G}$ of any tasks. Thus, the goal-transition patterns can be transferred to a GCRL task within $\mathcal{S}^{agent}$ via learned policies that execute the inherent goal-transition patterns mined in the pre-training environments. As our ultimate goal is to learn the composition of goal-transition patterns or skills, we can directly learn another policy that maximizes the expected entropy of goals to be explored in the pre-training environments without modeling the behavior for each goal-transition pattern explicitly. Thus, the behavior of frequently occurring goal-transition patterns is automatically encoded by the policy via learning. We formulate such policy learning as a *skill learning* process.

4.3 Skill learning

We denote a skill by a latent vector z, the set of all the pre-trained skills by $\mathcal{Z}$, and the corresponding multi-modal skill policy by $\pi_{\mathcal{Z}}$. For each skill, $\pi_{\mathcal{Z}}$ would select an action $a_t \sim \pi_{\mathcal{Z}}(a_t|s_t^{agent}, z)$. To learn a set of diverse skills, we formulate its learning objective as the mutual information between the skills and the goals conditioned on initial goal states by Eqs. 1a and 1b. However, previous skill learning methods often fail to learn a wide coverage of goals, which is attributed to the fact that there exist many optima in the mutual information function and covering more goals does not always lead to higher mutual information. Without loss of generality, we assume both the goal space $\mathcal{G}$ and the latent space $\mathcal{Z}$ for skills are discrete and it is common to have $|\mathcal{G}| > |\mathcal{Z}|$. Even when the mutual information $I(\mathcal{G};\mathcal{Z})$ has been maximized to be $\log|\mathcal{Z}|$ via Eq. 1a, the entropy of goals $H(\mathcal{G})$ can still vary from $\log|\mathcal{Z}|$ to $\log|\mathcal{G}|$. When $H(\mathcal{G})$ takes low values, the goal coverage appears poor, which motivates us to develop an alternative skill learning strategy.

Unlike the prior skill learning works, e.g., SNN4HRL (Florensa et al., 2017) and DIAYN (Eysenbach et al., 2018), we want a diverse set of skills by maximizing both $I(\mathcal{Z}, \mathcal{G})$ and $H(\mathcal{G})$.

In our work, we do not maximize $H(\mathcal{G})$ directly but $H(\mathcal{G}\clubsuit\mathcal{Z})$ instead given the fact that when $I(\mathcal{Z},\mathcal{G})$ is maximized, Eq. 1b leads to

$$
\begin{aligned}
H(\mathcal{G}) &= I(\mathcal{Z},\mathcal{G}) + H(\mathcal{G}|\mathcal{Z}) \\
&= H(\mathcal{Z}) - H(\mathcal{Z}|\mathcal{G}) + H(\mathcal{G}|\mathcal{Z}) \\
&= E_{z,g}[\log p(\mathbf{z}|g) - \log p(\mathbf{z}) - \log p(g|\mathbf{z})].
\end{aligned} \tag{4}
$$

In the skill learning process, however, we still cannot obtain the exact $p(\mathbf{z}|g)$ and $p(g|\mathbf{z})$ that requires integration over all reachable goals and skills. We approximate $p(\mathbf{z}|g)$ and $p(g|\mathbf{z})$ with $q(\mathbf{z}|g)$ and $q(g|\mathbf{z})$ by using the Monte Carlo method. Motivated by the previous works (Florensa et al., 2017; Eysenbach et al., 2018), we set the reward for mutual information maximization as

$$r^I_z(s_t, a_t) = \log q(\mathbf{z}|\phi(s_t)) - \log p(\mathbf{z}). \tag{5}$$

To maximize the entropy of $H(\mathcal{G}|\mathcal{Z})$, the distribution $p(g|\mathbf{z})$ is expected to be as uniform as possible. Thus, we design the reward as follows:

$$r^H_z(s_t, a_t) = \log\left[\max_{\hat{g}} q(\hat{g}|\mathbf{z}) - q(\phi(s_{t+1})|\mathbf{z})\right]. \tag{6}$$

Here, we encourage visiting those goals less explored. Thus, it will converge when the distribution of goals to be explored are uniform. Combining the rewards specified in Eqs. 5 and 6, we achieve the pseudo reward for our skill training as follows:

$$r_z(s_t, a_t) = r^I_{\mathbf{z}}(s_t, a_t) + \beta r^H_{\mathbf{z}}(s_t, a_t). \tag{7}$$

where β is a coefficient to trade-off between r^I_z and r^H_z.

In GCRL, transitioning between goals g and g' is often represented as $g \rightarrow g + \Delta(g, g')$, where $\Delta G = \Delta(g, g')$ signifies the desired goal transition. To optimize $\hat{H}_e(\mathcal{G})$, we actually learn the skills by maximizing $H(\Delta\mathcal{G})$ in a pre-training environment.

Algorithm 1 Goal Exploration Augmentation via Pre-trained Skills (GEAPS)

Given: Skill space $\mathcal{Z}$, pre-trained skill policy $\pi_{\mathcal{Z}}$, initial state for goal exploration s_0, skill horizon T^s, goal exploraton horizon T^e, replay buffer $\mathcal{B}$.

```
procedure GEAPS
    while $t \leq T^e$ do
        if $t$ mod $T^s = 0$ then
            sample a skill $z \sim p(z|s_t)$
        end if
        sample the action $a_t \sim \pi_{\mathcal{Z}}(s_t^{agent}, \boldsymbol{z})$
        $s_{t+1} \sim \mathcal{T}(s_{t+1}|s_t, a_t)$
        $t \leftarrow t + 1$
        save $(s_t, a_t, s_{t+1}, \varnothing)$ in replay buffer $\mathcal{B}$.
    end while
end procedure
```

4.4 Goal exploration augmentation strategy

The trained skill policy is used in π_e for goal exploration (c.f. Fig. 1). During goal exploration, goal-transition patterns are not available in all states, hence sometimes the desired transition of skills are unreachable. Thus, we switch a skill in every T^s ($T^s < T^e$) steps. We summarize our GEAPS algorithm in Algorithm 1 that enables the trained skills $\pi_{\mathcal{Z}}$ to be used for formulating an exploration policy π_e during goal exploration.

As depicted in Fig. 1, our GEAPS algorithm can be easily incorporated into the generic GCRL framework to improve exploration efficiency. In the goal pursuit process, an existing GCRL algorithm, e.g., Goal GAN (Florensa et al., 2018), Skew-Fit (Pong et al., 2020), or OMEGA (Pitis et al., 2020), used in our experiments, is employed to learn a policy, π_k^g, for an agent to decide a behavioral goal to pursue and a goal-conditioned policy, π_k^c, for an agent to achieve a goal. After the kth round of goal pursuit is completed, our GEAPS algorithm is invoked in the goal exploration stage to construct an exploration policy, π_k^e, for the agent to explore new goals. Alternating the goal pursuit and exploration processes makes the two algorithms reach a synergy to solve a sparse-reward long-horizon reinforcement learning task.

4.5 Theoretical analysis

In this subsection, we provide theoretical analyses concerning our entropy-maximization-based methods for skill learning and goal exploration. The proofs of those propositions can be found in Appendix 1.

4.5.1 On the role of skill composition in learning optimal exploration policy

Under the exploration policy π^e, we characterize Ω as the set of all possible trajectories encompassed within the exploration horizon T^e. Each exploration trajectory, represented by τ, conforms to the distribution portrayed by $\tau \sim p(\Omega)$. Consequently, the entropy $\hat{H}_e(\mathcal{G})$ can be articulated as the sum of the mutual information $I(\Omega;\mathcal{G})$ and the conditional entropy $H(\mathcal{G}|\Omega)$. This can be represented mathematically as follows:

$$\hat{H}_e(\mathcal{G}) = I(\Omega;\mathcal{G}) + H(\mathcal{G}|\Omega).$$

To maximize the mutual information $I(\Omega;\mathcal{G})$, we strive for each trajectory to cover a distinct subset of goals. With respect to optimizing $H(\mathcal{G}|\Omega)$, we aim for each trajectory to visit as many goals as feasible, maintaining uniform probability for visiting each goal within its respective subset. In exploration scenarios deploying uniform primitive actions, each trajectory bears an equivalent likelihood $\frac{1}{|\Omega|}$ of being generated. It is crucial to note that some trajectories may be limited to a single goal, whereas a specific goal could be visited by numerous trajectories.

For the optimal exploration policy, we denote the set of trajectories following the policy as $\Omega^* \subseteq \Omega$, with their corresponding distribution represented as p_{Ω^*}. Consequently, this optimal exploration policy culminates in the optimal entropy of prospective goals, denoted by $\hat{H}_e^*(\mathcal{G})$, as given as follows:

$$\hat{H}_e^*(\mathcal{G}) = I(\Omega^*;\mathcal{G}) + H(\mathcal{G}|\Omega^*).$$

However, directly optimizing the exploration policy within the trajectory space Ω to obtain Ω^* and p_{Ω^*} poses significant computational challenges. This complexity primarily arises from the exponential growth in the size of $|\Omega| = |\mathcal{A}|^{T^e}$ as a function of T^e. This difficulty is further compounded by the potential continuity of the action space. To mitigate these challenges, we consider the prospect of simplifying the optimization problem.

Proposition 2 *The optimal exploration policy leading to Ω^* with the distribution p_{Ω^*} can be composed via a set of skills $\mathcal{Z}$ ($|\mathcal{Z}| << |\Omega^*|$).*

Following this proposition, we firmly advocate the prospect of pre-trained skills as a viable mechanism for acquiring optimal settings. This approach offers a captivating alternative to the direct optimization method, providing promising opportunities for efficient exploration.

4.5.2 On the role of pre-trained skills in improving exploration efficiency

In a given environment, the occurrence of a goal-transition ΔG can be recurrent, and this repetitiveness is evident through various distinctive goal-transition patterns. To systematically understand these patterns, we formulate a goal-transition pattern as $\psi = \{s_{start}^{agent}, s_{end}^{agent}, \Delta G, \Delta A\}$. Within this formulation, s_{start}^{agent} and s_{end}^{agent}, which belong to the agent state space $\mathcal{S}^{agent}$, are the initial and terminal states of the agent for the goal-transition ΔG, respectively. ΔA denotes a sequence of actions that result in the accomplishment of ΔG. The term $|\Delta A|$ signifies the length of the action sequence ΔA, and the cardinality of ψ is denoted by $|\psi| = |\Delta A|$. The entirety of existing goal-transition patterns is symbolized by Ψ.

We now delve into an analysis of how goal-transition patterns can enhance exploration efficiency. Under the assumption that all goal-transition patterns in the pre-training environment are accessible, we aim to optimize $\hat{H}_e(\mathcal{G})$. Our initial focus lies on discerning the associations between these goal-transition patterns and individual episodes.

Proposition 3 *Given the horizon T, every trajectory τ can be decomposed into a sequence of goal-transition patterns.*

With the established connection between each trajectory and its corresponding goal-transition patterns in Proposition 3, we proceed to analyze the potential of enhancing exploration efficiency based on the goal-transition decomposition of each trajectory.

Proposition 4 *Given an exploration horizon of T^e, the substitution of goal-transition patterns within each trajectory $\tau \in \Omega$ with alternative patterns of smaller cardinality can yield equivalent exploration outcomes using an average number of steps that is less than or equal to the specified T^e.*

Proposition 4 presents a potential avenue for enhancing exploration efficiency over the exploration policy that relies on uniform primitive actions. In a practical scenario, goal-transition patterns of different cardinality coexist for the same goal transition, so the employment of goal-transition patterns typically results in a requirement for fewer time steps than T^e.

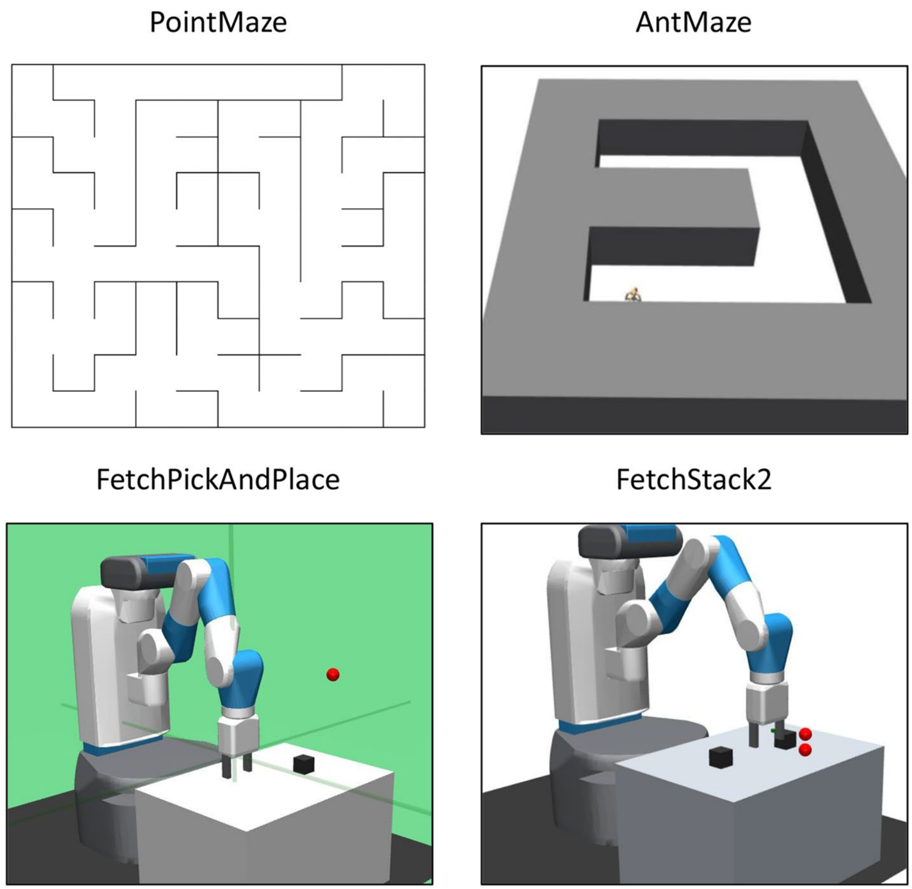

Fig. 2 Four sparse-reward and long-horizon benchmark tasks

Furthermore, the skill learning methodologies described in Sects. 4.2 and 4.3 are designed to exploit the potential of goal-transition patterns, thereby further improving the exploration efficiency.

5 Experiment

In this section, we evaluate the advantage of our GEAPS in terms of *success rate* and *sampling efficiency* on a set of sparse-reward and long-horizon GCRL benchmark tasks and demonstrate the effectiveness of our pre-trained skills in our GEAPS algorithm via a comparative study.

5.1 Environments and baselines

5.1.1 Environments

As shown in Fig. 2, we select four common long-horizon and sparse reward environments in our experiments. (i) `PointMaze` (Pitis et al., 2020; Trott et al., 2019): a 2-D maze task that a point navigates through a 10×10 maze from the bottom left corner to the top right one. Its observation is a two-dimensional vector indicating its position in the maze. The agent may be easily trapped in somewhere with dead ends hence hardly exploring new goals. (ii) `AntMaze` (Pitis et al., 2020; Trott et al., 2019): a robotic locomotion task that controls a 3-D four-legged robot through a long U-shaped hallway to reach the desired goal position. Its thirty-dimensional observation includes the robot's status and its location. The agent can only move in a slow and jittery manner, hence it is hard to explore new goals and learn goal-reaching behavior. (iii) `FetchPickAndPlace` (Plappert et al., 2018): a robot arm task where a robot arm grasps a box and moves it to a target position. The agent observes the positions of both gripper and target box as a 30-D vector, and its goal is a 3-D vector about the target position for the box. Another robot arm task, `FetchStack2` (Nair et al., 2018), aims to stack two boxes at a target location, requiring the agent to move them to target positions in order. Its observation and goal are 40-D and 6-D, respectively. The agent receives no reward until placing both boxes in the correct positions, and involving two boxes makes it more difficult to explore desired goals. Reaching the desired goals once on `PointMaze` and `AntMaze` is considered as a success, while the agent is only considered to succeed on `FetchPickAndPlace` and `FetchStack2` if it still satisfies the conditions of desired goals at the end of episodes.

5.1.2 Baselines

(i) Goal GAN (Florensa et al., 2018): a typical heuristic-driven method where the intermediate difficulty is used as a heuristic to select sub-goals via a generative model. (ii) Skew-Fit (Pong et al., 2020): an effective exploration-based method where sub-goals are generated via sampling from a learnt skewed distribution that is approximately uniform on achieved goals. (iii) OMEGA (Pitis et al., 2020): yet another effective exploration-based method where sub-goals are generated via sampling from the low-density region of an achieved goal distribution. While our GEAPS algorithm is applied to the above baselines to augment their goal exploration, we also compare three augmented GCRL models to the state-of-the-art (SOTA) model-based exploration in LEXA (Mendonca et al., 2021) of which explorer can augment suitable baselines, e.g., GCSL (Ghosh et al., 2020) and DDPG (Lillicrap et al., 2015).

5.2 Experimental settings and implementation

5.2.1 Experimental settings

Our experiments study the following questions: (Q1) How much is the sampling efficiency gained when our GEAPS is incorporated into a baseline on the condition that

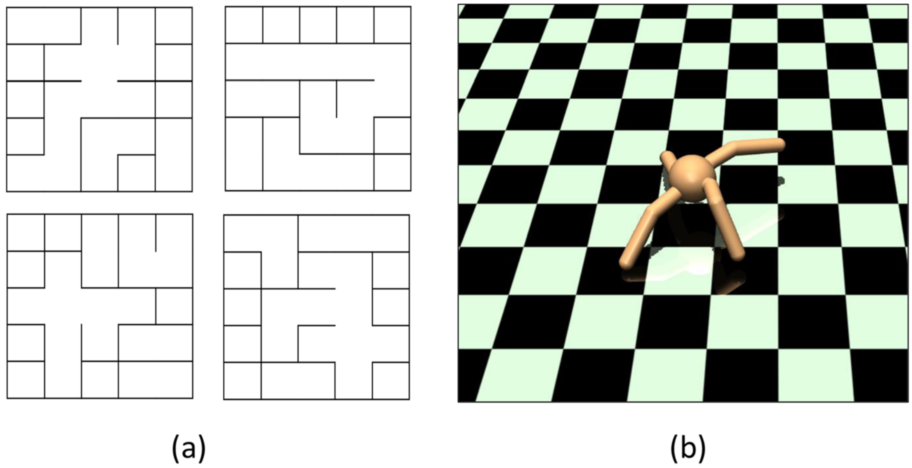

Fig. 3 **a** Four typical pre-training environments for `PointMaze`. **b** The pre-training environment for `AntMaze`

its performance is maintained or even improved? (Q2) What are the behavioral changes resulting from incorporating our GEAPS into a baseline? (Q3) Can our augmented models reach the performance yielded by compared to the model-based SOTA exploration in LEXA (Mendonca et al., 2021)? (Q4) What are the pre-trained skills resulting from our skill learning objective in contrast to those generated by the established skill learning methods such as SNN4HRL (Florensa et al., 2017) and EDL (Campos et al., 2020)?

For each baseline, we apply our GEAPS in goal exploration by keeping its original settings unchanged. Thus, we achieve three augmented models: Goal GAN+GEAPS, Skew-Fit+GEAPS and OMEGA+GEAPS, corresponding to three baselines. Five trials with different random seeds in each environment are conducted for reliability. Evaluation is made with a fixed budget; i.e., the training will be terminated after a (pre-set) number of steps if an agent still fails to reach the ultimate goals. The performances are evaluated in terms of the *success rate*, the most important performance evaluation criterion in reinforcement learning, and the *entropy of achieved goals*, a widely used evaluation criterion on sampling efficiency in GCRL.

5.2.2 Pre-training settings

(i) `PointMaze`: We generate 20 small 5×5 small mazes as pre-training environments, which include various topographies. In each episode, the agent is initialized in a random position of the central grid. In Fig. 3a, we exemplify four typical pre-training environments. The observation for the skill policy is the relative position with regard to the grid where the agent is located. We pre-train the skills of horizon two over those mazes via maximizing the average cumulative return averaging over the learning objective. (ii) `AntMaze`: We pre-train the skills on the `Ant` environment as shown in Fig. 3b, which keeps the same 3-D four-legged robots in an open environment and sets the skill horizon as 100.

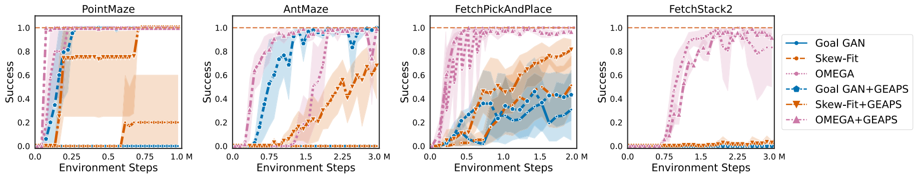

Fig. 4 Test success on the desired goal distribution throughout training on four environments for the baselines and the augmented models

(iii) FetchPickAndPlace and FetchStack2: The goals of two robot arm tasks are defined as the target positions of the relevant objects. Achieving a sub-goal during goal pursuit, typically inferred when the arm continues to hold the object, provides the basis for subsequent skill development. Consequently, each skill is deliberately handcrafted to direct the object along a random trajectory within a predetermined range, ensuring that collectively, the skills span all possible directions. This strategy guarantees an equal likelihood of encountering all potential goals within the boundary established by the exploration horizon T^e.

5.2.3 Implementation

Our GEAPS is implemented with the mrl:modular RL codebase (Pitis et al., 2020) and all the baselines adopt DDPG (Lillicrap et al., 2015) to train goal-conditioned behavior.[3] For three baselines (Florensa et al., 2018; Pong et al., 2020; Pitis et al., 2020), we use the source code provided by the authors and strictly adhere to their instructions in our experiments. LEXA is composed of a model-based exploration policy and a model-based goal-conditioned policy. As our focus is goal exploration, we use its exploration policy only and adopt model-free goal-conditioned policies optimized via GCSL (Ghosh et al., 2020) and DDPG (Lillicrap et al., 2015) for a fair comparison. To obtain the pre-trained skills for PointMaze and AntMaze, we use a multi-layer perceptron trained with the TRPO (Schulman et al., 2015) for stability in skill learning, while the skills for two robot arm tasks are handcrafted as moving along a direction sampled uniformly in various ranges. To pre-train the skills, we fix $\beta = 0.1$ in Eq. 7 for all our experiments. For goal exploration, we set the skill horizon T^s as 2, 25, 8 and 5 for PointMaze, AntMaze, FetchPickAndPlace and FetchStack2, respectively.

Appendixes 2 and 3 describe more technical and implementation details regarding the baselines and the skill learning used in our comparative study.

5.3 Experimental results

We report the main experimental results to provide the answers to four questions posed in Sect. 5.2.1.

[3] The code is available at: https://github.com/GEAPS/GEAPS.

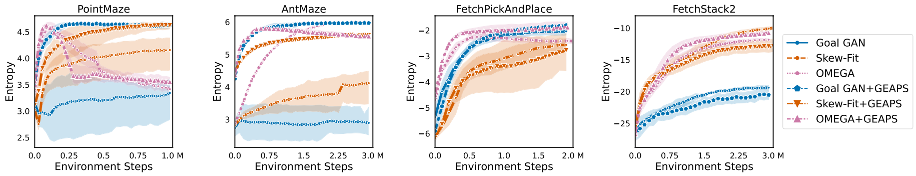

Fig. 5 Empirical entropy of the achieved goal distribution throughout training on four environments for the baselines and the augmented models

5.3.1 Results on goal exploration augmentation

To answer the first question, we report the results yielded by the baselines and their corresponding augmented models to gauge the gain made by our GEAPS. In our experiments, we terminate the training at one million steps for `PointMaze`, two million steps for `FetchPickAndPlace`, three million steps for `AntMaze` and `FetchStack2`. An episode consists of 50 steps for `PointMaze`, `FetchPickAndPlace` and `FetchStack2` and 500 steps for `AntMaze`, respectively. We report statistics (mean and standard deviation) over five seeds in each environment.

As shown in Fig. 4, our GEAPS has improved three baselines in different scales across the four environments. On `PointMaze`, Goal GAN is unable to solve the environment and Skew-Fit only manages to get 20% success at maximum. OMEGA achieves 100% success in about 0.2 million steps. In contrast, our GEAPS enables both Goal GAN and Skew-Fit to solve `PointMaze` to achieve 100% success in about 0.3 and 0.7 million steps. OMEGA+GEAPS is approximately twice faster as OMEGA to achieve 100% success. On `AntMaze`, we observe similar results; both Goal GAN and Skew-Fit fail in three million steps, while our GEAPS enables Skew-Fit to solve the environment with 60% success rates and even boost Goal GAN to have comparable results with OMEGA+GEAPS. OMEGA+GEAPS is about three times faster than OMEGA to reach over 90% success. On `FetchPickAndPlace`, our GEAPS boosts the success rates of Goal GAN and Skew-Fit by around 10% percent and 20% percent, respectively, for the same time steps. Although OMEGA+GEAPS reaches 100% success almost at the same time as the baseline, it is 40% faster than the baseline to reach 80% success. On `FetchStack2`, Goal GAN, Skew-Fit along with their augmented versions hardly solve the problems with at most 7% success observed for Skew-Fit+GEAPS and OMEGA+GEAPS yields the results comparable to OMEGA, which could be explained with the entropy of the achieved goal distribution.

As the entropy of the achieved goal distribution reflects the coverage of achieved goals in an environment, we show the empirical entropy of the achieved goals for the baselines and the augmented models in Fig. 5. On `PointMaze` and `AntMaze`, the entropy increases faster with the help of our GEAPS, and the improvements are especially dramatic on Goal GAN and Skew-Fit. On `FetchPickAndPlace`, GEAPS boosts the entropy of all three baselines on small scales at the beginning. On `FetchStack2`, GEAPS improves the entropy of OMEGA while deteriorating the entropy of Goal GAN and Skew-Fit marginally. The reason on the unimproved entropy on `FetchStack2` can be attributed to GEAPS that controls the goal-transition patterns for one object while making the other with little change, which leads to a slightly lower entropy of achieved goals.

We notice that in OMEGA (Pitis et al., 2020), several classical or SOTA baselines were tested on the same environments used in our work. As OMEGA beats those baselines with a huge margin, e.g., OMEGA is around 100 and 10 times faster than the best performer,

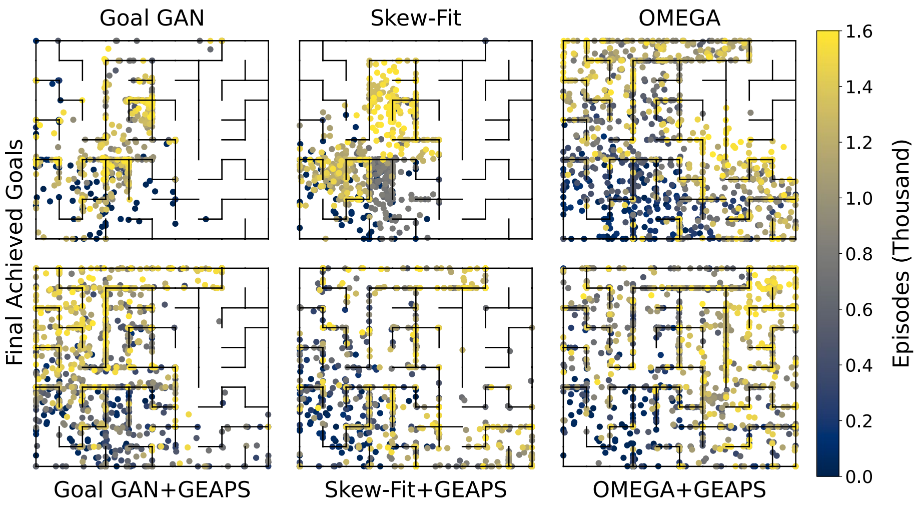

Fig. 6 Visualization of the final achieved goals in `PointMaze`: the baselines (top) versus the augmented models (bottom), where the training evolution process is indicated with the heatmap

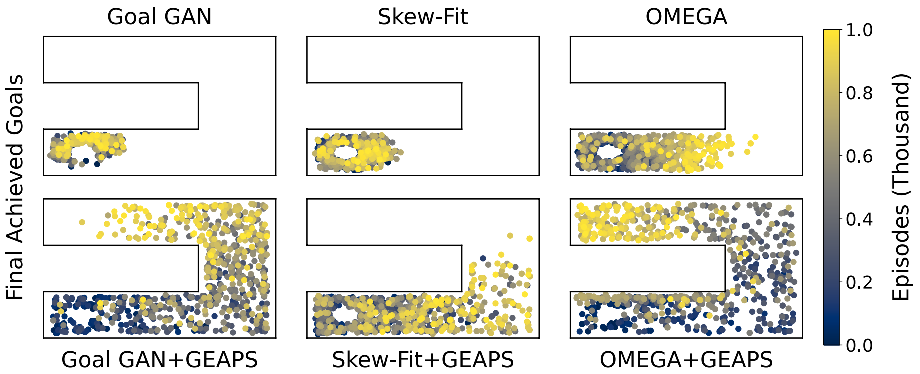

Fig. 7 Visualization of the final achieved goals in `AntMaze`: the baselines (top) versus the augmented models (bottom), where the training evolution process is indicated with the heatmap

PPO+SR (Trott et al., 2019), in solving `PointMaze` and `AntMaze`, respectively. Thus, the performance of OMEGA+GEAPS allows us to claim a bigger gain over those baselines used for comparison in OMEGA (Pitis et al., 2020).

5.3.2 Visualization of exploration behavior

To answer the second question, we visualize the final achieved goals and trajectories of goal selection at the end of the episodes. The visualization vividly exhibits the behavioral changes resulting from our goal exploration augmentation.

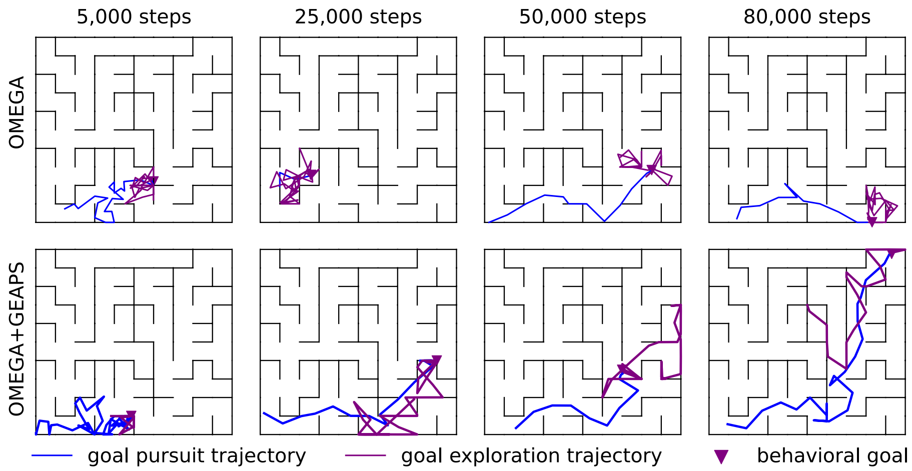

Fig. 8 Visualization of the goal pursuit and goal exploration trajectories made by OMEGA (top) and OMEGA+GEAPS (bottom) at different training steps for `PointMaze`

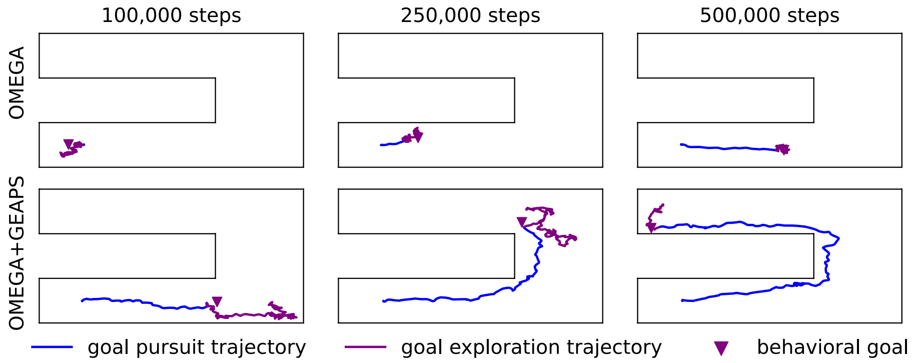

Fig. 9 Visualization of the goal pursuit and goal exploration trajectories made by OMEGA (top) and OMEGA+GEAPS (bottom) at different training steps for `AntMaze`

As shown in the top row of Fig. 6, all three baselines cannot reach the entire area of `PointMaze` at the end of 1600 episodes. Most goals reached by Goal GAN are located in the left half of the maze near the starting location. Most goals reached by Skew-Fit are located in a smaller area in the left half of the maze and goals mainly get stuck in two small areas close to or having a moderate distance to the starting location. OMEGA performs much better than other baselines as it covers the entire maze except those goals from the desired goal distribution in the top right corner. In contrast, it is evident from the bottom row of Fig. 6 that our GEAPS makes all baselines cover a larger area and alleviates the so-called “rich get richer" problem by sampling goals uniformly towards covering the entire maze. In particular, our GEAPS helps the baselines reach goals that spread outward from easy to hard goals as training advances and enables OMEGA to quickly transition to goals in the desired goal area.

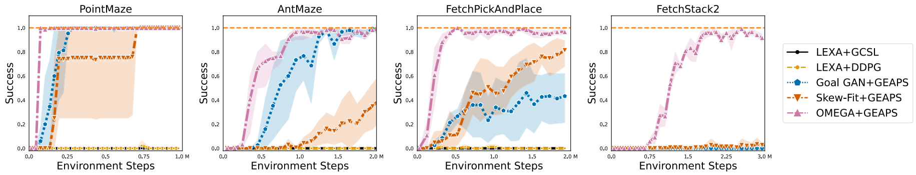

Fig. 10 Test success on the desired goal distribution throughout training on four environments for our augmented models and LEXA explorer-based models

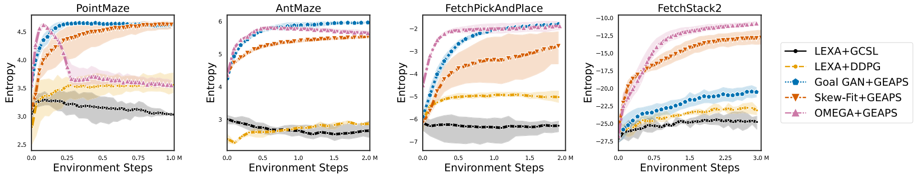

Fig. 11 Empirical entropy of the achieved goal distribution throughout training for our augmented models and LEXA explorer-based models

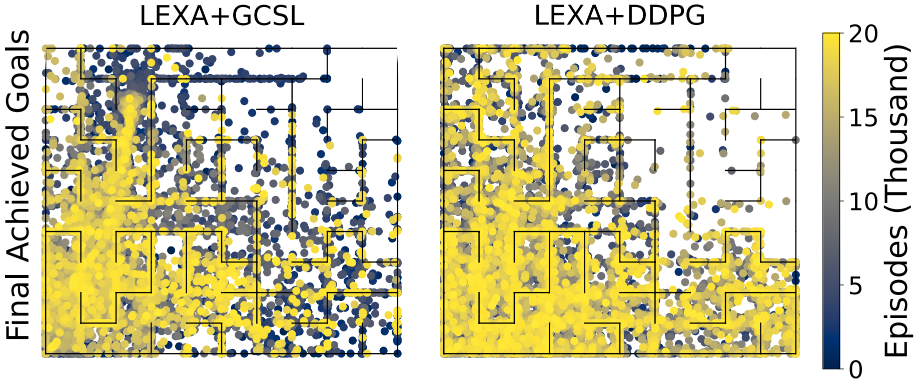

Fig. 12 Visualization of the final achieved goals on `PointMaze`: LEXA+GCSL and LEXA+DDPG, where the training evolution process is indicated by the heatmap

It is observed from Fig. 7 that by incorporating our GEAPS into three baselines, their behavioral changes on `AntMaze` are similar to those on `PointMaze`. At the end of 1000 episodes, no baselines can reach goals beyond the bottom of the hallway, and goals reached by Goal GAN and Skew-Fit are even trapped in the small areas close to the starting location. In contrast, the augmented models take advantage of our GEAPS, hence are able to reach goals in much larger areas. It is evident from the bottom row of Fig. 7 that OMEGA+GEAPS has already explored the goals within the desired goal area and Goal GAN+GEAPS has reached quite close to this area.

As OMEGA is the best performer among the three baselines and also uses a Go-Explore (Ecoffet et al., 2019) style strategy for exploration, we further visualize the goal pursuit and exploration trajectories made by OMEGA and OMEGA+GEAPS at different training steps. As described in Sect. 3, reaching a behavioral goal in goal pursuit triggers goal exploration. A trajectory can intuitively exhibit goal transitioned at

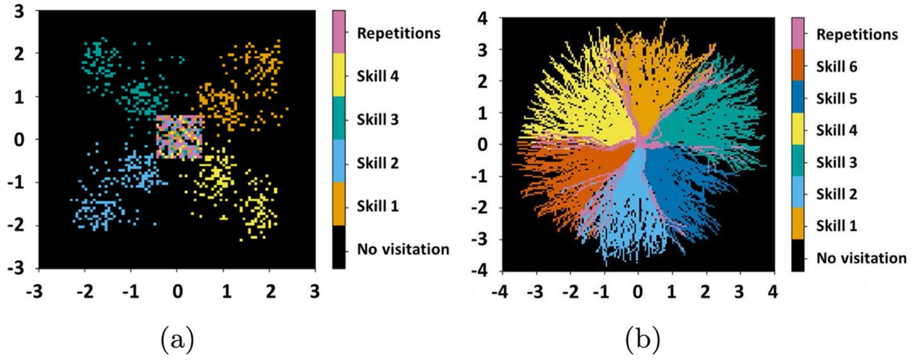

Fig. 13 Trajectories of pre-trained skills acquired by our skill learning method. **a** `PointMaze` in an empty maze. **b** `Ant` in an empty maze

different training steps to allow us to better understand the behavioral change resulting from our GEAPS. As shown in Figs. 8 and 9, OMEGA generally explores goals close to the reached behavioral goals “conservatively" with a heuristic (Pitis et al., 2020) in goal exploration, while OMEGA+GEAPS explores goals in a larger area around the reached behavioral goals “aggressively" by means of the frequently occurring goal-transition patterns encoded in the pre-trained skills, which vividly demonstrates the advantage of our proposed method in improving the exploration effectiveness during learning.

5.3.3 Comparison to LEXA explorer

To answer the third question, we compare the state-of-the-art LEXA explorer-based goal exploration argumentation to our augmented models. As shown in Fig. 10, within the same training budget, we only observe up to 7% success rates on `PointMaze` with LEXA+DDPG and no success achieved by the LEXA explorer-based models on other experiments. As shown in Fig. 11, the entropy of the LEXA explorer-based models are far below that of our augmented models. The reasons may be two-fold: (a) The LEXA explorer performs exploration via the disagreement of an ensemble of one-step world models and the disagreement is based on the novelty of states. Except for the `PointMaze`, the state space is not equivalent to the goal space and exploring more states does not necessarily contribute to exploring more goals. (b) LEXA performs with the goal-pursuit behavior on those goals uniformly sampled from the replay buffer only, which prevents it from enhancing the experience around the explored goals. Form Fig. 12, we observe that the LEXA explorer is able to explore those areas near the desired goal distribution on `PointMaze` while it fails to keep exploring those areas later. In contrast, our augmented models use different strategies to select those novel sub-goals to pursue, which enhances the experience around those novel goals. Our augmented models prioritize pursuing those novel goals to broaden the relevant experience in the replay buffer.

In addition, training a world model in LEXA is time-consuming and requires abundant data, while our GEAPS only needs the skills pre-trained with our alternative learning objective. Those skills acquired by pre-training are applicable to any relevant downstream tasks. In summary, the above results demonstrates that our model-free GEAPS is highly

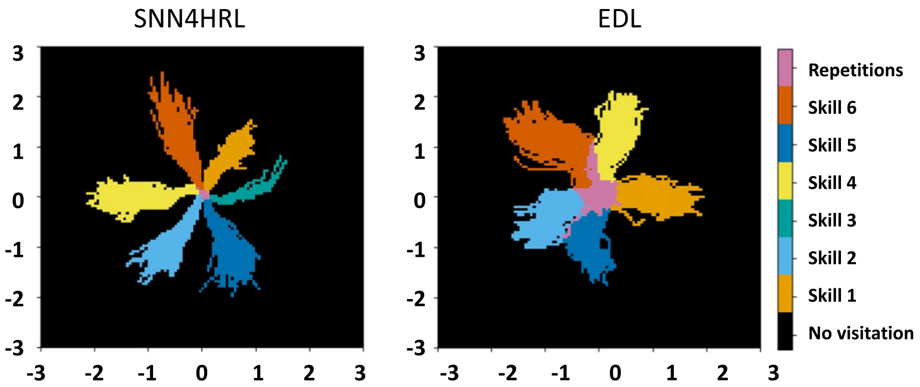

Fig. 14 Trajectories of pre-trained skills learned by SNN4HRL and EDL for `AntMaze`

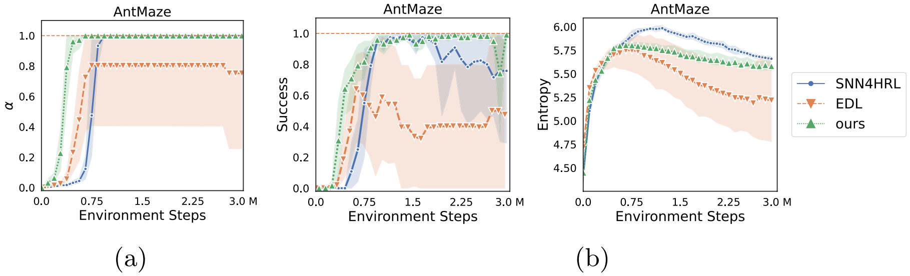

Fig. 15 **a** The dynamic weight α in OMEGA to trade-off the distributions of achieved goals and desired goals in the distributions of sub-goals throughout training on `AntMaze`. **b** Test success on the desired goal distribution and empirical entropy of the achieved goal distribution on `AntMaze` for OMEGA+GEAPS with the pre-trained skills resulting from different skill learning methods

competitive with the SOTA model-based explorer in LEXA especially for the tasks of which state space is not equivalent to their goal space.

5.3.4 Results on skill learning

To answer the fourth question, we first visualize the pre-trained skills resulting from our learning objective presented in Sect. 4.3. Figure 13 illustrates the trajectories of pre-trained skills for `PointMaze` and `AntMaze`. In Fig. 13a, we plot 50 trajectories for each skill pre-trained for `PointMaze` in an empty 5×5 maze. It is evident that the learned skills guide the agent to navigate along different diagonal directions so that the agent can transit to another grid quickly. The skills are intuitive and their effectiveness in boosting the sample efficiency have been demonstrated by the results reported in 5.3.1 and 5.3.2. Figure 13b shows the trajectories of the robot with the skills on the pre-training environment for `AntMaze` with equal probability for 100 episodes. As seen in Fig. 13b, such skills have good coverage of goals in almost all directions. Besides, each skill evenly covers almost the same portion and no skill predominates the coverage of goals.

For comparison, we further illustrate the trajectories of pre-trained skills by SNN4HRL (Florensa et al., 2017) and EDL (Campos et al., 2020) on the same pre-training environment for `AntMaze` in Fig. 14. In contrast to the skills acquired by our method in Fig. 13b, it is evident from Fig. 14 that both SNN4HRL and EDL cover much smaller areas and leave numerous directions uncovered. In our experiment, we observe that the skills acquired by SNN4HRL appear unstable, highly depending on the random seeds.

To investigate the impact of pre-trained skills in our GEAPS, we employ the skills acquired by the different skill learning methods in OMEGA+GEAPS for performance evaluation on `AntMaze`. In OMEGA (Pitis et al., 2020), the factor α calculated via Eq. 13 in Appendix 2 is inversely proportional to the KL divergence between the distribution of achieved goals p_{ag} and desired goals p_{dg} and its value is capped at one. It serves as a dynamic weight to balance both distributions in a mixture distribution for sub-goal sampling and recalculated at the end of each episode. When the α reaches one, the agent only samples sub-goals from the desired goal distribution, which marks the end of exploration about sub-goals other than desired goals. It is evident from Fig. 15a that the pre-trained skills by our method allow for reaching $\alpha = 1$ around 0.7 million steps, while the SNN4HRL skills have to take around one million steps and the EDL skills never lead to $\alpha = 1$. It is further observed from Fig. 15b that our skill learning objective results in earlier success on `AntMaze`; i.e., the agent with our skills starts to explore the desired goal distribution within 0.2 million steps, while the agents with the SNN4HRL and the EDL skills have to take around 0.5 million episodes and 0.3 million steps, respectively. In the later training stage, the agent with our skills maintains high success rates regardless of different random seeds. In contrast, the performance of the agents with the SNN4HRL and EDL skills is degraded substantially. Moreover, we observe that the agent with the EDL skills always fails to solve the `AntMaze` task. In summary, the above results suggest that our skill learning objective yields the quality skills required by our GEAPS.

6 Discussion

In this section, we discuss the limitations/issues arising from our work and make a connection between our method and other related works.

While the advantages of our approach have been demonstrated, several limitations and open problems still remain. First, our approach relies on the pre-trained skills obtained by skill learning in the environments similar to a target task. Our approach will not work if such environments are unavailable. It is also worth stating that the skill learning incurs an additional computational overhead but is rewarded with great exploration efficiency in GCRL to accomplish a sparse-reward long-horizon task. Next, our theoretical analysis establishes the theoretical justification for the benefits of utilizing pre-trained skills and the effectiveness achieved through our exploration strategy under specific conditions. However, further theoretical analyses concerning broader conditions are still pending. Then, the environments used for evaluation have pre-defined yet well-behaved goal spaces and goals have to be in a vectorial form. It is unclear on whether our approach works in the same manner for various scenarios, e.g., an agent has to specify and model/learn its own goal space (Pong et al., 2020), and goals are in other forms (Liu et al., 2022), e.g., image and language goals. After that, all the baselines used in our experiments are sub-goal selection based GCRL algorithms (Liu et al., 2022). Without a considerable extra effort, our GEAPS method cannot be applied to other types of GCRL algorithms such as the optimization-based and the relabelling GCRL algorithms (Liu et al., 2022) for goal exploration augmentation. Finally, our approach is memoryless and thus treats both achieved and new goals to be explored equally during data rollout. Equipped with a memory mechanism, our approach would prevent any visited states from being revisited to further improve the exploration efficiency. With memory and proper pre-trained skills, an agent may accomplish new tasks via searching without any further learning.

It is well known that skills and options have been used in hierarchical reinforcement learning (HRL) for for exploration and task simplification (Sutton, 1998). However, in the context of GCRL, the direct applicability of pre-trained skills for goal attainment and maintenance is quite limited. This is due to the potential for overshooting goals or stochastic reaching, as well as the narrow focus of skills on specific goals (Gehring et al., 2021). In contrast, our GEAPS method combines the benefits of pre-trained skills with the precision of primitive actions, aiming to enhance goal exploration and achieve goals effectively. Below, we summarize several key distinctions between our GEAPS method and existing works that utilize skills/options for exploration in HRL. First, our GEAPS expands the utilization of entropy maximization as a new learning objective in GCRL. By optimizing both achieved and prospective goals, our GEAPS enhances the efficiency of goal exploration. We specifically emphasize goal exploration and incorporate goal-transition patterns into the learning process, enabling more effective exploration even in the absence of precise dynamic knowledge. To the best of our knowledge, these distinctive features cannot be found in existing works on HRL in the context of GCRL. Next, in HRL, a higher-level agent selects from these options, treating them as indivisible actions or atomic actions. Despite exploring goals while executing a skill, HRL often necessitates revisiting goals using more granular options. In contrast, the skills trained in our GEAPS maximize their exploration capabilities based on goal-transition patterns specific to GCRL, allowing for interactions with a broader array of goals during execution. Our method enhances the efficiency of goal exploration and distinguishes our work from conventional HRL practices that prioritize re-engaging with the same set of goals. Even if the skills are pre-trained as sub-policies for specific sub-tasks in HRL (Gehring et al., 2021), each skill tends to primarily focus on a single goal associated with one of the sub-tasks. During execution, this narrow focus can severely limit the skill's ability to interact with a much wider range of goals that arise in GCRL. Then, distinct from the HRL approach, which typically presumes task decomposition through options, our method does not mandate the completion of tasks strictly through pre-trained skills. Rather, within the context of our GEAPS, these skills are intentionally trained to enhance their efficacy in goal exploration, drawing upon goal-transition patterns particular to GCRL. Pre-trained skills, developed with a focus on these specific goal-transition patterns, empower our GEAPS to foster efficient exploration. Our method aligns closely with the innate exploratory behaviors observed in humans and animals, thus encouraging more intuitive interactions with the environment. Finally, we acknowledge theoretical analyses on the exploration benefits of skills and options in HRL, such as the UCRL-SMDP framework (Fruit & Lazaric, 2017) that provides rigorous regret bounds for MDPs with options. However, the direct transfer of UCRL-SMDP to GCRL poses challenges due to disparities in reward mechanisms and the lack of historical data for novel goals. In contrast, our GEAPS addresses these challenges by efficiently navigating exploration in the absence of precise dynamic knowledge. While UCRL-SMDP may not directly aid in exploring unknown areas, a big challenge encountered in our work, it holds promise for enhancing policy optimization to efficiently reach already explored goals in the goal pursuit stage within the generic GCRL framework.

7 Conclusion

In this paper, we have proposed a novel learning objective that optimizes the entropy of both achieved and new goals in sub-goal selection based goal-conditioned reinforcement learning (GCRL). By optimizing this objective, we enhance the efficiency of goal exploration in complex environments, ultimately improving the performance of GCRL algorithms.

Our method incorporates skill learning, where frequently occurring goal-transition patterns are mined and composed into skills. These pre-trained skills are then utilized in goal exploration, allowing the agent to efficiently discover novel sub-goals. Through extensive evaluation on various sparse-reward long-horizon benchmark tasks and a theoretical analysis, we have demonstrated that integrating our method into state-of-the-art GCRL baselines significantly enhances their exploration efficiency while maintaining or improving their performance. The results of our research highlight the importance of effective goal exploration in addressing the challenges of sparse-reward long-horizon tasks. By augmenting the sub-goal section of GCRL models with our model-free goal exploration method, we achieve better coverage of the state space and improve sampling efficiency.

In our future work, there are several avenues for further investigation. First, we plan to conduct further theoretical analyses concerning broader conditions to gain deeper insights into the properties and guarantees of our proposed method. This will provide a solid foundation for understanding its advantages, limitations and potential extensions. Additionally, we aim to explore the application of our method in domains with image data, where the state space is more complex and requires specialized techniques.

In conclusion, our work contributes to the advancement of goal-conditioned reinforcement learning by offering an efficient goal exploration augmentation method. We believe that our research opens up new possibilities for addressing challenging sparse-reward long-horizon tasks in complex environments.

Appendix 1: Proof of Propositions

In this appendix, we provide proofs for the propositions formulated in Sect. 4 of the main text.

Proposition 1 *Let $H'_{ag}(\mathcal{G})$ represent the updated entropy of achieved goals following the goal exploration. This entropy is bounded from below by the sum of the weighted entropies of the original achieved goals and the goals encountered during goal exploration, namely, $c\,H_{ag}(\mathcal{G})$ and $(1-c)\,H_e(\mathcal{G})$. That is,*

$$H'_{ag}(\mathcal{G}) \geq cH_{ag}(\mathcal{G}) + (1-c)H_e(\mathcal{G}).$$

Proof We commence from the entropy definition of $H'_{ag}(\mathcal{G})$:

$$H'_{ag}(\mathcal{G}) = -(c\,p_{ag}(\mathcal{G}) + (1-c)\,p_e(\mathcal{G}))\log\,(c\,p_{ag}(\mathcal{G}) + (1-c)\,p_e(\mathcal{G})).$$

Applying Jensen's inequality due to the concave property of entropy, we find:

$$H'_{ag}(\mathcal{G}) \geq -c\,p_{ag}(\mathcal{G})\log p_{ag}(\mathcal{G}) - (1-c)\,p_e(\mathcal{G})\log p_e(\mathcal{G}).$$

Recognizing $-c\, p_{ag}(\mathcal{G}) \log p_{ag}(\mathcal{G})$ as $cH_{ag}(\mathcal{G})$ and $-(1-c)\, p_e(\mathcal{G}) \log p_e(\mathcal{G})$ as $(1-c)H_e(\mathcal{G})$, we thus establish:

$$H'_{ag}(\mathcal{G}) \geq cH_{ag}(\mathcal{G}) + (1-c)H_e(\mathcal{G}).$$

This completes the proof, demonstrating that the updated entropy $H'_{ag}(\mathcal{G})$ is bounded by the weighted sum of the original entropies.

Proposition 2 *The optimal exploration policy leading to Ω^* with the distribution p_{Ω^*} can be composed via a set of skills $\mathcal{Z}$ ($|\mathcal{Z}| << |\Omega^*|$).*

Proof We can cluster $|\Omega^*|$ into $|\mathcal{Z}|$ clusters and each cluster is represented by a latent vector $z \sim \mathcal{Z}$. Then, we have the corresponding distributions related to z.

$$p(z) = \sum_{\tau \in \Omega^*} p_{\Omega^*}(\tau)\mathbf{1}(\tau \in z), \tag{8}$$

$$p(\tau|z) = \frac{p_{\Omega^*}(\tau)p(z|\tau)}{p(z)} = \frac{p_{\Omega^*}(\tau)\mathbf{1}(\tau \in z)}{p(z)}. \tag{9}$$

In the above expressions, $\mathbf{1}(\tau \in z)$ denotes the indicator function, which equals 1 if τ belongs to the cluster represented by z and 0 otherwise. In this setting, we can transform $\hat{H}^*_e(\mathcal{G})$ with Eqs. 8 and 9 into

$$\hat{H}^*_e(\mathcal{G}) = I(\mathcal{Z};\mathcal{G}) + H(\mathcal{G}\clubsuit\mathcal{Z}).$$

Although the mutual information term, $I(\mathcal{Z};\mathcal{G})$, may decrease, the conditional entropy term, $H(\mathcal{G}\clubsuit\mathcal{Z})$, increases, maintaining the sum unchanged. For generating the optimal trajectories within each cluster, we can train a skill to produce those trajectories. The total number of such skills is $|\mathcal{Z}|$ and the condition $|\mathcal{Z}| << |\Omega^*|$ can be fulfilled with appropriate clustering. During exploration, each skill corresponding to $z \sim \mathcal{Z}$ is sampled with probability $p(z)$. In the execution of each skill, the trajectory τ is generated with probability $p(\tau|z)$.

Proposition 3 *Given the horizon T, every trajectory τ can be decomposed into a sequence of goal-transition patterns.*

Proof Our proof initiates by deconstructing the trajectory τ into two distinct sequences: the state sequence $S_\tau = (s_i)_{i=0}^{T}$ and the action sequence $A_\tau = (a_i)_{i=0}^{T-1}$. Upon acquiring S_τ, we derive the corresponding goal sequence $G_\tau = (\phi(s_i))_{i=0}^{T}$. This goal sequence is subsequently partitioned into its maximal homogeneous segments, each embodying repetitions of a singular unique goal. The number of such segments is denoted as $N_g(\tau)$. For each of these segments, we annotate the specific goal and the time step of its first occurrence, denoted as $((g_i, t_i))_{i=0}^{N_g(\tau)-1}$. Following this, we append the tuple $(\phi(s_T), T)$ to the sequence, resulting in $((g_i, t_i))_{i=0}^{N_g(\tau)}$. Consequently, the trajectory can be decomposed into a sequence of goal-transition patterns symbolized as $\{\psi_i\}_{i=0}^{N_g(\tau)-1}$, where each pattern ψ_i is defined as

$$\psi_i = \{s^{agent}_{t_i}, s^{agent}_{t_{i+1}}, \Delta(g_{t_i}, g_{t_{i+1}}), (a_j)_{j=t_i}^{t_{i+1}-1}\}.$$

Proposition 4 *Given an exploration horizon of T^e, the substitution of goal-transition patterns within each trajectory $\tau \in \Omega$ with alternative patterns of smaller cardinality can yield equivalent exploration outcomes using an average number of steps that is less than or equal to the specified T^e.*

Proof For any trajectory $\tau \in \Omega$, it can be decomposed into a sequence of goal-transition patterns $\{\psi_i\}_{i=0}^{N_g(\tau)-1}$ as outlined in Proposition 3. There exists an alternative goal-transition pattern to ψ_i for the transition $\Delta(g_{t_i}, g_{t_{i+1}})$ as follows: $\forall \psi_i \in \{\psi_i\}_{i=0}^{N_g(\tau)-1}, \exists \psi = \{s_{start}^{agent}, s_{end}^{agent}, \Delta G, \Delta A\} \in \Psi$, where $s_{start}^{agent} = s_{t_i}^{agent}$, $s_{end}^{agent} = s_{t_{i+1}}^{agent}$, $\Delta G = \Delta(g_{t_i}, g_{t_{i+1}})$ and $|\Delta A| \leq t_{i+1} - t_i$. By substituting ψ_i with the equivalent pattern necessitating the fewest steps, we can derive a new sequence of goal-transition patterns $\{\tilde{\psi}_i\}_{i=0}^{N_g(\tau)-1}$ such that $\sum_{i=0}^{N_g(\tau)-1} |\tilde{\psi}_i| \leq T_e$.

Appendix 2: Goal exploration

To facilitate the readability, we provide the further details omitted in Sect. 5 of the main text in this appendix, including the technical details of baselines and the state-of-the-art method LEXA explorer and the implementation details of baselines and LEXA explorer used in our experiments.

Technical details

Goal GAN

Goal GAN (Florensa et al., 2018) aims to select sub-goals of intermediate difficulties. Given the policy π_k at iteration k and a goal g, we denote its expected return as $R^g(\pi_k)$. Thus, the set of *Goals of Intermediate Difficulty* (GOID) is defined as follows:

$$\mathrm{GOID}_k \triangleq \{g : R_{min} \leq R^g(\pi_k) \leq R_{max}\}, \tag{10}$$

where R_{min} and R_{max} are the minimum and maximum expected return of goals for the agent to pursue, respectively. Also, R_{min} and R_{max} can be interpreted as the minimum and maximum success rates of reaching a goal within T steps. To identify the goals of intermediate difficulties, we adopt the same method used in Pitis et al. (2020); i.e., a discriminator is trained to distinguish whether a behavioral goal can be achieved from a specific goal. During training, the start state and the behavioral goal of each trajectory are taken as input, and the binary target would be one only if the behavioral goal was achieved within the trajectory. During goal sampling, the initial state and goal candidates are fed to a trained discriminator as input, which predicts the success probability $R^g(\pi_k)$ of reaching each candidate. Based on the prediction, the agent can decide the GOID set to be sampled from. Then, the GOID is further ranked according to how far $R^g(\pi_k)$ is close to 0.5.

Skew-fit

The key idea of Skew-Fit (Pong et al., 2020) is to increase the diversities of goals by maximising the entropy of achieved goals. Thus, Skew-Fit aims to train a generative model $q_{\phi}^{\mathcal{G}}$ that achieves maximum entropy on all the goals. To ensure its entropy is monotonically improved, it proposes to skew the distribution via sampling importance resampling as follows:

$$p_{skewed_t}(g) \triangleq \frac{1}{Z_{\alpha_1}} q_{\phi_t}^{\mathcal{G}}(g)^{\alpha_1} \delta(g \in \mathcal{G}_{\mathcal{B}}), \tag{11}$$

$$Z_{\alpha_1} = \sum_{g \in \mathcal{G}_{\mathcal{B}}} q_{\phi_t}^{\mathcal{G}}(g)^{\alpha_1}(g), g \overset{\text{iid}}{\sim} p_{\phi_t}^{\mathcal{G}}. \tag{12}$$

Here, $p_{\phi_t}^{\mathcal{G}}$ is the unknown underlying distribution of goals to be achieved via the policy at the tth iteration of training the generative model and is estimated via the approximation $p_{\phi_t}^{\mathcal{G}} \approx q_{\phi_t}^{\mathcal{G}}$. α_1 ($\alpha_1 < 0$) is used to balance the reliability of $q_{\phi_t}^{\mathcal{G}}(\mathcal{S})$ and the speed to increase the entropy of goal distribution. Then $q_{\phi_{t+1}}$ is trained to fit p_{skewed_t}, resulting in $q_{\phi_{t+1}} \approx p_{skewed_t}$. At the $(t+1)$th iteration, the goals can be sampled from p_{skewed_t} or $q_{\phi_{t+1}}$.

OMEGA

Given a distribution of desired goals p_{dg}, OMEGA (Pitis et al., 2020) aims at selecting a sub-goal that can minimize the KL divergence between p_{dg} and the distribution of achieved goals p_{ag}.

$$J_{original}(p_{ag}) = D_{KL}(p_{dg}||p_{ag}).$$

The above original learning objective is ill-conditioned and not finite for a long-horizon task since p_{ag} and p_{dg} do not overlap at the beginning. Therefore, this objective is amended via expanding the support of achieved goals to make $J_{original}(p_{ag})$ as soon as possible. It can be realized by the *Maximum Entropy Goal Achievement* (MEGA) objective that maximizes the entropy of achieved goals as follows:

$$J_{MEGA}(p_{ag}) = D_{KL}(\mathcal{U}(\text{supp}(p_{ag}))||p_{ag}).$$

where $\mathcal{U}(\text{supp}(p_{ag}))$ denotes the uniform distribution on the support of p_{ag}. Compared to MEGA, OMEGA uses a mixture distribution $p_{\alpha} = \alpha p_{dg} + (1-\alpha)\mathcal{U}(\text{supp}(p_{ag}))$ as the target in the optimization of KL divergence; i.e.,

$$J_{OMEGA}(p_{ag}) = D_{KL}(p_{\alpha}||p_{ag}).$$

The way to achieve α suggested in Pitis et al. (2020) is as follows:

$$\alpha = 1/\max(b + D_{KL}(p_{dg}||p_{ag}), 1), \tag{13}$$

where $b \leq 1$. To optimize the OMEGA objective, the agent would sample sub-goals from desired goals at α-probability and achieved goals at (1-α)-probability in the following way:

Table 1 Hyperparamters in DDPG

Hyperparamter	Value
Batch size	2000
Actor learning rate	$1e^{-3}$
Critic learning rate	$1e^{-3}$
Optimizer	Kingma & Ba (2014)
Activation	Hendrycks & Gimpel (2016)
Hidden layer sizes (actor and critic)	(512, 512, 512)
Target network update proportion	0.05
Target network update frequency	40 steps
Initial random data collection	5000 steps
Epsilon for random exploration	0.1
Replay buffer size	5,000,000
Discount factor	0.98 (0.99 for AntMaze)

Table 2 Hyperparamters for different tasks

Environment	Hyperparameter	Value
PointMaze	Relabelling strategy	rfaab_1_4_3_1_1
	train every	1
AntMaze	Relabelling strategy	rfaab_1_4_3_1_1
	train every	1
FetchPickAndPlace	Relabelling strategy	rfaab_1_5_2_1_1
	train every	4
FetchStack2	Relabelling strategy	rfaab_1_5_2_1_1
	train every	10

$$\hat{g} = \arg\min_{\hat{g}\in\mathcal{B}} p_{ag}(\hat{g}). \tag{14}$$

LEXA explorer

LEXA (Mendonca et al., 2021) is a model-based reinforcement learning algorithm with two components: explorer and achiever. The explorer acts for active exploration and trained to explore curious states via a world model. The explorer is trained with unsupervised rewards based on the disagreements of an ensemble of 1-step transition models that predict the next world model states from a current model state. The ensemble of the one-step models can be expressed as

$$\text{Ensemble: } f(s_t, \theta^m) = \hat{z}_{t+1}^m, m = 1 \dots M,$$

where $\hat{z}_{t+1}^m$ indicates the next model state predicted by model m in the ensemble of M models. Assume that there are D dimensions totally in the model state, the reward of state s is the averaged variance of the states predicted by the ensemble model across all dimensions:

$$r^e(s_t) = \frac{1}{D}\sum_{d=1}^{D} \mathrm{Var}_m[f(s_t, \theta^m)]_d. \tag{15}$$

For the achiever, we used the rewards from the environment to replace the unsupervised rewards used in Mendonca et al. (2021) for fair comparison in exploration. The achiever in our experiments is trained via the standard GCSL (Ghosh et al., 2020) in the open-source code provided by the authors where DDPG is used in the baselines.

Implementation details

DDPG

All baselines are implemented on the basis of DDPG (Lillicrap et al., 2015). The details of relevant hyperparameters used in DDPG are listed in Table 1. The training frequency varies over different tasks as reported in Table 2.

Relabelling techniques

During training, we adopt the same relabelling strategies `rfaab` used in Pitis et al. (2020): mixing different relabelling techniques `real`, `future`, `actual`, `achieved` and `behavioral` at a fixed ratio. `Real` stands for no relabelling. `Future`, `actual`, `achieved`, `behavioral` indicate relabelling with goals from future achieved goals in the belonging trajectories, all historically desired goals, all historically achieved goals and all historically behavioral goals, respectively. Their relative ratios are used to specify the specific technique. For example, `rfaab_1_4_3_1_1` denote no relabelling on 10% data and relabelling 40% with `future`, 30% with `achieved`, 10% with `actual` goals and 10% with `behavioral`. The relabelling strategies vary in different environments (see Table 2 for details).

Goal GAN

The neural network used as the discriminator has the same architecture as that of the critic in DDPG except that the sigmoid activation is used in the output layer. The discriminator is trained with a batch of 100 trajectories sampled from the 200 most recent ones for every 250 steps. The R_{min} and R_{max} are set to 0.25 and 0.75, respectively.

SkewFit

Following the same settings in Skew-Fit (Pong et al., 2020), we empoly the β-VAE as the generative model. Both the encoder and decoder of β-VAE have two hidden layers with [400, 300] ReLU units. Its latent dimension size is set to be the same as the size of the goal in the environment. In β-VAE, we set $\beta = 10$ as 10 and the $\alpha_1 = 2.5$ in Eqs. 11 and 12. We set the batch size as 64 for training β-VAE and adopt the same training setting in Skew-Fit (Pong et al., 2020): training every 4000 steps for 1000 batches in the first 40,000 steps and every 4000 steps for 200 batches afterwards.

OMEGA

We adopt the same settings used in OMEGA (Pitis et al., 2020) as follows. We set b in Eq. 13 to be −3.0. To approximate the probability $p_{ag}(\hat{g})$ for a given $\hat{g}$ in Eq. 14, we use the kernel density estimator (KDE) (Rosenblatt, 1956) with 0.1 bandwidth and Gaussian kernel as our density model. We fit the KDE model to 10,000 normalized achieved goals sampled from the replay buffer for every optimization step.

LEXA

We adopt RSSM (Hafner et al., 2019) as the world model. There are three hidden layers with [128, 128, 64] with [400, 300] ReLU units in both the encoder and the decoder. The hidden layer size for the recurrent model is set to 128. The sizes of the deterministic state and stochastic state are 128 and 32, respectively. We use 10 one-step world models (i.e., M = 10) to construct an ensemble world model that calculates the exploration rewards specified in Eq. 15. Each component world model consists of four hidden layers where each hidden layer has 400 ELU units (Clevert et al., 2015). In the GCSL implementation, we use the same actor architecture and the same learning rate used in DDPG as shown in Table 1 where only the `future` relabelling techniques are used during training.

Appendix 3: Skill learning

In this section, we provide the information on the main hyper-parameters used in our comparative study in skill learning.

SNN4HRL and ours

The skill policy network used in SNN4HRL (Florensa et al., 2017) has two hidden layers of 64 Tanh units. The policy network is trained with TRPO (Schulman et al., 2015) with learning rate 0.01 and batch size 50,000 for 300 iterations. For the reward computation, we discretize the goal space into grids of size 0.2×0.2 to calculate the rewards. Our skill learning method shares the same hyper-parameters with SNN4HRL methods except for the entropy term $\mathcal{H}\Leftarrow\mathcal{G}\clubsuit\mathcal{Z}\Rightarrow$ weighted by 0.1.

EDL

EDL (Campos et al., 2020) consists of state marginal matching (SMM) (Lee et al. 2019), VQ-VAE (Van Den Oord et al., 2017) and skill learning. We adopt the same hyper-parameters and learning methods used in the original square maze environments (Campos et al., 2020). Nevertheless, to adapt it to the `Ant` environments, we increase the environment steps per cycle to 30 and batch size to 1024 in SMM and set the number of epochs as 100 for VQ-VAE training and the number of rollouts per cycle as 6. Finally, the training epochs for skill training is set to 10.

Acknowledgements The authors would like to express our gratitude to anonymous reviewers for their comments that improve the presentation of our manuscript.

Author contributions LW: Methodology, Visualization, Software, Writing—original draft. KC: Conceptualization, Resources, Supervision, Validation, Writing—review & editing.

Funding Lisheng Wu's research was supported by the UoM-CSC scholarship.

Data availability Open-sourced benchmarks were used in our work.

Code availability Our source code is available at: https://github.com/GEAPS/GEAPS.

Declarations

Conflict of interest This research was fully conducted at The University of Manchester (email domain: `manchester.ac.uk`). Therefore, all authors certify that they have no affiliations with or involvement in any other organization or entity with any financial interest or non-financial interest in the subject matter or materials discussed in this manuscript.

References

Campero, A., Raileanu, R., Küttler, H., Tenenbaum, J. B., Rocktäschel, T., & Grefenstette, E. (2020). Learning with AMIGo: Adversarially motivated intrinsic goals. arXiv preprint arXiv:2006.12122

Campos, V., Trott, A., Xiong, C., Socher, R., Giró-i-Nieto, X., & Torres, J. (2020). Explore, discover and learn: Unsupervised discovery of state-covering skills. In *International conference on machine learning* (pp. 1317–1327). PMLR.

Clevert, D.-A., Unterthiner, T., & Hochreiter, S. (2015). Fast and accurate deep network learning by exponential linear units (elus). arXiv preprint arXiv:1511.07289

Ecoffet, A., Huizinga, J., Lehman, J., Stanley, K. O., Clune, J. (2019). Go-explore: A new approach for hard-exploration problems. arXiv preprint arXiv:1901.10995

Eysenbach, B., Gupta, A., Ibarz, J., & Levine, S. (2018). Diversity is all you need: Learning skills without a reward function. arXiv preprint arXiv:1802.06070

Florensa, C., Duan, Y., & Abbeel, P. (2017). Stochastic neural networks for hierarchical reinforcement learning. arXiv preprint arXiv:1704.03012

Florensa, C., Held, D., Geng, X., & Abbeel, P. (2018). Automatic goal generation for reinforcement learning agents. In *International conference on machine learning* (pp. 1515–1528). PMLR.

Fruit, R., & Lazaric, A. (2017). Exploration-exploitation in MDPs with options. In *Artificial intelligence and statistics* (pp. 576–584). PMLR.

Gehring, J., Synnaeve, G., Krause, A., & Usunier, N. (2021). Hierarchical skills for efficient exploration. *Advances in Neural Information Processing Systems, 34*, 11553–11564.

Ghosh, D., Gupta, A., Reddy, A., Fu, J., Devin, C. M., Eysenbach, B., & Levine, S. (2020). Learning to reach goals via iterated supervised learning. In *International conference on learning representations*.

Hafner, D., Lillicrap, T., Fischer, I., Villegas, R., Ha, D., Lee, H., & Davidson, J. (2019). Learning latent dynamics for planning from pixels. In *International conference on machine learning* (pp. 2555–2565). PMLR.

Hartikainen, K., Geng, X., Haarnoja, T., & Levine, S. (2020). Dynamical distance learning for semi-supervised and unsupervised skill discovery. arXiv preprint arXiv:1907.08225

Hendrycks, D., & Gimpel, K. (2016). Gaussian error linear units (GELUs). arXiv preprint arXiv:1606.08415

Hoang, C., Sohn, S., Choi, J., Carvalho, W., & Lee, H. (2021). Successor feature landmarks for long-horizon goal-conditioned reinforcement learning. *Advances in Neural Information Processing Systems, 34*, 26963–26975.

Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., Tunyasuvunakool, K., Bates, R., Žídek, A., Potapenko, A., & Bridgland, A. (2021). Highly accurate protein structure prediction with AlphaFold. *Nature, 596*(7873), 583–589.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980

Konidaris, G. D., & Barto, A. G. (2007). Building portable options: Skill transfer in reinforcement learning. In *International joint conference on aritificial intelligence* (vol. 7, pp. 895–900).

Lee, L., Eysenbach, B., Parisotto, E., Xing, E., Levine, S., & Salakhutdinov, R. (2019). Efficient exploration via state marginal matching. arXiv preprint arXiv:1906.05274

Levine, S., Finn, C., Darrell, T., & Abbeel, P. (2016). End-to-end training of deep visuomotor policies. *Journal of Machine Learning Research, 17*(1), 1334–1373.

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

Liu, M., Zhu, M. Z., & Zhang, W. (2022). Goal-conditioned reinforcement learning: Problems and solutions. In *International joint conference on artificial intelligence (IJCAI-22)* (pp. 5502–5511).

Mendonca, R., Rybkin, O., Daniilidis, K., Hafner, D., & Pathak, D. (2021). Discovering and achieving goals with world models. *Advances in Neural Information Processing Systems, 34*, 24379–24391.

Nair, A., McGrew, B., Andrychowicz, M., Zaremba, W., & Abbeel, P. (2018). Overcoming exploration in reinforcement learning with demonstrations. In *IEEE international conference on robotics and automation* (pp. 6292–6299). IEEE.

Pitis, S., Chan, H., & Zhao, S. (2020). mrl: modular RL. GitHub.

Pitis, S., Chan, H., Zhao, S., Stadie, B., & Ba, J. (2020). Maximum entropy gain exploration for long horizon multi-goal reinforcement learning. In *International conference on machine learning* (pp. 7750–7761). PMLR.

Plappert, M., Andrychowicz, M., Ray, A., McGrew, B., Baker, B., Powell, G., Schneider, J., Tobin, J., Chociej, M., Welinder, P., & Kumar, V. (2018). Multi-goal reinforcement learning: Challenging robotics environments and request for research. arXiv preprint arXiv:1802.09464

Pong, V. H., Dalal, M., Lin, S., Nair, A., Bahl, S., & Levine, S. (2020). Skew-fit: State-covering self-supervised reinforcement learning. In: *International conference on machine learning*. PMLR.

Rosenblatt, M. (1956). Remarks on some nonparametric estimates of a density function. *The Annals of Mathematical Statistics, 27*, 832–837.

Schaul, T., Horgan, D., Gregor, K., & Silver, D. (2015). Universal value function approximators. In *International conference on machine learning* (pp. 1312–1320). PMLR.

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

Sharma, A., Gu, S., Levine, S., Kumar, V., & Hausman, K. (2019). Dynamics-aware unsupervised discovery of skills. arXiv preprint arXiv:1907.01657

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., & Chen, Y. (2017). Mastering the game of go without human knowledge. *Nature, 550*(7676), 354–359.

Silver, D., Singh, S., Precup, D., & Sutton, R. S. (2021). Reward is enough. *Artificial Intelligence, 299*, 103535.

Sutton, R. S. (1998). Between MDPs and semi-MDPs: Learning, planning, and representing knowledge at multiple temporal scales.

Trott, A., Zheng, S., Xiong, C., & Socher, R. (2019). Keeping your distance: Solving sparse reward tasks using self-balancing shaped rewards. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems* (pp. 10376–10386) Curran Associates Inc: Red Hook, NY.

Van Den Oord, A., Vinyals, O., & Kavukcuoglu, K. (2017). Neural discrete representation learning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems* (pp. 6309–6318) Curran Associates Inc: Long Beach, CA