MHER: Model-based Hindsight Experience Replay

Rui Yang*[1], Meng Fang[2,3], Lei Han[2], Yali Du[4], Feng Luo[1], Xiu Li[1]
[1]Tsinghua University, [2]Tencent Robotics X
[3]Eindhoven University of Technology, [4]King's College London
`yangrui19@mails.tsinghua.edu.cn, m.fang@tue.nl, lxhan@tencent.com`
`yali.du@kcl.ac.uk, {luof19@mails, li.xiu@sz}.tsinghua.edu.cn`

Abstract

Solving multi-goal reinforcement learning (RL) problems with sparse rewards is generally challenging. Existing approaches have utilized goal relabeling on collected experiences to alleviate issues raised from sparse rewards. However, these methods are still limited in efficiency and cannot make full use of experiences. In this paper, we propose *Model-based Hindsight Experience Replay* (MHER), which exploits experiences more efficiently by leveraging environmental dynamics to generate virtual achieved goals. Replacing original goals with virtual goals generated from interaction with a trained dynamics model leads to a novel relabeling method, *model-based relabeling* (MBR). Based on MBR, MHER performs both reinforcement learning and supervised learning for efficient policy improvement. Theoretically, we also prove the supervised part in MHER, i.e., goal-conditioned supervised learning with MBR data, optimizes a lower bound on the multi-goal RL objective. Experimental results in several point-based tasks and simulated robotics environments show that MHER achieves significantly higher sample efficiency than previous model-free and model-based multi-goal methods.

1 Introduction

Although reinforcement learning (RL) has been shown to be effective in a range of reward-driven problems [20, 18, 13, 19], current RL algorithms require massive amounts of training data [33] and lack sample efficiency in sparse reward settings [1]. In multi-goal RL, the problem of efficiency becomes more prominent as agents are required to accomplish multiple goals simultaneously. One of the most essential factors affecting sample efficiency in multi-goal RL is the sparse reward, in which case informative learning signals are very limited. Previous works have proposed many solutions such as reward shaping [22], curriculum learning [3], exploration [23, 34], and hindsight relabeling [16, 1]. Among those solutions, learning from mistakes is a useful strategy to handle sparse rewards in multi-goal RL settings. Hindsight Experience Replay (HER) [1] remarkably improves sample efficiency through goal-relabeling that relabels failed experiences with actually achieved goals. Following HER, a few works are put forward to improve goal sampling methods [9, 24, 17, 8], or utilize hindsight knowledge for supervised policy learning [31, 12] and adversarial imitation learning [7].

Goal relabeling provides a practical paradigm for generating pseudo demonstrations to train control policies, and deep reinforcement learning algorithms further improve upon the efficiency of relabeling strategies [16, 1]. Most goal relabeling methods depend on trajectories and goals that an agent collects from environments. However, intelligent agents can achieve goals in complex environments even though they never encounter the exact same situation [26]. This ability requires building representations of the dynamics from past experience that enable generalization to novel situations [27, 30]. Modeling dynamics offers an explicit way to represent an agent's knowledge about the task.

*Work was done during the internship with Tencent Robotics X.

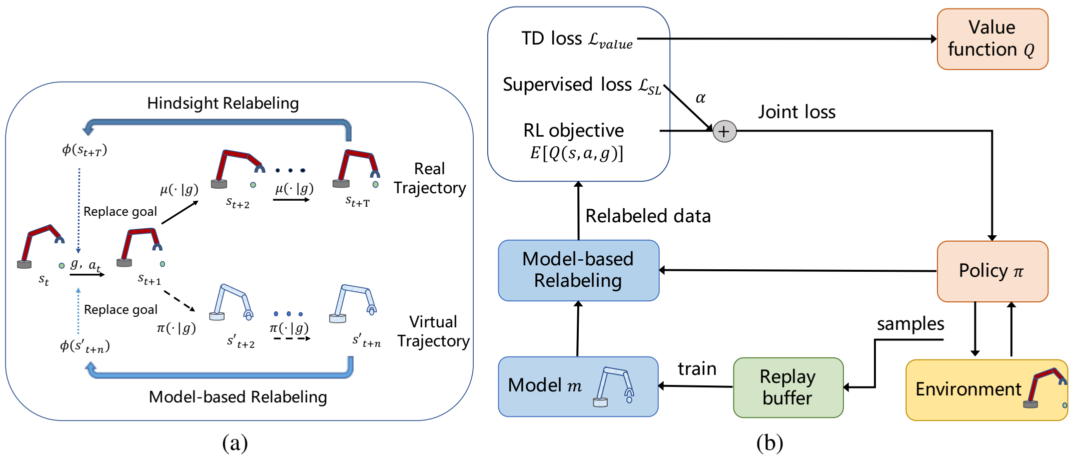


Figure 1: (a) Diagram of model-based relabeling. The real trajectory is collected by a past behavior policy μ and the virtual trajectory is generated from the interaction of the current policy π and a trained dynamics model. Hindsight relabeling such as HER uses real achieved goals (e.g., $\phi(s_{t+T})$, ϕ is a state-to-goal mapping) to relabel, while model-based relabeling utilizes virtual achieved goals (e.g., $\phi(s'_{t+n})$). (b) Overall framework of MHER. Detailed descriptions can be found in Section 4.4.

Existing methods [21, 15, 14, 28] have apply neural models to greatly facilitate predicting physical dynamics and the consequences of actions, and provide a strong inductive bias for generalization to novel environment situations. With the dynamics model and the current policy, we can predict future states along with achieved goals. Can we exploit goals generated from model-based interaction for sample efficient multi-goal reinforcement learning?

In this paper, we propose a novel framework, *Model-based Hindsight Experience Replay* (MHER), utilizing environmental dynamics to handle sparse rewards. In MHER, we introduce a new relabeling method, *model-based relabeling*, and then minimize a joint loss based on the model-based relabeled data for efficient policy learning. Unlike previous hindsight relabeling methods that relabel transitions with goals achieved at a later point during the same trajectory, model-based relabeling leverages dynamics models to generate pseudo goals for guiding the learning of the policy, as shown in Figure 1. The pseudo goals are collected in an efficient way without interaction with environments. Then goal-relabeling through imagined trajectories allows an agent to re-interpret its actions using a different goal from the perspective of the latest policy, leading to an implicit curriculum of goal relabeling guided by the policy. With the model-based relabeled data, we apply both supervised learning and reinforcement learning to update the policy. We theoretically prove that the policy can be improved by minimizing the goal-conditioned supervised learning loss [12] with the model-based relabeled data. To evaluate the performance of MHER, we conduct experiments on both point-based and Mujoco environments. Experimental results [2] show that MHER achieves significantly higher sample efficiency than previous works such as HER, Curriculum-guided HER [9], Maximum Entropy-based Prioritization [34], and Goal-Conditioned Supervised Learning [12].

The main contributions of this paper can be summarized as follows:

- We present a new goal relabeling method, model-based relabeling, leveraging dynamics models to handle sparse rewards in multi-goal RL;
- We apply supervised learning on the relabeled data and introduce a joint loss for RL training. We also prove that minimizing the supervised loss using the model-based relabeled data is equivalent to optimizing a lower bound on the original multi-goal RL objective;
- Empirical results on several benchmark environments demonstrate that the proposed method, MHER, exceeds previous multi-goal RL algorithms in sample efficiency.

[2] https://github.com/YangRui2015/Model-basedHER.

2 Related Work

Our work concentrates on sample efficiency in multi-goal reinforcement learning (RL) with sparse rewards. Hindsight Experience Replay (HER) [1] introduces hindsight relabeling for multi-goal RL, opening up a new way to learn from failed experiences with sparse and binary rewards. Based on HER, a few studies have been investigated to find more efficient goal sampling ways. Curriculum-guided HER (CHER) [9] selects goals in a heuristic way to balance the diversity of selected goals and the proximity to original goals. Focusing on the long horizon problem in multi-goal RL, Maximum Entropy Goal Achievement [24] samples from the frontier of achieved goals and gradually increases the entropy of achieved goals. Recent woks [17, 8] view hindsight relabeling as the inverse RL and sample relabeling goals according to their cumulative return or value function. Considering the visual tasks, [25] leverages GAN to generate virtual goals for relabeling, but it requires collecting a special dataset of near-goal states. In contrast to previous works, relabeling goals of MHER are generated from the interaction between current policy and a learned dynamics model.

In addition to improving the goal sampling methods, other works introduce prioritization replay and supervised learning to address sample inefficiency in multi-goal RL. Energy-Based Prioritization proposes to more frequently replay trajectories with higher energy. Maximum Entropy-based Prioritization (MEP) [34] prioritizes experiences based on trajectory entropy. By incorporating a few expert demonstrations, *goalGAIL* [7] significantly speeds up the convergence of policy via adversarial imitation learning. Self-supervised methods provides another simple but effective way for multi-goal RL. Policy Continuation with Hindsight Inverse Dynamics (PCHID) [31] extends hindsight inverse dynamics to the multi-step situation following dynamic programming, thus enabling learning in a self-imitated scheme. Goal-Conditioned Supervised Learning (GCSL) [12] provides theoretical guarantees that supervised learning from hindsight relabeled experiences optimizes a lower bound on the goal-oriented RL objective. Unlike PCHID and GCSL which both use real achieved goals for supervision, we leverage virtual goals for supervision and provide theoretical guarantees that supervised learning with virtual goals can lead to policy improvement.

Model-based RL algorithms have been studied for a long history and generally obtain higher sample efficiency over model-free algorithms [2]. Dyna [32] generates virtual samples with a trained dynamics model to augment training data. Learned dynamics models can also be incorporated into model-free algorithms to accelerate learning of policies and value functions [21]. Model-based value expansion (MVE) [10] improves model-free value estimation with predictive transitions. To tackle with the model bias in learned models, STEVE [5] uses an ensemble of models for a robust prediction. Model-Based Policy Optimization (MBPO) [15] proves a monotonic improvement with limited use of a predictive model. However, most of the model-based methods are designed for tasks with dense rewards. PlanGAN [6] is the first algorithm to use GANs and models for planning in sparse-reward multi-goal tasks. But PlanGAN is a planning method and requires a huge amount of computation to simulate trajectories for selecting a single action, and it may also suffer from the accumulated model-based errors in the long planning horizon. Different from these works, our work only utilizes the virtual achieved goals to avoid training with full virtual states and alleviate the impact of model bias.

3 Preliminaries

3.1 Multi-goal Reinforcement Learning

Multi-goal reinforcement learning (RL) can be characterized by the tuple $\langle S, A, G, r, p, \gamma \rangle$, where S, A, G, p, γ refer to state space, action space, goal space, transition function, and discount factor, respectively, and $r : S \times A \times G \to \mathbb{R}$ is the goal-conditioned reward function [26]. A commonly used sparse reward function in multi-goal setting is defined as:

$$r(s_t, a_t, g) = \begin{cases} 0, & ||\phi(s_t) - g||_2^2 < \text{threshold} \\ -1, & \text{otherwise} \end{cases}, \tag{1}$$

where $\phi : S \to G$ is a mapping function from states to achieved goals and is assumed to be available [1]. Agents are required to learn a policy $\pi : S \times G \to A$ to maximize returns of reaching goals from the desired distribution $p(g)$:

$$J(\pi) = E_{g\sim p(g), a_t\sim\pi, s_{t+1}\sim p(\cdot|s_t,a_t)}\Big[\sum_{t=0}^{\infty}\gamma^t r(s_t,a_t,g)\Big] \tag{2}$$

3.2 Hindsight Experience Replay

Hindsight Experience Replay (HER) learns from failed experiences and tackles with sparse rewards in multi-goal RL. Given a trajectory $\tau = \{(s_t, a_t, g, r_t, s_{t+1})\}_{t=1}^{T}$ of length T, HER alternates g and r_t in the t-th transition $(s_t, a_t, g, r_t, s_{t+1})$ with a future achieved goal in the same trajectory $g' = \phi(s_{t+k}), 1 \le k \le T-t$ and $r'_t = r(s_t, a_t, g')$ computed by Eq. (1). After relabeling, transitions in failed trajectories can be assigned nonnegative rewards, therefore HER addresses the core issue of sparse rewards. HER can be combined with any off-policy algorithms such as DQN [20], DDPG [18], TD3 [11], and SAC [13].

In our paper, we adopt the DDPG+HER framework following [1, 9]. DDPG is an off-policy actor-critic algorithm consisting of a deterministic policy π and a value function $Q : S\times A\times G \to \mathbb{R}$. When collecting data with the policy π, Gaussian noise with zero mean and constant standard deviation is applied for exploration, as implemented in [11]. We denote the data distribution after hindsight relabeling as B_h. The value function Q is updated to minimize the TD error:

$$\mathcal{L}_{critic} = E_{(s_t,a_t,g',r'_t,s_{t+1})\sim B_h}\big[(y_t - Q(s_t,a_t,g'))^2\big], \tag{3}$$

where

$$y_t = r'_t + \gamma Q(s_{t+1}, \pi(s_{t+1}, g'), g').$$

The policy π is trained with policy gradient on the following loss:

$$\mathcal{L}_{actor} = -E_{(s_t,g')\sim B_h}\big[Q(s_t, \pi(s_t,g'), g')\big].$$

4 Methodology

In this section we will describe how our method, MHER, integrates environment dynamics into Hindsight Experience Replay [1] for training goal-conditioned policies. First we introduce the way we train the dynamics model that maps current state and action to the next state [32, 10]. Then we propose a novel model-based relabeling technique based on the learned model and introduce a joint loss combining policy gradient [29, 18] and goal-conditioned supervised learning [12] on the model-based relabeled data. Finally, we describe the overall framework of MHER.

4.1 Dynamics Models

When considering deterministic dynamics, the most direct way of training dynamics models is to learn to predict the next state given current state and action. Let $m(s_t, a_t)$ denote a learned dynamics function that takes the current state s_t and action a_t as input and outputs an estimation of the next state s_{t+1}. However, this model can be difficult to learn when the states s_t and s_{t+1} are very similar and the action has seemingly little effect on the changes in continuous control environments [21]. Therefore we follow [21] to learn a dynamics function that predicts the change between states by minimizing the following loss:

$$\mathcal{L}_{model} = E_{(s_t,a_t,s_{t+1})\sim B}\big[\|(s_{t+1}-s_t) - m(s_t,a_t)\|_2^2\big], \tag{4}$$

where the data (s_t, a_t, s_{t+1}) is sampled from the replay buffer B. Note that the dynamics model is independent of goals and rewards, therefore the data to train the model m can also be sampled from the relabeled data distribution such as B_h. With the trained dynamics model, current state s_t and action a_t, we can predict the next state as $s_{t+1} = s_t + m(s_t, a_t)$.

4.2 Model-based Relabeling

The insight of model-based relabeling (MBR) is that states in the virtual trajectory generated by the dynamics model can also be viewed as achieved goals for the starting state, so it can be used for relabeling the original transition. As shown in Figure 1, given a transition $(s_t, a_t, s_{t+1}, r_t, g)$ collected

Algorithm 1: MHER Framework

```
1  Given model-based interaction steps $n$, and $\alpha$ in the joint policy loss $\mathcal{L}_{joint}$;
2  Initialize policy $\pi$ and value function $Q$;
3  Warmup the dynamics model $m$ by minimizing $\mathcal{L}_{model}$ in Eq. (4) with random samples;
4  for episode = $1, 2, \ldots, M$ do
5      Sample a desired goal $g$;
6      Collect a trajectory with the policy $\pi$ and save to the replay buffer $B$;
7      Sample a minibatch $b$ from the replay buffer : $\{(s_t, a_t, s_{t+1}, r_t, g)_i\}_{i=1}^N \sim B$;
8      Update the dynamics model $m$ with $b$;
9      for $i = 1, 2, \ldots, N$ do
10         $(s_t, a_t, s_{t+1}, r_t, g) \leftarrow b_i$;
           // model-based relabeling
11         $s'_{t+1} = s_{t+1}, v = \{s'_{t+1}\}$ ;
12         for $j = 1, 2, \ldots, n$ do
13             $a'_{t+j} = \pi(s'_{t+j}, g)$;
14             $s'_{t+j+1} = s'_{t+j} + m(s'_{t+j}, a'_{t+j})$;
15             Append $s'_{t+j+1}$ to $v$ ;
16         end for
17         Sample random future state $s \sim v$;
18         Get virtual achieved goal $g' = \phi(s)$;
19         Recompute $r' = r(s_t, a_t, g')$ using reward function in Eq. (1);
20         $b_i \leftarrow (s_t, a_t, r', s_{t+1}, g')$;
21     end for
22     Update value function $Q$ with $b$ to minimize $\mathcal{L}_{value}$ in Eq. (7);
23     Update policy $\pi$ with $b$ to minimize $\mathcal{L}_{joint}$ in Eq. (6) ;
24 end for
```

by a behavior policy, MBR starts at s_{t+1} and interacts with the dynamics model m using current policy π for n steps. After interaction with the model, we have a virtual trajectory of $\{s'_{t+i}, a'_{t+i}, s'_{t+i+1}\}_{i=0}^n$, where $i = 0$ is the original transition and $a'_{t+i} = \pi(s'_{t+i}, g), s'_{t+i+1} = s'_{t+i} + m(s'_{t+i}, a'_{t+i}), 1 \leq i \leq n$. Then, any state in the virtual trajectory implies an achieved goal guided by current policy. MBR randomly samples from the virtual achieved goals $g' = \phi(s'_{t+j}), 1 \leq j \leq n$ to relabel the original transition: $(s_t, a_t, s_{t+1}, r'_t, g')$, where ϕ is the state-to-goal mapping and $r'_t = r(s_t, a_t, g')$ is computed according to Eq. (1). In our relabeled transitions, only the goal g' is virtually generated and other items are real experiences, avoiding the usage of full virtual states for training the value function. More detailed descriptions are provided in Appendix C. Note that the model-based interaction is driven by the current policy π under original goals g, thus MBR gradually pushes the relabeling goals towards the desired goal as the policy π improves. Theoretically, optimizing $J(\pi)$ is equivalent to minimizing an upper bound on the expected distance between virtual achieved goals and original desired goals. The proof can be found in Appendix A.3. We will also verify this property empirically in Section 5.4.

Two major advantages of model-based relabeling over previous relabeling strategies are as follows:

(1) MBR takes advantages of the current policy and environmental dynamics to generate more diverse goals for accelerating policy learning.

(2) As the policy improves, relabeling goals will gradually approach the assigned targets, therefore an implicit curriculum of automatic relabeling is introduced.

4.3 Learning Policy with Model-based Relabeled Data

After model-based relabeling (MBR), we further perform policy update based on the relabeled data. Similar to [12], the policy can be optimized through supervised learning on the model-based relabeled data. However, the difference is that we use virtually generated goals g' for supervision and minimize a mean square error rather than optimize the maximum likelihood, which are substantially the same under certain assumptions. We denote the model-based relabeled transition distribution as B_m, then the supervised loss $\mathcal{L}_{SL}$ is defined as:

$$\mathcal{L}_{SL} = E_{(s_t,a_t,g')\sim B_m}\left[\|a_t - \pi(s_t, g')\|_2^2\right], \tag{5}$$

where g' is relabeled using MBR. Theoretically, we prove that minimizing $\mathcal{L}_{SL}$ in Eq. (5) is equivalent to maximizing a lower bound of $J(\pi)$, which is formally presented below.

Theorem 1. *Given a Diagonal Gaussian policy with a mean vector $\pi(s, g')$ and a non-zero positive constant variance σ^2, the discount factor γ, the real environmental dynamics $p(\cdot|s, a)$, a learned dynamics model $p_m(\cdot|s, a)$, the model-based relabeled data distribution B_m, and n-step model-based interactions, minimizing the supervised loss $\mathcal{L}_{SL}$ in Eq. (5) is equivalent to maximizing a lower bound on the multi-goal RL objective.*

$$J(\pi) \geq -\frac{n\gamma^n}{2\sigma^2}\mathcal{L}_{SL} + C_1\epsilon_m + C_2,$$

where C_1, C_2 are two constants independent of policy π. ϵ_m is the model error bounded at each timestep t: $\epsilon_m = max_t E_{s\sim\pi_{D,t}}[D_{TV}(p(s'|s, a)\|p_m(s'|s, a))]$, and π_D is the data collecting policy of B_m.

The detailed proof is provided in Appendix A.1.

Previous works update policy via either policy gradient [1, 9] or supervised learning [12] with the hindsight relabeled data introduced in Sec 3.2. In contrast to these works, we propose a joint loss combining policy gradient and the supervised loss $\mathcal{L}_{SL}$ for more efficient policy learning. Specifically, at each iteration during training, the policy is trained to minimize the following joint loss on the model-based relabeled data B_m:

$$\begin{aligned}\mathcal{L}_{joint} = &-\mathbb{E}_{(s_t,g')\sim B_m}\left[Q(s_t, \pi(s_t, g'), g')\right] \\ &+ \alpha\mathbb{E}_{(s_t,a_t,g')\sim B_m}\left[\|a_t - \pi(s_t, g')\|_2^2\right],\end{aligned} \tag{6}$$

where $\alpha > 0$ is the weight balancing the expected return and the supervised loss $\mathcal{L}_{SL}$. The Q-function is updated to minimize the TD error on the model-based relabeled data B_m:

$$\mathcal{L}_{value} = E_{(s_t,a_t,g',r'_t,s_{t+1})\sim B_m}\left[(y_t - Q(s_t, a_t, g'))^2\right], \tag{7}$$

where $y_t = r'_t + \gamma Q(s_{t+1}, \pi(s_{t+1}, g'), g')$.

4.4 Algorithm

The overall framework of model-based hindsight experience replay (MHER) is presented in Algorithm 1 and Figure 1 (b). First, the dynamics model m is trained to minimize the loss $\mathcal{L}_{model}$ using Eq. (4) in both warm-up and training periods. For every episode, we sample a goal g from the desired goal distribution and collect a trajectory to the replay buffer B using current policy. After collecting data, we sample a minibatch b from the replay buffer B. For each transition in the minibatch, we leverage the current policy π to rollout a n-step trajectory with the learned dynamics model m and perform model-based relabeling as explained in Section 4.2. After MBR, the minibatch b belongs to the model-based relabeled distribution B_m and is sent for training the Q network and the policy network. The Q network is updated according to Eq. (7) and the policy is trained to minimize the joint loss $\mathcal{L}_{joint}$ in Eq. (6).

5 Experiments

We conduct experiments on both continuous Point2D and Mujoco environments and compare the performance of MHER against a number of leading multi-goal RL algorithms for sparse reward environments. We also demonstrate the effectiveness of our goal-relabeling method by visualizing the distribution of relabeling goals.

5.1 Experimental Settings

Environments Our experimental environments consist of two Point2D environments, one Sawyer robots, and two Mujoco robots modified from OpenAI Gym [4]. All of the five environments' states, actions, and goals are continuous. In the first two point-based environments the blue point aims to reach the green circle. The other four environments (FetchReach-v1, SawyerReachXYZEnv-v1, Reacher-v2) control a robot to reach a target position in the goal space. More detailed task description can be found in Appendix F.

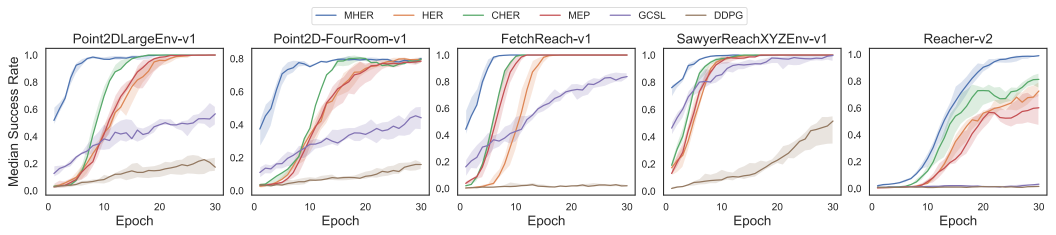


Figure 2: Median test success rate (line) with interquartile range (shaded area) on Point2D and Mujoco environments acorss 10 random seeds.

Baseline Implementation All the implementations of baseline algorithms are taken from their open-source code except GCSL, which we implement a deterministic version to fairly compare with other algorithms. For GCSL [12], we only maintain a policy network and train it to minimize the loss $\mathcal{L}_{GCSL} = E_{(s_t,a_t,g')\sim B_h}\left[\|a_t - \pi(s_t, g')\|_2^2\right]$, where g' is relabeled using future achieved goals similar to HER and B_h refers to the data distribution after hindsight relabeling.

Implementation of MHER As for the implementation of MHER, actor and critic networks are both 3-layer fully connected networks with 256 units each layer. All the networks are updated with Adam optimizer, learning rate 1×10^{-3}, and Polyak averaging coefficient of 0.9. To encourage exploration, the probability of random action is set as 0.3, and the scale of Gaussian noisy is 0.2. Following [1, 9], we perform model-based relabeling with a probability of 0.8. Note that the portion of data without MBR is not sent for goal-conditioned supervised learning, which can be implemented with a mask.

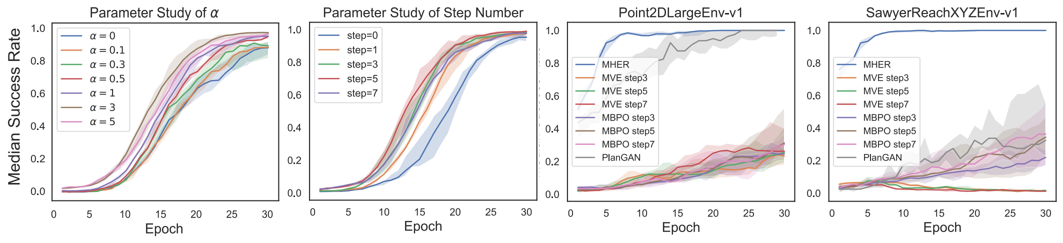
Training Settings We train all the algorithms for 30 epochs with one rollout worker. Each epoch contains 1 (Point2DLargeEnv-v1, Point2D-FourRoom-v1), 5 (FetchReach-v1, SawyerReachXYZEnv-v1), or 15 (Reacher-v2) episodes according to the difficulty of the environments. Every episode contains 100 interaction steps with the environment. At the end of each episode, we train all the algorithms for 5 batches with a batch size of 64.

Evaluation Settings After each epoch, we evaluate every algorithm for 100 episodes, and report the median test success rate and the interquartile range across 5 random seeds. To train the dynamics model m for MHER, we use a fully connected network with 4 hidden layers and 256 neurons each layer. In the warmup period, we train m for 100 updates with a batch size of 512 and a learning rate of 0.001. When training with MHER, we update m with the sampled minibatch for 2 times. More detailed hyperparameters are provided in Appendix D.

5.2 Benchmark Results

We compare MHER with baselines such as DDPG, HER [1], CHER [9], MEP [34], and GCSL [12] in five benchmark environments. For MHER, the model-based interaction step is set as 5 and the parameter α in the joint loss is set as 3. The median test success rates are reported in Figure 2. It is apparent that MHER achieves significantly higher performance than those baselines with a much faster learning speed. The results also show that DDPG learns slowly in all the environments, while HER, CHER, MEP are three effective baselines. Besides, GCSL's performance in different environments is not consistent, e.g., in Reacher-v2 it is very close to DDPG but in SawyerReachXYZEnv-v1 it outperforms DDPG by a large margin.

In Figure 3 (a) and 3 (b), we study how the parameters α and model-based interaction steps n impact MHER's performance. The parameter α impacts the weight of the supervised policy loss in $\mathcal{L}_{joint}$ when optimizing the policy, and $\alpha = 0$ indicates learning without the supervised loss. The results in Figure 3 (a) suggest that as α increases, performance increases when $\alpha \leq 3$ and decreases when $\alpha > 3$. As for the model-based interaction steps, more steps may contain more long-distance information but too many steps can lead to irrelevant goals. The results in Figure 3 (b) verify our analysis and the horizon of 5 achieves the best performance in the candidate set $\{0, 1, 3, 5, 7\}$. More experimental results are provided in Appendix B.



(a) (b) (c) (d)

Figure 3: (a)(b) Parameter study of α and step number in Reacher-v2 environment. (c)(d) Comparison results with model-based based reinforcement learning baselines, MVE and MBPO, in Point2DLargeEnv-v1 and SawyerReachXYZEnv-v1 environments.

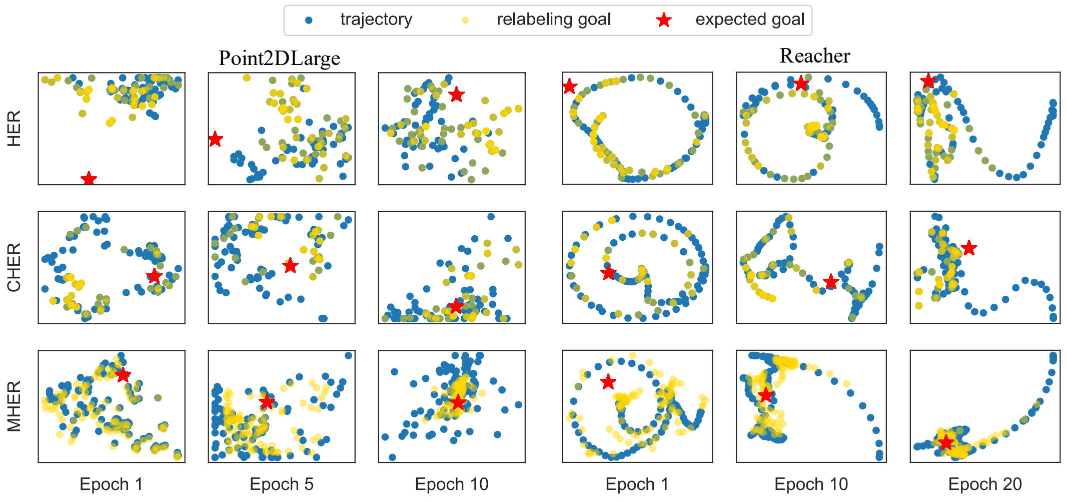


Figure 4: Relabeling goal distributions of HER, CHER, and MHER in Point2DLargeEnv-v1 and Reacher-v2. Blue points are real states in a trajectory, and red stars are their expected goals. Yellow points are relabeling goals for each transition in the trajectory.

5.3 Results of Model-based RL baselines

We also make fair comparison with model-based RL baselines in Figure 3 (c) and (d), including Model-based Value Expansion (MVE) [10] and Model-based Policy Optimization (MBPO) [15]. Implementation details of MVE and MBPO are provided in Appendix D. From the experimental results, we can conclude that previous model-based RL methods contribute little to multi-goal RL tasks with sparse rewards. In contrast to these algorithms, MHER can exploit environmental dynamics to learn policies more efficiently in the sparse reward setting. We also compare with PlanGAN [6], a planning method rather than a RL method, in Appendix B. The results show that MHER can achieve comparable or higher performance compared with PlanGAN, although PlanGAN requires extremely large amount of computations (detailed analysis is provided in Appendix B.1) to simulate trajectories for selecting actions.

5.4 Relationship with Curriculum Goal Relabeling

Prior work, Curriculum-guided HER (CHER) [9] introduces an explicit curriculum for automatically selecting relabeling goals. Nevertheless, the curriculum is hand-designed and needs to be adjusted through hyperparameters.

In Figure 4, we compare the relabeling goals of HER, CHER, and MHER in Point2DLargeEnv-v1 and Reacher-v2 environments. We can observe that HER selects goals randomly along the future achieved goals, leading to more goals in the end of trajectories. Besides, CHER selects goals to

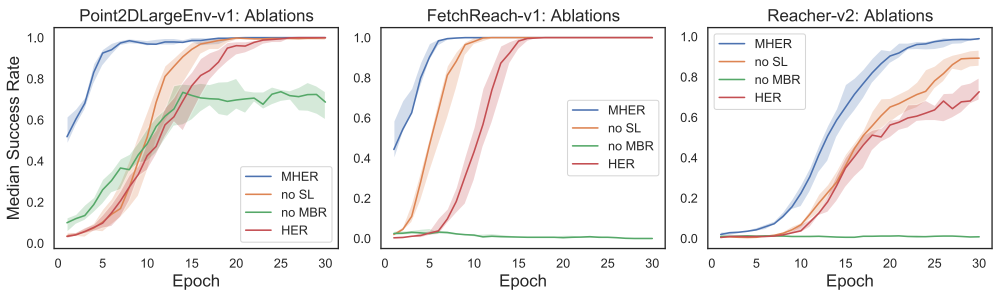


Figure 5: Ablation studies in Point2DLargeEnv-v1, FetchReach-v1, and Reacher-v2.

balance the diversity of selected goals and the proximity to the expected goal. Moreover, HER and CHER only select real achieved goals, thus the set of sampling goals is limited. On the contrary, MHER generates virtual relabeling goals according to the dynamics model and the current policy under the original desired goal. In the early stage, relabeling goals of MHER are near achieved goals. As the policy improves, they gradually approach their expected goals. The learning process of MHER is automatically adjusted by the policy, therefore MHER can substantially achieve more efficient curriculum learning.

5.5 Ablation Studies

To analyze the importance of *model-based relabeling* (MBR) and *supervised learning on the MBR data* (denoted as SL) in the MHER framework, we design ablation experiments to compare MHER variants with HER. For MBR and SL, the number of model-based interaction steps is 5 and $\alpha = 3$ by default. We experiment with the following settings:

- *MHER*: DDPG+MBR+SL;
- *no SL*: DDPG+MBR, which is equivalent to $\alpha = 0$;
- *no MBR*: DDPG+SL, adding an auxiliary task to DDPG that only uses model-based relabeled transitions to minimize the supervised policy loss.

Empirical results in Figure 5 demonstrate that MBR is more important than SL in the MHER framework. DDPG+MBR learns faster than HER, but DDPG+SL learns very slowly in FetchReach-v1 and Reacher-v2, and cannot converge to 100% success rate in Point2DLargeEnv-v1, which indicates that SL is less effective compared to MBR. By combining MBR and SL, the performance is significantly improved. The intuition behind the results is that MBR and SL achieve mutual improvement under the MHER framework, because MBR provides goals following an efficient curriculum to train the policy, and the supervised policy loss further improves the policy that guides the curriculum of MBR.

6 Conclusion

In this paper, we propose the framework of *Model-based Hindsight Experience Replay* (MHER) to improve sample efficiency in multi-goal RL with sparse rewards. In MHER, we introduce virtual goals for goal relabeling and policy improvement, corresponding to model-based relabeling (MBR) and a joint loss combining policy gradient and supervised learning with MBR data. Experiments in a range of continuous multi-goal tasks demonstrate that MHER achieves significantly higher sample efficiency than previous goal-conditioned works such as HER, CHER, MEP and GCSL. Moreover, we show that MHER is efficient in the following aspects: (1) virtual goals generated from interaction with the trained model are not limited to real experiences; (2) generated goals follow an efficient curriculum guided by the policy; (3) policy learning takes advantage of both reinforcement learning and supervised improvement. In the future, we would like to work on more efficient works inspired by MHER.

References

[1] Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, OpenAI Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. In *Advances in Neural Information Processing Systems*, pages 5048–5058, 2017.

[2] Christopher G Atkeson and Juan Carlos Santamaria. A comparison of direct and model-based reinforcement learning. In *IEEE International Conference on Robotics and Automation*, volume 4, pages 3557–3564, 1997.

[3] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *International Conference on Machine Learning*, pages 41–48, 2009.

[4] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

[5] Jacob Buckman, Danijar Hafner, George Tucker, Eugene Brevdo, and Honglak Lee. Sample-efficient reinforcement learning with stochastic ensemble value expansion. In *NeurIPS*, 2018.

[6] Henry Charlesworth and Giovanni Montana. Plangan: Model-based planning with sparse rewards and multiple goals. In *Advances in Neural Information Processing Systems*, 2020.

[7] Yiming Ding, Carlos Florensa, Pieter Abbeel, and Mariano Phielipp. Goal-conditioned imitation learning. *Advances in Neural Information Processing Systems*, 32:15324–15335, 2019.

[8] Ben Eysenbach, Xinyang Geng, Sergey Levine, and Russ R. Salakhutdinov. Rewriting history with inverse RL: hindsight inference for policy improvement. In *Advances in Neural Information Processing Systems*, 2020.

[9] Meng Fang, Tianyi Zhou, Yali Du, Lei Han, and Zhengyou Zhang. Curriculum-guided hindsight experience replay. In *Advances in Neural Information Processing Systems*, pages 12623–12634, 2019.

[10] Vladimir Feinberg, Alvin Wan, Ion Stoica, Michael I Jordan, Joseph E Gonzalez, and Sergey Levine. Model-based value expansion for efficient model-free reinforcement learning. In *Proceedings of the 35th International Conference on Machine Learning (ICML 2018)*, 2018.

[11] Scott Fujimoto, Herke Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International Conference on Machine Learning*, pages 1587–1596, 2018.

[12] Dibya Ghosh, Abhishek Gupta, Ashwin Reddy, Justin Fu, Coline Manon Devin, Benjamin Eysenbach, and Sergey Levine. Learning to reach goals via iterated supervised learning. In *International Conference on Learning Representations*, 2021.

[13] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning*, pages 1861–1870, 2018.

[14] Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations*, 2019.

[15] Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to trust your model: Model-based policy optimization. In *Advances in Neural Information Processing Systems*, pages 12498–12509, 2019.

[16] Leslie Pack Kaelbling. Learning to achieve goals. In *International Joint Conference on Artificial Intelligence*, pages 1094–1099, 1993.

[17] Alexander C. Li, Lerrel Pinto, and Pieter Abbeel. Generalized hindsight for reinforcement learning. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

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

[19] Jiafei Lyu, Xiaoteng Ma, Jiangpeng Yan, and Xiu Li. Efficient continuous control with double actors and regularized critics. *CoRR*, abs/2106.03050, 2021.

[20] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

[21] Anusha Nagabandi, Gregory Kahn, Ronald S Fearing, and Sergey Levine. Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning. In *IEEE International Conference on Robotics and Automation*, pages 7559–7566, 2018.

[22] Andrew Y Ng, Daishi Harada, and Stuart Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *International Conference on Machine Learning*, volume 99, pages 278–287, 1999.

[23] Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *International Conference on Machine Learning*, pages 2778–2787. PMLR, 2017.

[24] Silviu Pitis, Harris Chan, Stephen Zhao, Bradly Stadie, and Jimmy Ba. Maximum entropy gain exploration for long horizon multi-goal reinforcement learning. In *International Conference on Machine Learning*, pages 7750–7761, 2020.

[25] Himanshu Sahni, Toby Buckley, Pieter Abbeel, and Ilya Kuzovkin. Addressing sample complexity in visual tasks using HER and hallucinatory gans. In *Advances in Neural Information Processing Systems*, pages 5823–5833, 2019.

[26] Tom Schaul, Daniel Horgan, Karol Gregor, and David Silver. Universal value function approximators. In *International Conference on Machine Learning*, pages 1312–1320, 2015.

[27] Tom Schaul and Mark B Ring. Better generalization with forecasts. In *IJCAI*, pages 1656–1662, 2013.

[28] Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, et al. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588(7839):604–609, 2020.

[29] David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. In *International conference on machine learning*, pages 387–395. PMLR, 2014.

[30] Satinder P Singh, Michael L Littman, Nicholas K Jong, David Pardoe, and Peter Stone. Learning predictive state representations. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pages 712–719, 2003.

[31] Hao Sun, Zhizhong Li, Xiaotong Liu, Bolei Zhou, and Dahua Lin. Policy continuation with hindsight inverse dynamics. In *Advances in Neural Information Processing Systems*, pages 10265–10275, 2019.

[32] Richard S Sutton. Dyna, an integrated architecture for learning, planning, and reacting. *ACM Sigart Bulletin*, 2(4):160–163, 1991.

[33] Jane X Wang, Zeb Kurth-Nelson, Dhruva Tirumala, Hubert Soyer, Joel Z Leibo, Remi Munos, Charles Blundell, Dharshan Kumaran, and Matt Botvinick. Learning to reinforcement learn. *arXiv preprint arXiv:1611.05763*, 2016.

[34] Rui Zhao, Xudong Sun, and Volker Tresp. Maximum entropy-regularized multi-goal reinforcement learning. In *International Conference on Machine Learning*, pages 7553–7562, 2019.

[35] Menghui Zhu, Minghuan Liu, Jian Shen, Zhicheng Zhang, Sheng Chen, Weinan Zhang, Deheng Ye, Yong Yu, Qiang Fu, and Wei Yang. Mapgo: Model-assisted policy optimization for goal-oriented tasks. *arXiv preprint arXiv:2105.06350*, 2021.