Maximum Entropy-Regularized Multi-Goal Reinforcement Learning

Rui Zhao [1 2]   **Xudong Sun** [1]   **Volker Tresp** [1 2]

Abstract

In Multi-Goal Reinforcement Learning, an agent learns to achieve multiple goals with a goal-conditioned policy. During learning, the agent first collects the trajectories into a replay buffer, and later these trajectories are selected randomly for replay. However, the achieved goals in the replay buffer are often biased towards the behavior policies. From a Bayesian perspective, when there is no prior knowledge about the target goal distribution, the agent should learn uniformly from diverse achieved goals. Therefore, we first propose a novel multi-goal RL objective based on weighted entropy. This objective encourages the agent to maximize the expected return, as well as to achieve more diverse goals. Secondly, we developed a maximum entropy-based prioritization framework to optimize the proposed objective. For evaluation of this framework, we combine it with Deep Deterministic Policy Gradient, both with or without Hindsight Experience Replay. On a set of multi-goal robotic tasks of OpenAI Gym, we compare our method with other baselines and show promising improvements in both performance and sample-efficiency.

1. Introduction

Reinforcement Learning (RL) (Sutton & Barto, 1998) combined with Deep Learning (DL) (Goodfellow et al., 2016) has led to great successes in various tasks, such as playing video games (Mnih et al., 2015), challenging the World Go Champion (Silver et al., 2016), and learning autonomously to accomplish different robotic tasks (Ng et al., 2006; Peters & Schaal, 2008; Levine et al., 2016; Chebotar et al., 2017; Andrychowicz et al., 2017).

[1]Faculty of Mathematics, Informatics and Statistics, Ludwig Maximilian University of Munich, Munich, Bavaria, Germany [2]Siemens AG, Munich, Bavaria, Germany. Correspondence to: Rui Zhao <zhaorui.in.germany@gmail.com>.

One of the biggest challenges in RL is to make the agent learn efficiently in applications with sparse rewards. To tackle this challenge, Lillicrap et al. (2015) developed the Deep Deterministic Policy Gradient (DDPG), which enables the agent to learn continuous control, such as manipulation and locomotion. Schaul et al. (2015a) proposed Universal Value Function Approximators (UVFAs), which generalize not just over states, but also over goals, and extend value functions to multiple goals. Furthermore, to make the agent learn faster in sparse reward settings, Andrychowicz et al. (2017) introduced Hindsight Experience Replay (HER), which encourages the agent to learn from the goal-states it has achieved. The combined use of DDPG and HER allows the agent to learn to accomplish more complex robot manipulation tasks. However, there is still a huge gap between the learning efficiency of humans and RL agents. In most cases, an RL agent needs millions of samples before it is able to solve the tasks, while humans only need a few samples (Mnih et al., 2015).

In previous works, the concept of maximum entropy has been used to encourage exploration during training (Williams & Peng, 1991; Mnih et al., 2015; Wu & Tian, 2016). Recently, Haarnoja et al. (2017) introduced Soft-Q Learning, which learns a deep energy-based policy by evaluating the maximum entropy of actions for each state. Soft-Q Learning encourages the agent to learn all the policies that lead to the optimum (Levine, 2018). Furthermore, Soft Actor-Critic (Haarnoja et al., 2018c) demonstrated a better performance while showing compositional ability and robustness of the maximum entropy policy in locomotion (Haarnoja et al., 2018a) and robot manipulation tasks (Haarnoja et al., 2018b). The agent aims to maximize the expected reward while also maximizing the entropy to succeed at the task while acting as randomly as possible. Based on maximum entropy policies, Eysenbach et al. (2018) showed that the agent is able to develop diverse skills solely by maximizing an information theoretic objective without any reward function. For multi-goal and multi-task learning (Caruana, 1997), the diversity of training sets helps the agent transfer skills to unseen goals and tasks (Pan et al., 2010). The variability of training samples mitigates overfitting and helps the model to better generalize (Goodfellow
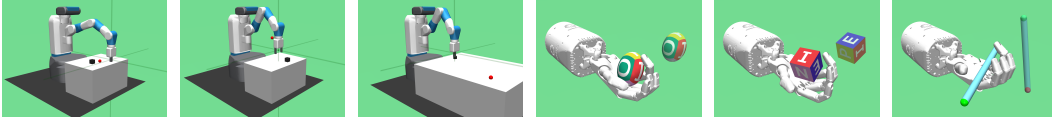
Figure 1. Robot arm Fetch and Shadow Dexterous hand environment: `FetchPush`, `FetchPickAndPlace`, `FetchSlide`, `HandManipulateEgg`, `HandManipulateBlock`, and `HandManipulatePen`.

et al., 2016). In our approach, we combine maximum entropy with multi-goal RL to help the agent to achieve unseen goals by learning uniformly from diverse achieved goals during training.

We observe that during experience replay the uniformly sampled trajectories are biased towards the behavior policies, with respect to the achieved goal-states. Consider training a robot arm to reach a certain point in a space. At the beginning, the agent samples trajectories using a random policy. The sampled trajectories are centered around the initial position of the robot arm. Therefore, the distribution of achieved goals, i.e., positions of the robot arm, is similar to a Gaussian distribution around the initial position, which is non-uniform. Sampling from such a distribution is biased towards the current policies. From a Bayesian point of view (Murphy, 2012), the agent should learn uniformly from these achieved goals, when there is no prior knowledge of the target goal distribution.

To correct this bias, we propose a new objective which combines maximum entropy and the multi-goal RL objective. This new objective uses entropy as a regularizer to encourage the agent to traverse diverse goal-states. Furthermore, we derive a safe lower bound for optimization. To optimize this surrogate objective, we implement maximum entropy-based prioritization as a simple yet effective solution.

2. Preliminary

2.1. Settings

Environments: We consider multi-goal reinforcement learning tasks, like the robotic simulation scenarios provided by OpenAI Gym (Plappert et al., 2018), where six challenging tasks are used for evaluation, including push, slide, pick & place with the robot arm, as well as hand manipulation of the block, egg, and pen, as shown in Figure 1. Accordingly, we define the following terminologies for this specific kind of multi-goal scenarios.

Goals: The goals g are the desired positions and the orientations of the object. Specifically, we use g^e, with e standing for environment, to denote the real goal which serves as the input from the environment, in order to distinguish it from the achieved goal used in Hindsight settings (Andrychowicz et al., 2017). Note that in this paper we consider the case where the goals can be represented by states, which leads

us to the concept of achieved goal-state g^s, with details explained below.

States, Goal-States and Achieved Goals: The state s consists of two sub-vectors, the achieved goal-state s^g, which represents the position and orientation of the object being manipulated, and the context state s^c, i.e. $s = (s^g \| s^c)$, where $\|$ denotes concatenation.

In our case, we define $g^s = s^g$ to represent an achieved goal that has the same dimension as the real goal g^e from the environment. The context state s^c contains the rest information about the state, including the linear and angular velocities of all robot joints and of the object. The real goals g^e can be substituted by the achieved goals g^s to facilitate learning. This goal relabeling technique was proposed by Andrychowicz et al. (2017) as Hindsight Experience Replay.

Achieved Goal Trajectory: A trajectory consisting solely of goal-states is represented as τ^g. We use τ^g to denote all the achieved goals in the trajectory τ, i.e., $\tau^g = (g_0^s, ..., g_T^s)$.

Rewards: We consider sparse rewards r. There is a tolerated range between the desired goal-states and the achieved goal-states. If the object is not in the tolerated range of the real goal, the agent receives a reward signal -1 for each transition; otherwise, the agent receives a reward signal 0.

Goal-Conditioned Policy: In multi-goal settings, the agent receives the environmental goal g^e and the state input $s = (s^g \| s^c)$. We want to train a goal-conditioned policy to effectively generalize its behavior to different environmental goals g^e.

2.2. Reinforcement Learning

We consider an agent interacting with an environment. We assume the environment is fully observable, including a set of state $\mathcal{S}$, a set of action $\mathcal{A}$, a distribution of initial states $p(s_0)$, transition probabilities $p(s_{t+1} \mid s_t, a_t)$, a reward function $r: \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, and a discount factor $\gamma \in [0, 1]$.

Deep Deterministic Policy Gradient: For continuous control tasks, the Deep Deterministic Policy Gradient (DDPG) shows promising performance, which is essentially an off-policy actor-critic method (Lillicrap et al., 2015).

Universal Value Function Approximators: For multi-

goal continuous control tasks, DDPG can be extended by Universal Value Function Approximators (UVFA) (Schaul et al., 2015a). UVFA essentially generalizes the Q-function to multiple goal-states, where the Q-value depends not only on the state-action pairs, but also on the goals.

Hindsight Experience Replay: For robotic tasks, if the goal is challenging and the reward is sparse, the agent could perform badly for a long time before learning anything. Hindsight Experience Replay (HER) encourages the agent to learn from whatever goal-states it has achieved. Andrychow-icz et al. (2017) show that HER makes training possible in challenging robotic tasks via goal relabeling, i.e., randomly substituting real goals with achieved goals.

2.3. Weighted Entropy

Guiaşu (1971) proposed weighted entropy, which is an extension of Shannon entropy. The definition of weighted entropy is given as

$$\mathcal{H}_p^w = -\sum_{k=1}^{K} w_k p_k \log p_k, \tag{1}$$

where w_k is the weight of the elementary event and p_k is the probability of the elementary event.

3. Method

In this section, we formally describe our method, including the mathematical derivation of the Maximum Entropy-Regularized Multi-Goal RL objective and the Maximum Entropy-based Prioritization framework.

3.1. Multi-Goal RL

In this paper, we consider multi-goal RL as goal-conditioned policy learning (Schaul et al., 2015a; Andrychowicz et al., 2017; Rauber et al., 2017; Plappert et al., 2018). We denote random variables by upper case letters and the values of random variables by corresponding lower case letters. For example, let $\text{Val}(X)$ denote the set of valid values to a random variable X, and let $p(x)$ denote the probability function of random variable X.

Consider that an agent receives a goal $g^e \in \text{Val}(G^e)$ at the beginning of the episode. The agent interacts with the environment for T timesteps. At each timestep t, the agent observes a state $s_t \in \text{Val}(S_t)$ and performs an action $a_t \in \text{Val}(A_t)$. The agent also receives a reward conditioned on the input goal $r(s_t, g^e) \in \mathbb{R}$.

We use $\boldsymbol{\tau} = s_1, a_1, s_2, a_2, \ldots, s_{T-1}, a_{T-1}, s_T$ to denote a trajectory, where $\boldsymbol{\tau} \in \text{Val}(\boldsymbol{\mathcal{T}})$. We assume that the probability $p(\boldsymbol{\tau} \mid g^e, \boldsymbol{\theta})$ of trajectory $\boldsymbol{\tau}$, given goal g^e and a policy

parameterized by $\boldsymbol{\theta} \in \text{Val}(\boldsymbol{\Theta})$, is given as

$$p(\boldsymbol{\tau} \mid g^e, \boldsymbol{\theta}) = p(s_1) \prod_{t=1}^{T-1} p(a_t \mid s_t, g^e, \boldsymbol{\theta}) p(s_{t+1} \mid s_t, a_t).$$

The transition probability $p(s_{t+1} \mid s_t, a_t)$ states that the probability of a state transition given an action is independent of the goal, and we denote it with $S_{t+1} \perp\!\!\!\perp G^e \mid S_t, A_t$. For every $\boldsymbol{\tau}, g^e$, and $\boldsymbol{\theta}$, we also assume that $p(\boldsymbol{\tau} \mid g^e, \boldsymbol{\theta})$ is non-zero. The expected return of a policy parameterized by $\boldsymbol{\theta}$ is given as

$$\begin{aligned}
\eta(\boldsymbol{\theta}) &= \mathbb{E}\left[\sum_{t=1}^{T} r(S_t, G^e) \mid \boldsymbol{\theta}\right] \\
&= \sum_{g^e} p(g^e) \sum_{\boldsymbol{\tau}} p(\boldsymbol{\tau} \mid g^e, \boldsymbol{\theta}) \sum_{t=1}^{T} r(s_t, g^e).
\end{aligned} \tag{2}$$

Off-policy RL methods use experience replay (Lin, 1992; Mnih et al., 2015) to leverage bias over variance and potentially improve sample-efficiency. In the off-policy case, the objective, Equation (2), is given as

$$\eta^{\mathcal{R}}(\boldsymbol{\theta}) = \sum_{\boldsymbol{\tau}, g^e} p_{\mathcal{R}}(\boldsymbol{\tau}, g^e \mid \boldsymbol{\theta}) \sum_{t=1}^{T} r(s_t, g^e), \tag{3}$$

where $\mathcal{R}$ denotes the replay buffer. Normally, the trajectories $\boldsymbol{\tau}$ are randomly sampled from the buffer. However, we observe that the trajectories in the replay buffer are often imbalanced with respect to the achieved goals $\boldsymbol{\tau}^g$. Thus, we propose Maximum Entropy-Regularized Multi-Goal RL to improve performance.

3.2. Maximum Entropy-Regularized Multi-Goal RL

In multi-goal RL, we want to encourage the agent to traverse diverse goal-state trajectories, and at the same time, maximize the expected return. This is like maximizing the empowerment (Mohamed & Rezende, 2015) of an agent attempting to achieve multiple goals. We propose the reward-weighted entropy objective for multi-goal RL, which is given as

$$\begin{aligned}
\eta^{\mathcal{H}}(\boldsymbol{\theta}) &= \mathcal{H}_p^w(\boldsymbol{\mathcal{T}}^g) \\
&= \mathbb{E}_p\left[\log \frac{1}{p(\boldsymbol{\tau}^g)} \sum_{t=1}^{T} r(S_t, G^e) \mid \boldsymbol{\theta}\right].
\end{aligned} \tag{4}$$

For simplicity, we use $p(\boldsymbol{\tau}^g)$ to represent $\sum_{g^e} p_{\mathcal{R}}(\boldsymbol{\tau}^g, g^e \mid \boldsymbol{\theta})$, which is the occurrence probability of the goal-state trajectory $\boldsymbol{\tau}^g$. The expectation is calculated based on $p(\boldsymbol{\tau}^g)$ as well, so the proposed objective is the weighted entropy (Guiaşu, 1971; Kelbert et al., 2017) of $\boldsymbol{\tau}^g$, which we denote as $\mathcal{H}_p^w(\boldsymbol{\mathcal{T}}^g)$, where the weight w is the accumulated reward $\sum_{t=1}^{T} r(s_t, g^e)$ in our case.

The objective function, Equation (4), has two interpretations. The first interpretation is to maximize the weighted expected return, where the rare trajectories have larger weights. Note that when all trajectories occur uniformly, this weighting mechanism has no effect. The second interpretation is to maximize a reward-weighted entropy, where the more rewarded trajectories have higher weights. This objective encourages the agent to learn how to achieve diverse goal-states, as well as to maximize the expected return.

In Equation (4), the weight, $\log\left(1/p(\boldsymbol{\tau}^g)\right)$, is unbounded, which makes the training of the universal function approximator unstable. Therefore, we propose a safe surrogate objective, $\eta^{\mathcal{L}}$, which is essentially a lower bound of the original objective.

3.3. Surrogate Objective

To construct the safe surrogate objective, we sample the trajectories from the replay buffer with a proposal distribution, $q(\boldsymbol{\tau}^g) = \frac{1}{Z} p(\boldsymbol{\tau}^g)\left(1 - p(\boldsymbol{\tau}^g)\right)$. $p(\boldsymbol{\tau}^g)$ represents the distribution of the goal trajectories in the replay buffer. The surrogate objective is given in Theorem 3, which is proved to be a lower bound of the original objective, Equation (4).

Theorem 1. *The surrogate $\eta^{\mathcal{L}}(\boldsymbol{\theta})$ is a lower bound of the objective function $\eta^{\mathcal{H}}(\boldsymbol{\theta})$, i.e., $\eta^{\mathcal{L}}(\boldsymbol{\theta}) < \eta^{\mathcal{H}}(\boldsymbol{\theta})$, where*

$$\eta^{\mathcal{H}}(\boldsymbol{\theta}) = \mathcal{H}_p^w(\boldsymbol{\mathcal{T}}^g)$$
$$= \mathbb{E}_p\left[\log\frac{1}{p(\boldsymbol{\tau}^g)}\sum_{t=1}^{T} r(S_t, G^e) \mid \boldsymbol{\theta}\right] \quad (5)$$

$$\eta^{\mathcal{L}}(\boldsymbol{\theta}) = Z \cdot \mathbb{E}_q\left[\sum_{t=1}^{T} r(S_t, G^e) \mid \boldsymbol{\theta}\right] \quad (6)$$

$$q(\boldsymbol{\tau}^g) = \frac{1}{Z} p(\boldsymbol{\tau}^g)\left(1 - p(\boldsymbol{\tau}^g)\right) \quad (7)$$

Z is the normalization factor for $q(\boldsymbol{\tau}^g)$. $\mathcal{H}_p^w(\boldsymbol{\mathcal{T}}^g)$ is the weighted entropy (Guiaşu, 1971; Kelbert et al., 2017), where the weight is the accumulated reward $\sum_{t=1}^{T} r(S_t, G^e)$, in our case.

Proof. See Appendix. $\qquad\square$

3.4. Prioritized Sampling

To optimize the surrogate objective, Equation (16), we cast the optimization process into a prioritized sampling framework. At each iteration, we first construct the proposal distribution $q(\boldsymbol{\tau}^g)$, which has a higher entropy than $p(\boldsymbol{\tau}^g)$. This ensures that the agent learns from a more diverse goal-state distribution. In Theorem 4, we prove that the entropy

with respect to $q(\boldsymbol{\tau}^g)$ is higher than the entropy with respect to $p(\boldsymbol{\tau}^g)$.

Theorem 2. *Let the probability density function of goals in the replay buffer be*

$$p(\boldsymbol{\tau}^g), where\ p(\boldsymbol{\tau}_i^g) \in (0,1)\ and\ \sum_{i=1}^{N} p(\boldsymbol{\tau}_i^g) = 1. \quad (8)$$

Let the proposal probability density function be defined as

$$q(\boldsymbol{\tau}_i^g) = \frac{1}{Z} p(\boldsymbol{\tau}_i^g)\left(1 - p(\boldsymbol{\tau}_i^g)\right),\ where\ \sum_{i=1}^{N} q(\boldsymbol{\tau}_i^g) = 1. \quad (9)$$

Then, the proposal goal distribution has an equal or higher entropy

$$\mathcal{H}_q(\boldsymbol{\mathcal{T}}^g) - \mathcal{H}_p(\boldsymbol{\mathcal{T}}^g) \geq 0. \quad (10)$$

Proof. See Appendix. $\qquad\square$

3.5. Estimation of Distribution

To optimize the surrogate objective with prioritized sampling, we need to know the probability distribution of a goal-state trajectory $p(\boldsymbol{\tau}^g)$. We use a Latent Variable Model (LVM) (Murphy, 2012) to model the underlying distribution of $p(\boldsymbol{\tau}^g)$, since LVM is suitable for modeling complex distributions.

Specifically, we use $p(\boldsymbol{\tau}^g \mid z_k)$ to denote the latent-variable-conditioned goal-state trajectory distribution, which we assume to be Gaussians. z_k is the k-th latent variable, where $k \in \{1, ..., K\}$ and K is the number of the latent variables. The resulting model is a Mixture of Gaussians(MoG), mathematically,

$$p(\boldsymbol{\tau}^g \mid \boldsymbol{\phi}) = \frac{1}{Z} \sum_{i=k}^{K} c_k \mathcal{N}(\boldsymbol{\tau}^g | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k), \quad (11)$$

where each Gaussian, $\mathcal{N}(\boldsymbol{\tau}^g | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$, has its own mean $\boldsymbol{\mu}_k$ and covariance $\boldsymbol{\Sigma}_k$, c_k represents the mixing coefficients, and Z is the partition function. The model parameter $\boldsymbol{\phi}$ includes all mean $\boldsymbol{\mu}_i$, covariance $\boldsymbol{\Sigma}_i$, and mixing coefficients c_k.

In prioritized sampling, we use the complementary predictive density of a goal-state trajectory $\boldsymbol{\tau}^g$ as the priority, which is given as

$$\bar{p}(\boldsymbol{\tau}^g \mid \boldsymbol{\phi}) \propto 1 - p(\boldsymbol{\tau}^g \mid \boldsymbol{\phi}). \quad (12)$$

The complementary density describes the likelihood that a goal-state trajectory $\boldsymbol{\tau}^g$ occurs in the replay buffer. A high complementary density corresponds to a rare occurrence of the goal trajectory. We want to over-sample these rare goal-state trajectories during replay to increase the entropy
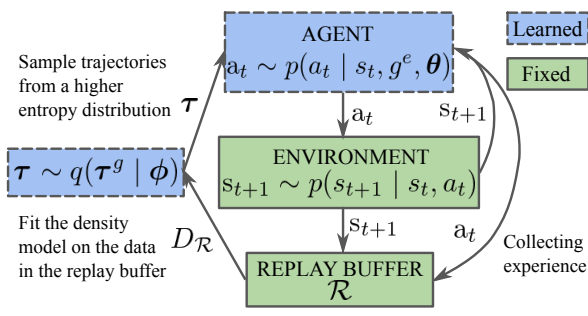
Algorithm 1 Maximum Entropy-based Prioritization (MEP)

---
while *not converged* **do**
    Sample goal $g^e \sim p(g^e)$ and initial state $s_0 \sim p(s_0)$
    for *steps_per_epoch* **do**
        for *steps_per_episode* **do**
            Sample action $a_t \sim p(a_t \mid s_t, g^e, \boldsymbol{\theta})$ from behavior policy.
            Step environment: $s_{t+1} \sim p(s_{t+1} \mid s_t, a_t)$.
            Update replay buffer $\mathcal{R}$.
            Construct prioritized sampling distribution:
            $q(\boldsymbol{\tau}^g) \propto (1 - p(\boldsymbol{\tau}^g \mid \boldsymbol{\phi}))p(\boldsymbol{\tau}^g)$ with higher $\mathcal{H}_q(\boldsymbol{\mathcal{T}}^g)$.
            Sample trajectories $\boldsymbol{\tau} \sim q(\boldsymbol{\tau}^g \mid \boldsymbol{\phi})$
            Update policy ($\boldsymbol{\theta}$) to max. $\mathbb{E}_q\left[r(S, G)\right]$ via DDPG, HER.
    Update density model ($\boldsymbol{\phi}$).

Figure 2. **MEP Algorithm**: We update the density model to construct a higher entropy distribution of achieved goals and update the agent with the more diversified training distribution.

of the training distribution. Therefore, we use the complementary density to construct the proposal distribution as a joint distribution

$$
\begin{aligned}
q(\boldsymbol{\tau}^g) &\propto \bar{p}(\boldsymbol{\tau}^g \mid \boldsymbol{\phi})p(\boldsymbol{\tau}^g) \\
&\propto (1 - p(\boldsymbol{\tau}^g \mid \boldsymbol{\phi}))p(\boldsymbol{\tau}^g) \\
&\approx p(\boldsymbol{\tau}^g) - p(\boldsymbol{\tau}^g)^2.
\end{aligned} \tag{13}
$$

3.6. Maximum Entropy-Based Prioritization

With prioritized sampling, the agent learns to maximize the return of a more diverse goal distribution. When the agent replays the samples, it first ranks all the trajectories with respect to their proposal distribution $q(\boldsymbol{\tau}^g)$, and then uses the ranking number directly as the probability for sampling. This means that rare goals have high ranking numbers and, equivalently, have higher priorities to be replayed. Here, we use the ranking instead of the density. The reason is that the rank-based variant is more robust since it is neither affected by outliers nor by density magnitudes. Furthermore, its heavy-tail property also guarantees that samples will be diverse (Schaul et al., 2015b). Mathematically, the probability of a trajectory to be replayed after the prioritization is:

$$
q(\boldsymbol{\tau}_i^g) = \frac{\text{rank}(q(\boldsymbol{\tau}_i^g))}{\sum_{n=1}^{N} \text{rank}(q(\boldsymbol{\tau}_n^g))}, \tag{14}
$$

where N is the total number of trajectories in the replay buffer and $\text{rank}(\cdot)$ is the ranking function.

We summarize the complete training algorithm in Algorithm 1 and in Figure 2. In short, we propose Maximum Entropy-Regularized Multi-Goal RL (Section 3.2) to enable RL agents to learn more efficiently in multi-goal tasks (Section 3.1). We integrate a goal entropy term into the normal expected return objective. To maximize the objective, Equation (4), we derive a surrogate objective in Theorem 3, i.e., a lower bound of the original objective. We use prioritized

sampling based on a higher entropy proposal distribution at each iteration and utilize off-policy RL methods to maximize the expected return. This framework is implemented as Maximum Entropy-based Prioritization (MEP).

4. Experiments

We test the proposed method on a variety of simulated robotic tasks, see Section 2.1, and compare it to strong baselines, including DDPG and HER. To the best of our knowledge, the most similar method to MEP is Prioritized Experience Replay (PER) (Schaul et al., 2015b). In the experiments, we first compare the performance improvement of MEP and PER. Afterwards, we compare the time-complexity of the two methods. We show that MEP improves performance with much less computational time than PER. Furthermore, the motivations of PER and MEP are different. The former uses TD-errors, while the latter is based on an entropy-regularized objective function.

In this section, we investigate the following questions:

1. Does incorporating goal entropy via MEP bring benefits to off-policy RL algorithms, such as DDPG or DDPG+HER?

2. Does MEP improve sample-efficiency of state-of-the-art RL approaches in robotic manipulation tasks?

3. How does MEP influence the entropy of the achieved goal distribution during training?

Our code is available online at https://github.com/ruizhaogit/mep.git. The implementation uses OpenAI Baselines (Dhariwal et al., 2017) with a backend of TensorFlow (Abadi et al., 2016).
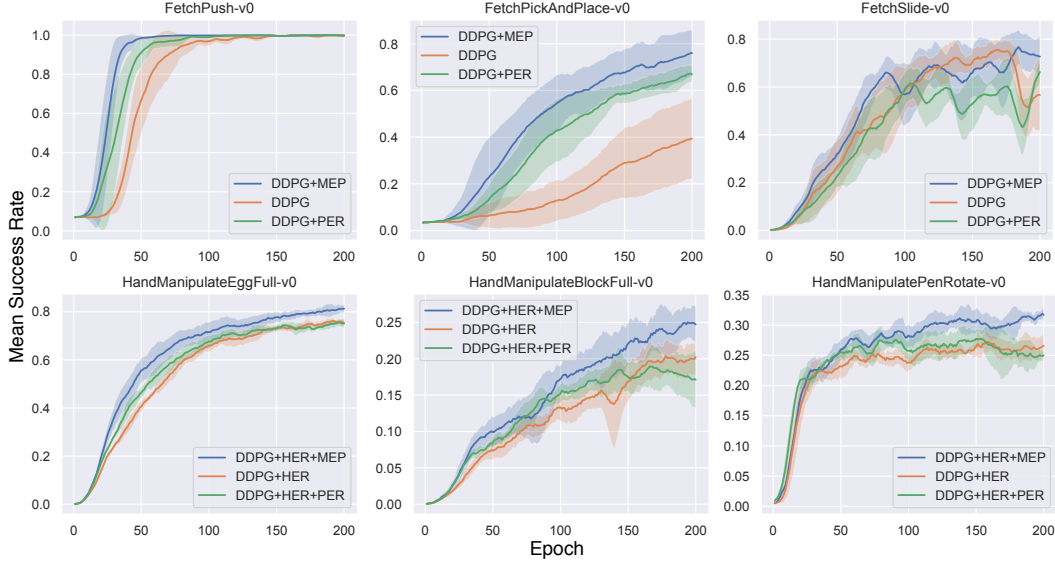
Figure 3. Mean success rate with standard deviation in all six robot environments

Table 1. Mean success rate (%) and training time (hour) for all six environments

Method	Push		Pick & Place		Slide	
	success	time	success	time	success	time
DDPG	99.90%	5.52h	39.34%	5.61h	75.67%	5.47h
DDPG+PER	99.94%	30.66h	67.19%	25.73h	66.33%	25.85h
DDPG+MEP	**99.96%**	6.76h	**76.02%**	6.92h	**76.77%**	6.66h

Method	Egg		Block		Pen	
	success	time	success	time	success	time
DDPG+HER	76.19%	7.33h	20.32%	8.47h	27.28%	7.55h
DDPG+HER+PER	75.46%	79.86h	18.95%	80.72h	27.74%	81.17h
DDPG+HER+MEP	**81.30%**	17.00h	**25.00%**	19.88h	**31.88%**	25.36h

4.1. Performance

To test the performance difference among methods including DDPG, DDPG+PER, and DDPG+MEP, we run the experiment in the three robot arm environments. We use the DDPG as the baseline here because the robot arm environment is relatively simple. In the more challenging robot hand environments, we use DDPG+HER as the baseline method and test the performance among DDPG+HER, DDPG+HER+PER, and DDPG+HER+MEP. To combine PER with HER, we calculate the TD-error of each transition based on the randomly selected achieved goals. Then we prioritize the transitions with higher TD-errors for replay.

Now, we compare the mean success rates. Each experiment is carried out with 5 random seeds and the shaded area represents the standard deviation. The learning curve with respect to training epochs is shown in Figure 3. For all experiments,

we use 19 CPUs and train the agent for 200 epochs. After training, we use the best-learned policy for evaluation and test it in the environment. The testing results are the mean success rates. A comparison of the performances along with the training time is shown in Table 1.

From Figure 3, we can see that MEP converges faster in all six tasks than both the baseline and PER. The agent trained with MEP also shows a better performance at the end of the training, as shown in Table 1. In Table 1, we can also see that the training time of MEP lies in between the baseline and PER. It is known that PER can become very time-consuming (Schaul et al., 2015b), especially when the memory size N is very large. The reason is that PER uses TD-errors for prioritization. After each update of the model, the agent needs to update the priorities of the transitions in the replay buffer, which is $O(\log N)$. In our experiments, we use the efficient implementation based on the "sum-tree" data
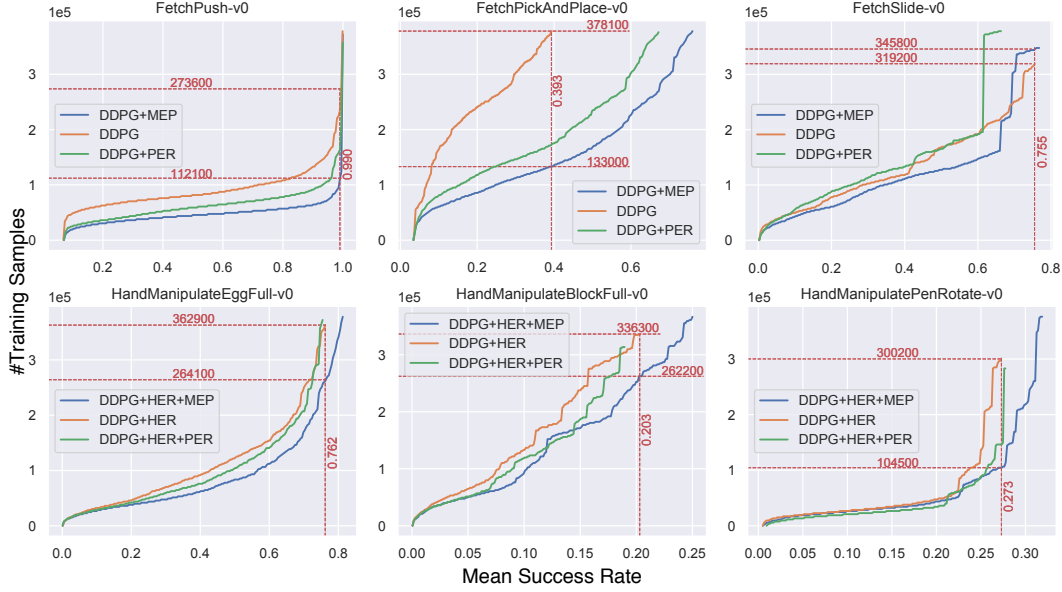
Figure 4. Number of training samples needed with respect to mean success rate for all six environments (the lower the better)

structure, which can be relatively efficiently updated and sampled from (Schaul et al., 2015b). To be more specific, MEP consumes much less computational time than PER. For example in the robot arm environments, on average, DDPG+MEP consumes about 1.2 times the training time of DDPG. In comparison, DDPG+PER consumes about 5 times the training time as DDPG. In this case, MEP is 4 times faster than PER. MEP is faster because it only updates the trajectory density once per epoch and can easily be combined with any multi-goal RL methods, such as DDPG and HER.

Table 1 shows that baseline methods with MEP result in better performance in all six tasks. The improvement increases by up to 39.34 percentage points compared to the baseline methods. The average improvement over the six tasks is 9.15 percentage points. We can see that MEP is a simple, yet effective method and it improves state-of-the-art methods.

4.2. Sample-Efficiency

To compare sample-efficiency of the baseline and MEP, we compare the number of training samples needed for a certain mean success rate. The comparison is shown in Figure 4. From Figure 4, in the `FetchPush-v0` environment, we can see that for the same 99% mean success rate, the baseline DDPG needs 273,600 samples for training, while DDPG+MEP only needs 112,100 samples. In this case, DDPG+MEP is more than twice (2.44) as sample-efficient as DDPG. Similarly, in the other five environments, MEP improves sample-efficiency by factors around one to three. In conclusion, for all six environments, MEP is able to

improve sample-efficiency by an average factor of two (1.95) over the baseline's sample-efficiency.

4.3. Goal Entropy

To verify that the overall MEP procedure works as expected, we calculated the entropy value of the achieved goal distribution $\mathcal{H}_p(\mathcal{T}^g)$ with respect to the epoch of training. The experimental results are averaged over 5 different random seeds. Figure 5 shows the mean entropy values with its standard deviation in three different environments. From Figure 5, we can see that the implemented MEP algorithm indeed increases the entropy of the goal distribution. This affirms the consistency of the stated theory with the implemented MEP framework.

5. Related Work

Maximum entropy was used in RL by Williams & Peng (1991) as an additional term in the loss function to encourage exploration and avoid local minimums (Mnih et al., 2016; Wu & Tian, 2016; Nachum et al., 2016; Asadi & Littman, 2016). A similar idea has also been utilized in the deep learning community, where entropy loss was used as a regularization technique to penalize over-confident output distributions (Pereyra et al., 2017). In RL, the entropy loss adds more cost to actions that dominate quickly. A higher entropy loss favors more exploration (Mnih et al., 2016). Neu et al. (2017) gave a unified view on entropy-regularized Markov Decision Processes (MDP) and discussed the convergence properties of entropy-regularized RL, including TRPO (Schulman et al., 2015) and A3C (Mnih et al., 2016).
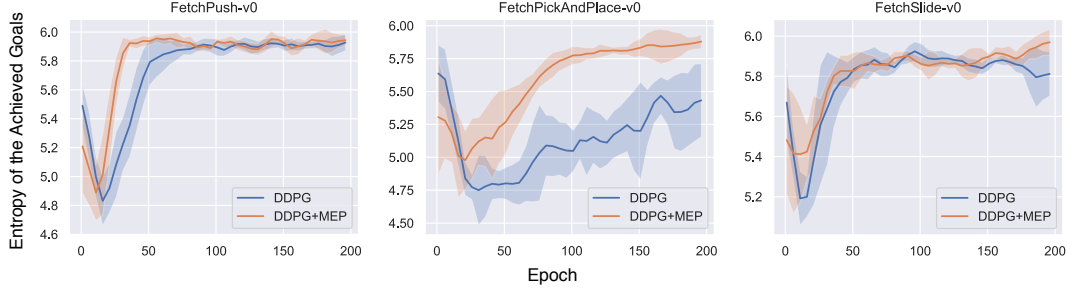
Figure 5. Entropy values of the achieved goal distribution $\mathcal{H}_p(\mathcal{T}^g)$ during training

More recently, Haarnoja et al. (2017) and Levine (2018) proposed deep energy-based policies with state conditioned entropy-based regularization, which is known as Soft-Q Learning. They showed that maximum entropy policies emerge as the solution when optimal control is cast as probabilistic inference. Concurrently, Schulman et al. (2017) showed the connection and the equivalence between Soft-Q Learning and policy gradients. Maximum entropy policies are shown to be robust and lead to better initializations for RL agents (Haarnoja et al., 2018a;b). Based on maximum entropy polices, Eysenbach et al. (2018) developed an information theoretic objective, which enables the agent to automatically discover different sets of skills.

Unlike aforementioned works (Williams & Peng, 1991; Mnih et al., 2016; Haarnoja et al., 2017), the information theoretic objective (Eysenbach et al., 2018) uses state, not actions, to calculate the entropy for distinguishing different skills. Our work is similar to this previous work (Eysenbach et al., 2018) in the sense that we also use the states, instead of actions, to calculate the entropy term and encourage the trained agent to cover a variety of goal-states. Our method generalizes to multi-goal and multi-task RL (Kaelbling, 1993; Sutton et al., 1999; Bakker & Schmidhuber, 2004; Sutton et al., 2011; Szepesvari et al., 2014; Schaul et al., 2015a; Pinto & Gupta, 2017; Plappert et al., 2018).

The entropy term that we used in the multi-goal RL objective is maximized over goal-states. We use maximum goal entropy as a regularization for multi-goal RL, which encourages the agent to learn uniformly with respect to goals instead of experienced transitions. This corrects the bias introduced by the agent's behavior policies. For example, the more easily achievable goals are generally dominant in the replay buffer. The goal entropy-regularized objective allows the agent to learn to achieve the unknown real goals, as well as various virtual goals.

We implemented the maximum entropy regularization via prioritized sampling based on achieved goal-states. We believe that the most similar framework is prioritized experience replay (Schaul et al., 2015b). Prioritized experience replay was introduced by Schaul et al. (2015b) as an improvement to the experience replay in DQN (Mnih et al., 2015). It prioritizes the transitions with higher TD-error in the replay buffer to speed up training. The prioritized experience replay is motivated by TD-errors. However, the motivation of our method comes from information theory—maximum entropy. Compared to prioritized experience replay, our method performs superior empirically and consumes much less computational time.

The intuition behind our method is to assign priority to those under-represented goals, which are relatively more valuable to learn from (see Appendix). Essentially, our method samples goals from an entropy-regularized distribution, rather than from a true replay buffer distribution, which is biased towards the behavior policies. Similar to recent work on goal sampling methods (Forestier et al., 2017; Péré et al., 2018; Florensa et al., 2018; Zhao & Tresp, 2018; Nair et al., 2018; Warde-Farley et al., 2018), our aim is to model a goal-conditioned MDP. In the future, we want to further explore the role of goal entropy in multi-goal RL.

6. Conclusion

This paper makes three contributions. First, we propose the idea of Maximum Entropy-Regularized Multi-Goal RL, which is essentially a reward-weighted entropy objective. Secondly, we derive a safe surrogate objective, i.e., a lower bound of the original objective, to achieve stable optimization. Thirdly, we implement a novel Maximum Entropy-based Prioritization framework for optimizing the surrogate objective. Overall, our approach encourages the agent to achieve a diverse set of goals while maximizing the expected return.

We evaluated our approach in multi-goal robotic simulations. The experimental results showed that our approach improves performance and sample-efficiency of the agent while keeping computational time under control. More precisely, the results showed that our method improves performance by 9 percentage points and sample-efficiency by a factor of two compared to state-of-the-art methods.

References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al. Tensorflow: a system for large-scale machine learning. In *OSDI*, volume 16, pp. 265–283, 2016.

Andrychowicz, M., Wolski, F., Ray, A., Schneider, J., Fong, R., Welinder, P., McGrew, B., Tobin, J., Abbeel, O. P., and Zaremba, W. Hindsight experience replay. In *Advances in Neural Information Processing Systems*, pp. 50485058, 2017.

Asadi, K. and Littman, M. L. An alternative softmax operator for reinforcement learning. *arXiv preprint arXiv:1612.05628*, 2016.

Bakker, B. and Schmidhuber, J. Hierarchical reinforcement learning based on subgoal discovery and subpolicy specialization. In *Proc. of the 8-th Conf. on Intelligent Autonomous Systems*, pp. 438–445, 2004.

Benesty, J., Chen, J., Huang, Y., and Cohen, I. Pearson correlation coefficient. In *Noise reduction in speech processing*, pp. 1–4. Springer, 2009.

Caruana, R. Multitask learning. *Machine learning*, 28(1): 41–75, 1997.

Chebotar, Y., Kalakrishnan, M., Yahya, A., Li, A., Schaal, S., and Levine, S. Path integral guided policy search. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pp. 3381–3388. IEEE, 2017.

Dhariwal, P., Hesse, C., Klimov, O., Nichol, A., Plappert, M., Radford, A., Schulman, J., Sidor, S., Wu, Y., and Zhokhov, P. Openai baselines. https://github.com/openai/baselines, 2017.

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

Florensa, C., Held, D., Geng, X., and Abbeel, P. Automatic goal generation for reinforcement learning agents. In *International Conference on Machine Learning*, pp. 1514–1523, 2018.

Forestier, S., Mollard, Y., and Oudeyer, P.-Y. Intrinsically motivated goal exploration processes with automatic curriculum learning. *arXiv preprint arXiv:1708.02190*, 2017.

Goodfellow, I., Bengio, Y., Courville, A., and Bengio, Y. *Deep learning*, volume 1. MIT press Cambridge, 2016.

Guiaşu, S. Weighted entropy. *Reports on Mathematical Physics*, 2(3):165–179, 1971.

Haarnoja, T., Tang, H., Abbeel, P., and Levine, S. Reinforcement learning with deep energy-based policies. *arXiv preprint arXiv:1702.08165*, 2017.

Haarnoja, T., Hartikainen, K., Abbeel, P., and Levine, S. Latent space policies for hierarchical reinforcement learning. *arXiv preprint arXiv:1804.02808*, 2018a.

Haarnoja, T., Pong, V., Zhou, A., Dalal, M., Abbeel, P., and Levine, S. Composable deep reinforcement learning for robotic manipulation. *arXiv preprint arXiv:1803.06773*, 2018b.

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *arXiv preprint arXiv:1801.01290*, 2018c.

Kadelburg, Z., Dukic, D., Lukic, M., and Matic, I. Inequalities of karamata, schur and muirhead, and some applications. *The Teaching of Mathematics*, 8(1):31–45, 2005.

Kaelbling, L. P. Hierarchical learning in stochastic domains: Preliminary results. In *Proceedings of the tenth international conference on machine learning*, volume 951, pp. 167–173, 1993.

Kelbert, M., Stuhl, I., and Suhov, Y. Weighted entropy: basic inequalities. *Modern Stochastics: Theory and Applications*, 4(3):233–252, 2017. doi: 10.15559/17-VMSTA85. URL www.i-journals.org/vmsta.

Levine, S. Reinforcement learning and control as probabilistic inference: Tutorial and review. *arXiv preprint arXiv:1805.00909*, 2018.

Levine, S., Finn, C., Darrell, T., and Abbeel, P. End-to-end training of deep visuomotor policies. *The Journal of Machine Learning Research*, 17(1):13341373, 2016.

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

Lin, L.-J. Self-improving reactive agents based on reinforcement learning, planning and teaching. *Machine learning*, 8(3-4):293321, 1992.

Marshall, A. W., Olkin, I., and Arnold, B. C. *Inequalities: theory of majorization and its applications*, volume 143. Springer, 1979.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540): 529, 2015.

Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., and Kavukcuoglu, K. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pp. 1928–1937, 2016.

Mohamed, S. and Rezende, D. J. Variational information maximisation for intrinsically motivated reinforcement learning. In *Advances in neural information processing systems*, pp. 2125–2133, 2015.

Murphy, K. P. Machine learning: A probabilistic perspective. adaptive computation and machine learning, 2012.

Nachum, O., Norouzi, M., and Schuurmans, D. Improving policy gradient by exploring under-appreciated rewards. *arXiv preprint arXiv:1611.09321*, 2016.

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

Neu, G., Jonsson, A., and Gómez, V. A unified view of entropy-regularized markov decision processes. *arXiv preprint arXiv:1705.07798*, 2017.

Ng, A. Y., Coates, A., Diel, M., Ganapathi, V., Schulte, J., Tse, B., Berger, E., and Liang, E. Autonomous inverted helicopter flight via reinforcement learning. In *Experimental Robotics IX*, pp. 363372. Springer, 2006.

Pan, S. J., Yang, Q., et al. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2010.

Péré, A., Forestier, S., Sigaud, O., and Oudeyer, P.-Y. Unsupervised learning of goal spaces for intrinsically motivated goal exploration. *arXiv preprint arXiv:1803.00781*, 2018.

Pereyra, G., Tucker, G., Chorowski, J., Kaiser, Ł., and Hinton, G. Regularizing neural networks by penalizing confident output distributions. *arXiv preprint arXiv:1701.06548*, 2017.

Peters, J. and Schaal, S. Reinforcement learning of motor skills with policy gradients. *Neural networks*, 21(4): 682697, 2008.

Petrov, F. Shannon entropy of $p(x)(1 - p(x))$ is no less than entropy of $p(x)$. MathOverflow. URL https://mathoverflow.net/q/320726. URL:https://mathoverflow.net/q/320726 (version: 2019-01-12).

Pinto, L. and Gupta, A. Learning to push by grasping: Using multiple tasks for effective learning. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pp. 2161–2168. IEEE, 2017.

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

Rauber, P., Mutz, F., and Schmidhuber, J. Hindsight policy gradients. *arXiv preprint arXiv:1711.06006*, 2017.

Schaul, T., Horgan, D., Gregor, K., and Silver, D. Universal value function approximators. In *International Conference on Machine Learning*, pp. 1312–1320, 2015a.

Schaul, T., Quan, J., Antonoglou, I., and Silver, D. Prioritized experience replay. *arXiv preprint arXiv:1511.05952*, 2015b.

Schulman, J., Levine, S., Abbeel, P., Jordan, M., and Moritz, P. Trust region policy optimization. In *International Conference on Machine Learning*, pp. 1889–1897, 2015.

Schulman, J., Chen, X., and Abbeel, P. Equivalence between policy gradients and soft q-learning. *arXiv preprint arXiv:1704.06440*, 2017.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484489, 2016.

Sutton, R. S. and Barto, A. G. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

Sutton, R. S., Precup, D., and Singh, S. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, 112(1-2): 181–211, 1999.

Sutton, R. S., Modayil, J., Delp, M., Degris, T., Pilarski, P. M., White, A., and Precup, D. Horde: A scalable real-time architecture for learning knowledge from unsupervised sensorimotor interaction. In *The 10th International Conference on Autonomous Agents and Multiagent Systems-Volume 2*, pp. 761–768. International Foundation for Autonomous Agents and Multiagent Systems, 2011.

Szepesvari, C., Sutton, R. S., Modayil, J., Bhatnagar, S., et al. Universal option models. In *Advances in Neural Information Processing Systems*, pp. 990–998, 2014.

Warde-Farley, D., Van de Wiele, T., Kulkarni, T., Ionescu, C., Hansen, S., and Mnih, V. Unsupervised control through non-parametric discriminative rewards. *arXiv preprint arXiv:1811.11359*, 2018.

Williams, R. J. and Peng, J. Function optimization using connectionist reinforcement learning algorithms. *Connection Science*, 3(3):241–268, 1991.

Wu, Y. and Tian, Y. Training agent for first-person shooter game with actor-critic curriculum learning. 2016.

Zhao, R. and Tresp, V. Energy-based hindsight experience prioritization. In *Proceedings of the 2nd Conference on Robot Learning*, pp. 113–122, 2018.

Zhao, R. and Tresp, V. Curiosity-driven experience prioritization via density estimation. *arXiv preprint arXiv:1902.08039*, 2019.

A. Proof of Theorem 1

Theorem 3. *The surrogate $\eta^{\mathcal{L}}(\boldsymbol{\theta})$ is a lower bound of the objective function $\eta^{\mathcal{H}}(\boldsymbol{\theta})$, i.e., $\eta^{\mathcal{L}}(\boldsymbol{\theta}) < \eta^{\mathcal{H}}(\boldsymbol{\theta})$, where*

$$\eta^{\mathcal{H}}(\boldsymbol{\theta}) = \mathcal{H}_p^w(\boldsymbol{\mathcal{T}}^g) = \mathbb{E}_p \left[\log \frac{1}{p(\boldsymbol{\tau}^g)} \sum_{t=1}^{T} r(S_t, G^e) \mid \boldsymbol{\theta} \right] \tag{15}$$

$$\eta^{\mathcal{L}}(\boldsymbol{\theta}) = Z \cdot \mathbb{E}_q \left[\sum_{t=1}^{T} r(S_t, G^e) \mid \boldsymbol{\theta} \right] \tag{16}$$

$$q(\boldsymbol{\tau}^g) = \frac{1}{Z} p(\boldsymbol{\tau}^g) \left(1 - p(\boldsymbol{\tau}^g) \right) \tag{17}$$

Z is the normalization factor for $q(\boldsymbol{\tau}^g)$. $\mathcal{H}_p^w(\boldsymbol{\mathcal{T}}^g)$ is the weighted entropy (Guiaşu, 1971; Kelbert et al., 2017), where the weight is the accumulated reward $\sum_{t=1}^{T} r(S_t, G^e)$ in our case.

Proof.

$$\eta^{\mathcal{L}}(\boldsymbol{\theta}) = Z \cdot \mathbb{E}_q \left[\sum_{t=1}^{T} r(S_t, G^e) \mid \boldsymbol{\theta} \right] \tag{18}$$

$$= \sum_{\boldsymbol{\tau}^g} Z \cdot q(\boldsymbol{\tau}^g) \sum_{t=1}^{T} r(s_t, g^e) \tag{19}$$

$$= \sum_{\boldsymbol{\tau}^g} \frac{Z}{Z} p(\boldsymbol{\tau}^g)(1 - p(\boldsymbol{\tau}^g)) \sum_{t=1}^{T} r(s_t, g^e) \tag{20}$$

$$< \sum_{\boldsymbol{\tau}^g} -p(\boldsymbol{\tau}^g) \log p(\boldsymbol{\tau}^g) \sum_{t=1}^{T} r(s_t, g^e) \tag{21}$$

$$= \mathbb{E}_p \left[\log \frac{1}{p(\boldsymbol{\tau}^g)} \sum_{t=1}^{T} r(S_t, G^e) \mid \boldsymbol{\theta} \right] \tag{22}$$

$$= \mathcal{H}_p^w(\boldsymbol{\mathcal{T}}^g) \tag{23}$$

$$= \eta^{\mathcal{H}}(\boldsymbol{\theta}) \tag{24}$$

In the inequality, we use the property $\log x < x - 1$. $\qquad\square$

B. Proof of Theorem 2

Theorem 4. *Let the probability density function of goals in the replay buffer be*

$$p(\boldsymbol{\tau}^g), \text{where } p(\boldsymbol{\tau}_i^g) \in (0, 1) \text{ and } \sum_{i=1}^{N} p(\boldsymbol{\tau}_i^g) = 1. \tag{25}$$

Let the proposal probability density function be defined as

$$q(\boldsymbol{\tau}_i^g) = \frac{1}{Z} p(\boldsymbol{\tau}_i^g) \left(1 - p(\boldsymbol{\tau}_i^g) \right), \text{ where } \sum_{i=1}^{N} q(\boldsymbol{\tau}_i^g) = 1. \tag{26}$$

Then, the proposal goal distribution has an equal or higher entropy

$$\mathcal{H}_q(\boldsymbol{\mathcal{T}}^g) - \mathcal{H}_p(\boldsymbol{\mathcal{T}}^g) \geq 0. \tag{27}$$

Proof. For clarity, we define the notations in this proof as $p_i = p(\boldsymbol{\tau}_i^g)$ and $q_i = q(\boldsymbol{\tau}_i^g)$.

Note that the definition of Entropy is

$$\mathcal{H}_p = \sum_i -p_i \log(p_i), \tag{28}$$

where the ith summand is $p_i \log(p_i)$, which is a concave function. Since the goal distribution has a finite support I, we have the real-valued vector $(p_1, \ldots, p_N)$ and $(\frac{1}{Z}q_1, \ldots, \frac{1}{Z}q_N)$.

We use Karamata's inequality (Kadelburg et al., 2005), which states that if the vector $(p_1, \ldots, p_N)$ majorizes $(\frac{1}{Z}q_1, \ldots, \frac{1}{Z}q_N)$ then the summation of the concave transformation of the first vector is smaller than the concave transformation of the second vector.

In our case, the concave transformation is the weighted information at the ith position $-p_i \log(p_i)$, where the weight is the probability p_i (entropy is the expectation of information). Therefore, the proof of the theorem is also a proof of the majorizing property of p over q (Petrov).

We denote the proposal goal distribution as

$$q_i = f(p_i) = \frac{1}{Z}p_i(1 - p_i). \tag{29}$$

Note that in our case, the partition function Z is a constant.

Majorizing has three requirements (Marshall et al., 1979).

The first requirement is that both vectors must sum up to one. This requirement is already met because

$$\sum_i p_i = \sum_i q_i = 1. \tag{30}$$

The second requirement is that monotonicity exits. Without loss of generality, we assume the probabilities are sorted:

$$p_1 \geq p_2 \geq \ldots \geq p_N \tag{31}$$

Thus, if $i > j$ then

$$f(p_i) - f(p_j) = \frac{1}{Z}p_i(1 - p_i) - \frac{1}{Z}p_j(1 - p_j) \tag{32}$$

$$= \frac{1}{Z}[(p_i - p_j) - (p_i + p_j)(p_i - p_j)] \tag{33}$$

$$= \frac{1}{Z}(p_i - p_j)(1 - p_i - p_j) \tag{34}$$

$$\geq 0. \tag{35}$$

which means that if the original goal probabilities are sorted, the transformed goal probabilities are also sorted,

$$f(p_1) \geq f(p_2) \geq \ldots \geq f(p_N). \tag{36}$$

The third requirement is that for an arbitrary cutoff index k, there is

$$p_1 + \ldots p_k < q_1 + \ldots + q_k. \tag{37}$$

To prove this, we have

$$p_1 + \ldots + p_k = \frac{p_1 + \ldots + p_k}{1} \tag{38}$$

$$= \frac{p_1 + \ldots + p_k}{p_1 + \ldots + p_N} \tag{39}$$

$$\geq f(p_1) + \ldots + f(p_k) \tag{40}$$

$$= \frac{1}{Z}[p_1(1 - p_1) + \ldots + p_k(1 - p_k)] \tag{41}$$

$$= \frac{1}{Z}[p_1 + \ldots + p_k - (p_1^2 + \ldots + p_k^2)] \tag{42}$$

Note that, we multiply $Z * 1$ to each side of

$$Z = p_1(1 - p_1) + \ldots + p_N(1 - p_N). \tag{43}$$

Then we have

$$(p_1 + \ldots + p_k)Z * 1 \geq p_1 + \ldots + p_k - (p_1^2 + \ldots + p_k^2) * 1. \tag{44}$$

Now, we substitute the expression of Z and then have

$$(p_1 + \ldots + p_k)[p_1(1 - p_1) + \ldots + p_N(1 - p_N)] \geq [p_1 + \ldots + p_k - (p_1^2 + \ldots + p_k^2)] * 1. \tag{45}$$

We express 1 as a series of terms $\sum_i p_i$, we have

$$(p_1 + \ldots + p_k)[p_1(1 - p_1) + \ldots + p_N(1 - p_N)] \geq [p_1 + \ldots + p_k - (p_1^2 + \ldots + p_k^2)] * [(p_1 + \ldots + p_k) + (p_{k+1} + \ldots p_N)]. \tag{46}$$

We use the distributive law to the right side and have

$$\begin{aligned} &(p_1 + \ldots + p_k)[p_1(1 - p_1) + \ldots + p_N(1 - p_N)] \\ &\geq [p_1 + \ldots + p_k] * [(p_1 + \ldots + p_k) + (p_{k+1} + \ldots p_N)] - [(p_1^2 + \ldots + p_k^2)] * [(p_1 + \ldots + p_k) + (p_{k+1} + \ldots p_N)]. \end{aligned} \tag{47}$$

We move the first term on the right side to the left and use the distributive law then have

$$(p_1 + \ldots + p_k)[-1 * (p_1^2 + \ldots + p_N^2))] \geq -[(p_1^2 + \ldots + p_k^2)] * [(p_1 + \ldots + p_k) + (p_{k+1} + \ldots p_N)]. \tag{48}$$

We use the distributive law again on the right side and move the first term to the left and use the distributive law then have

$$(p_1 + \ldots + p_k)[-1 * (p_{k+1}^2 + \ldots + p_N^2))] \geq -[(p_1^2 + \ldots + p_k^2)] * [(p_{k+1} + \ldots p_N)]. \tag{49}$$

We remove the minus sign then have

$$(p_1 + \ldots + p_k)[(p_{k+1}^2 + \ldots + p_N^2))] \leq [(p_1^2 + \ldots + p_k^2)] * [(p_{k+1} + \ldots p_N)]. \tag{50}$$

To prove the inequality above, it suffices to show that the inequality holds true for each associated term of the multiplication on each side of the inequality.

Suppose that

$$i \leq k < j \tag{51}$$

then we have

$$p_i > p_j. \tag{52}$$

As mentioned above, the probabilities are sorted in descending order. We have

$$p_i p_j^2 - p_i^2 p_j = p_i p_j (p_j - p_i) < 0 \tag{53}$$

then

$$p_i p_j^2 < p_i^2 p_j. \tag{54}$$

Therefore, we have proved that the inequality holds true for an arbitrary associated term, which also applies when they are added up. $\square$
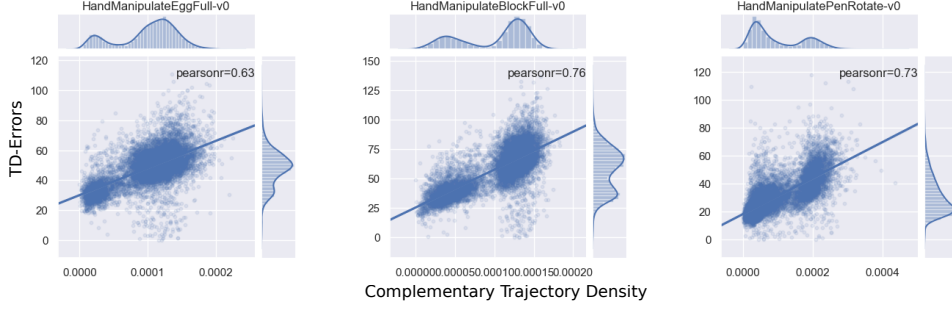
Figure 6. Pearson correlation between the complementary density $\bar{p}(\tau^g)$ and TD-errors in the middle of training

C. Insights

To further understand why maximum entropy in goal space facilitates learning, we look into the TD-errors during training. We investigate the correlation between the complementary predictive density $\bar{p}(\tau^g \mid \phi)$ and the TD-errors of the trajectory. The Pearson correlation coefficients, i.e., Pearson's r (Benesty et al., 2009), between the density $\bar{p}(\tau^g \mid \phi)$ and the TD-errors of the trajectory are 0.63, 0.76, and 0.73, for the hand manipulation of egg, block, and pen tasks, respectively. The plot of the Pearson correlation is shown in Figure 6. The value of Pearson's r is between 1 and -1, where 1 is total positive linear correlation, 0 is no linear correlation, and -1 is total negative linear correlation. We can see that the complementary predictive density is correlated with the TD-errors of the trajectory with an average Pearson's r of 0.7. This proves that the agent learns faster from a more diverse goal distribution. Under-represented goals often have higher TD-errors, and thus are relatively more valuable to learn from. Therefore, it is helpful to maximize the goal entropy and prioritize the under-represented goals during training.