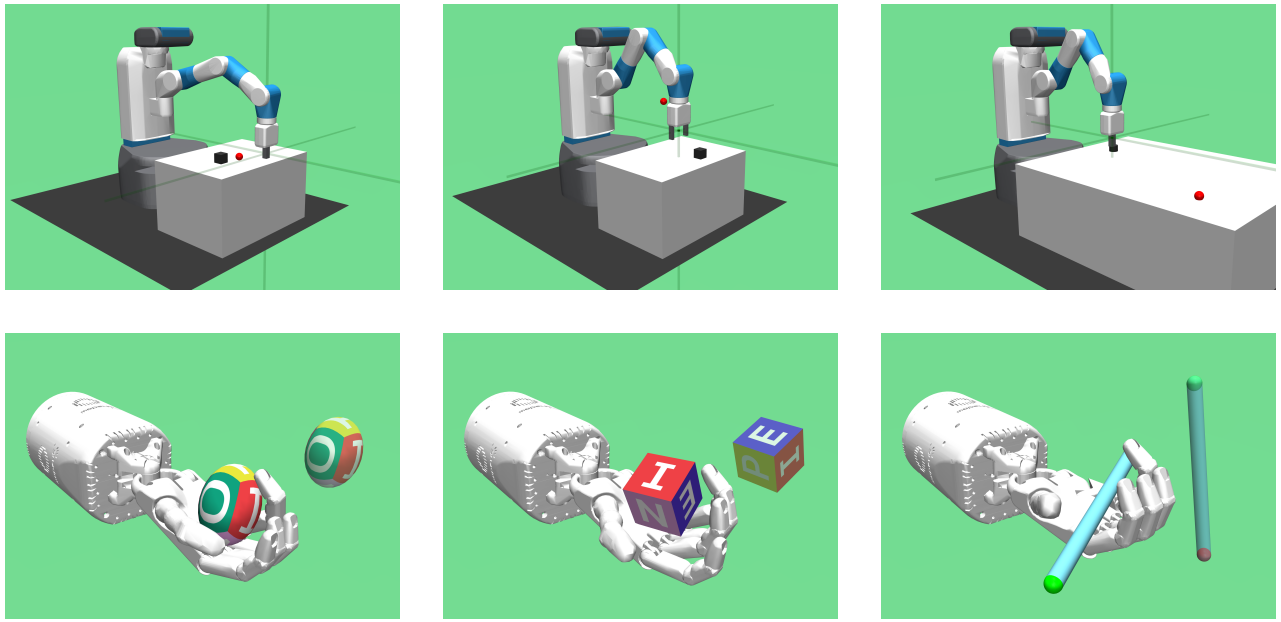
SIEMENS
Ingenuity for life

Maximum Entropy-Regularized Multi-Goal Reinforcement Learning

Rui Zhao*, Xudong Sun, Volker Tresp
Siemens AG & Ludwig Maximilian University of Munich | June 2019 | ICML 2019

Introduction

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.



OpenAI Gym Robotic Simulations

Motivation

- We observed that the achieved goals in the replay buffer are often biased towards the behavior policies.

- From a Bayesian perspective (Murphy, 2012), when there is no prior knowledge of the target goal distribution, the agent should learn uniformly from diverse achieved goals.

- We want to encourage the agent to achieve a diverse set of goals while maximizing the expected return.

Contributions

- First, we propose a novel multi-goal RL objective based on weighted entropy, which is essentially a reward-weighted entropy objective.

- Secondly, we derive a safe surrogate objective, that is, a lower bound of the original objective, to achieve stable optimization.

- Thirdly, we developed a Maximum Entropy-based Prioritization (MEP) framework to optimize the derived surrogate objective.

- We evaluate the proposed method in the OpenAI Gym robotic simulations.

A Novel Multi-Goal RL Objective Based on Weighted Entropy

Guiacsu [1971] proposed weighted entropy, which is an extension of Shannon entropy. The definition of weighted entropy is given by

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

where w is the weight of the event and p is the probability of the event.

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

This objective encourages the agent to maximize the expected return as well as to achieve more diverse goals.

* We use $\boldsymbol{\tau}^g$ to denote all the achieved goals in the trajectory $\boldsymbol{\tau}$, i.e., $\boldsymbol{\tau}^g = (g_0^s, ..., g_T^s)$.

A Safe Surrogate Objective

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

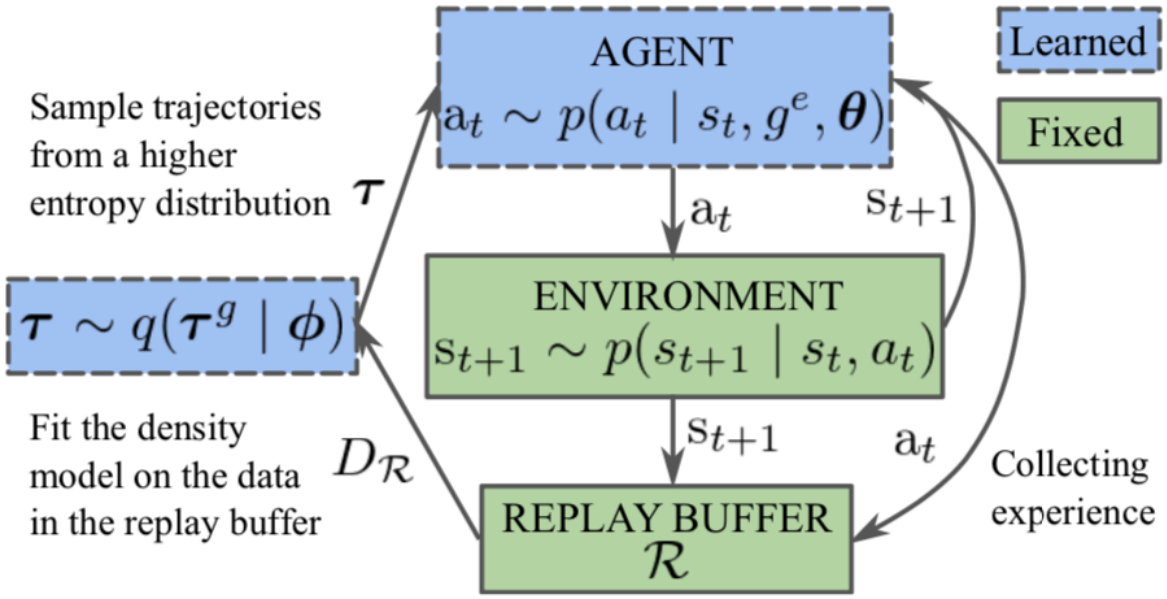
$$\eta^{\mathcal{H}}(\boldsymbol{\theta}) = \mathcal{H}_p^w(\boldsymbol{\mathcal{T}}^g) = \mathbb{E}_p \left[\log \frac{1}{p(\boldsymbol{\tau}^g)} \sum_{t=1}^{T} r(S_t, G^e) \mid \boldsymbol{\theta} \right]$$

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

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

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

Maximum Entropy-based Prioritization (MEP)



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 ($\boldsymbol{\phi}$).

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.

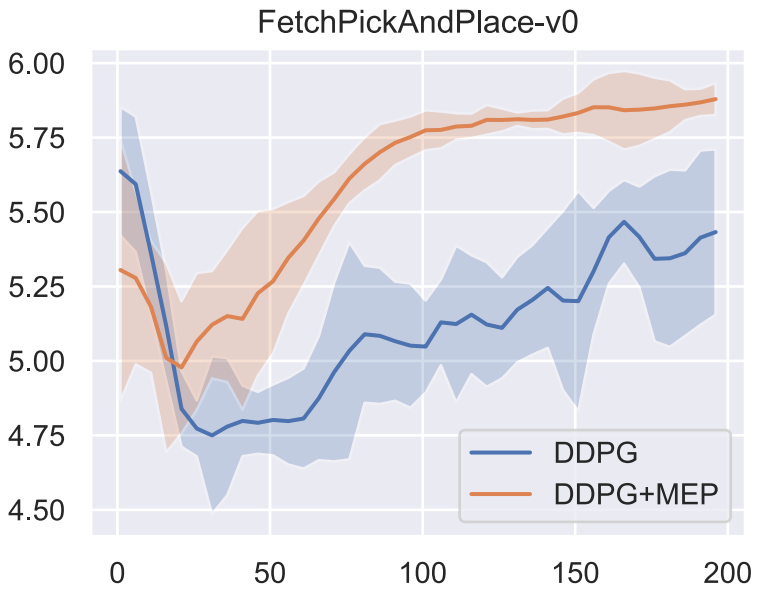
Mean success rate and training time

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

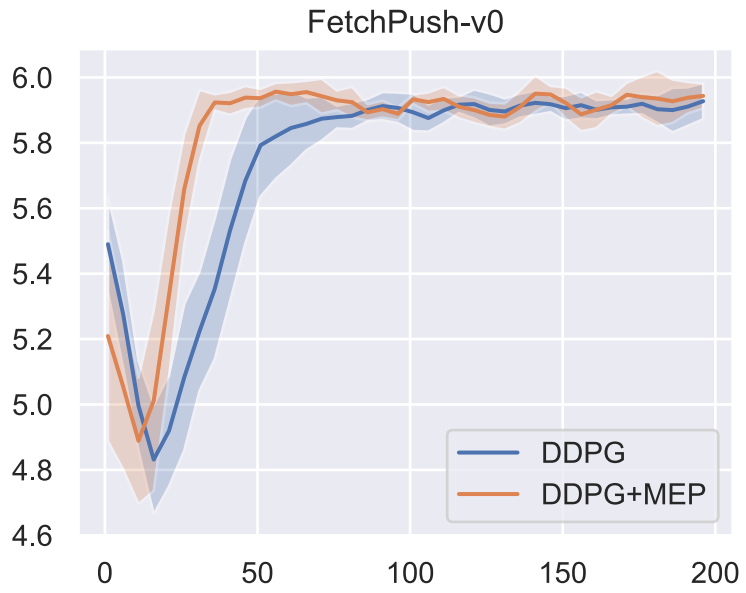
	Push		Pick & Place		Slide	
Method	success	time	success	time	success	time
DDPG	99.90%	5.52h	39.34%	5.61h	75.67%	5.47h
DDPG+PER	99.94%	30.66h	67.19%	25.73h	66.33%	25.85h
DDPG+MEP	**99.96%**	6.76h	**76.02%**	6.92h	**76.77%**	6.66h

	Egg		Block		Pen	
Method	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

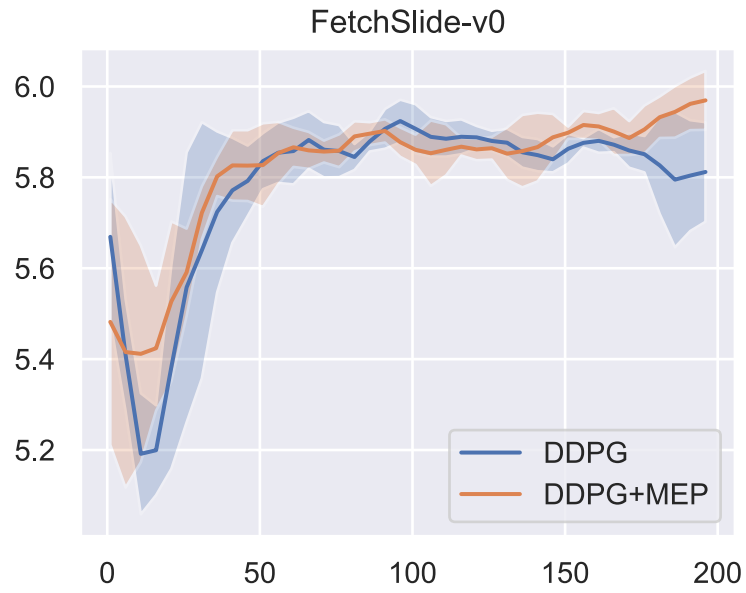
Entropy of achieved goals versus and training epoch



No MEP: 5.13 ± 0.33
With MEP: 5.59 ± 0.34



No MEP: 5.73 ± 0.33
With MEP: 5.81 ± 0.30



No MEP: 5.78 ± 0.21
With MEP: 5.81 ± 0.18

Summary and Take-home Message

- Our approach improves performance by nine percentage points and sample-efficiency by a factor of two while keeping computational time under control.

- Training the agent with many different kinds of goals, i.e., a higher entropy goal distribution, helps the agent to learn.

- The code is available on GitHub: https://github.com/ruizhaogit/mep

- Poster: 06:30 -- 09:00 PM @ Pacific Ballroom #32

Thank you!