Experience Replay Optimization

Daochen Zha, Kwei-Herng Lai, Kaixiong Zhou and Xia Hu

Department of Computer Science and Engineering, Texas A&M University

{daochen.zha, khlai037, zkxiong, xiahu}@tamu.edu

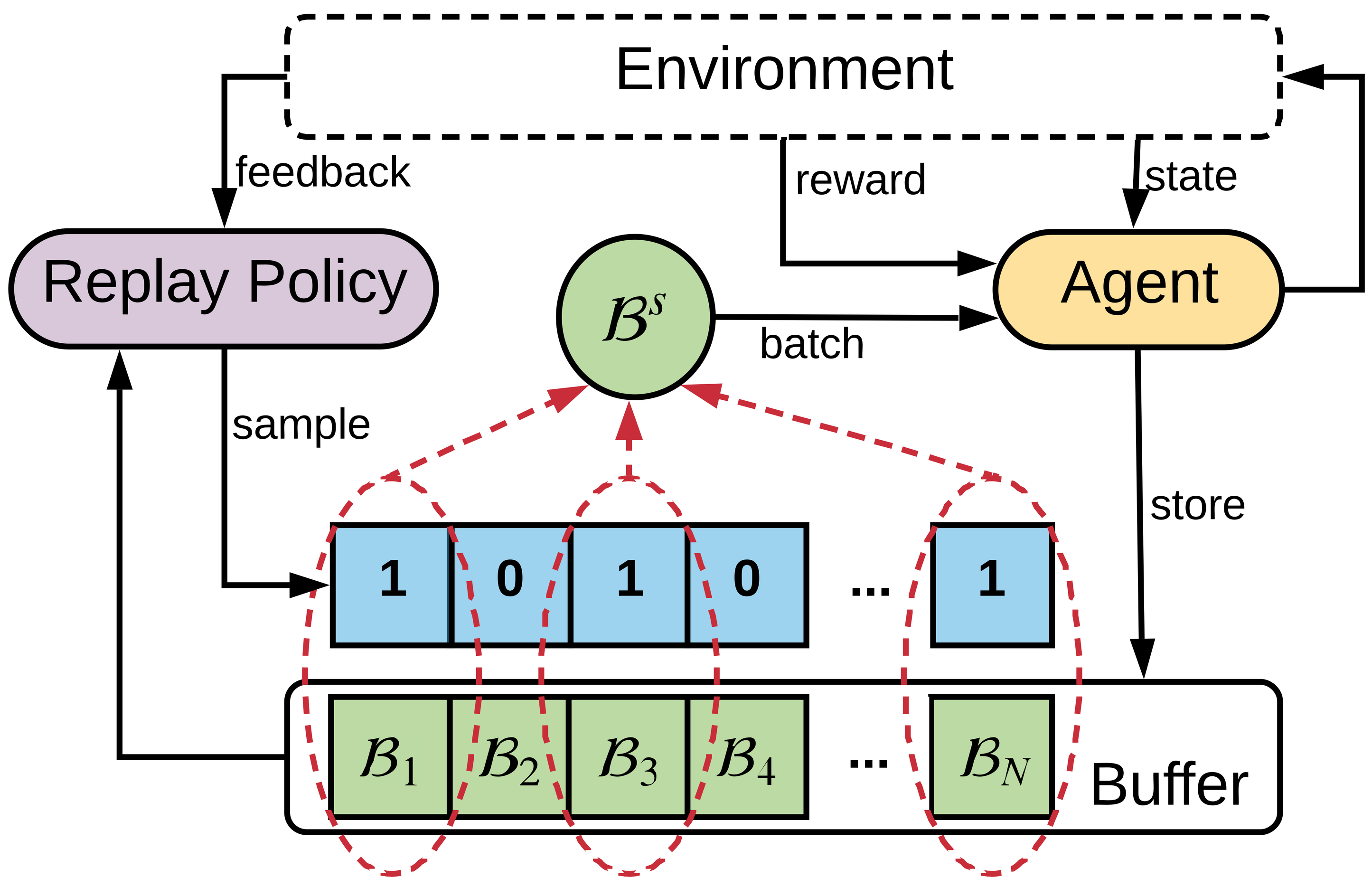
Introduction

- Experience replay is a memory buffer that stores past transitions (experiences) which are replayed for later use.
- It is a key technique behind off-policy RL algorithms. It greatly stabilizes the training and improves the sample efficiency.
- Uniform sampling is usually adopted. However, not all transitions are of equal importance: the agent can learn more efficiently from some experiences than from others.
- Some rule-based replay strategies have been studied. However, they may not be able to adapt to different tasks and algorithms.

Motivation

- Humans tend to replay the memories that will lead to the most rewarding future decisions.
- We are motivated to use the feedback from the environment as a rewarding signal to adjust the replay strategy.

Methodology



Sampling with Replay Policy: the replay policy is described as a priority score function $\phi(\mathbf{f}_{\mathcal{B}_i}|\theta^{\phi}) \in (0, 1)$, in which higher value indicates higher probability of a transition being replayed. We maintain a vector $\boldsymbol{\lambda}$ to store these scores:

$$\boldsymbol{\lambda} = \{\phi(\mathbf{f}_{\mathcal{B}_i}|\theta^{\phi})|\mathcal{B}_i \in \mathcal{B}\} \in \mathbb{R}^N. \tag{1}$$

In training, we then sample a subset $\mathcal{B}^s$ according to

$$\begin{aligned} \mathbf{I} &\sim \text{Bernoulli}(\boldsymbol{\lambda}), \\ \mathcal{B}^s &= \{\mathcal{B}_i|\mathcal{B}_i \in \mathcal{B} \wedge \mathbf{I}_i = 1\}. \end{aligned} \tag{2}$$

Then $\mathcal{B}^s$ is used to update the agent with standard procedures.

Training with Policy Gradient: The replay-reward is defined as the improvement of the cumulative reward:

$$r^r = r^c_{\pi} - r^c_{\pi'}. \tag{3}$$

By using the REINFORCE trick, we can calculate the gradient of the improvement $\mathcal{J}$ w.r.t θ^{ϕ}:

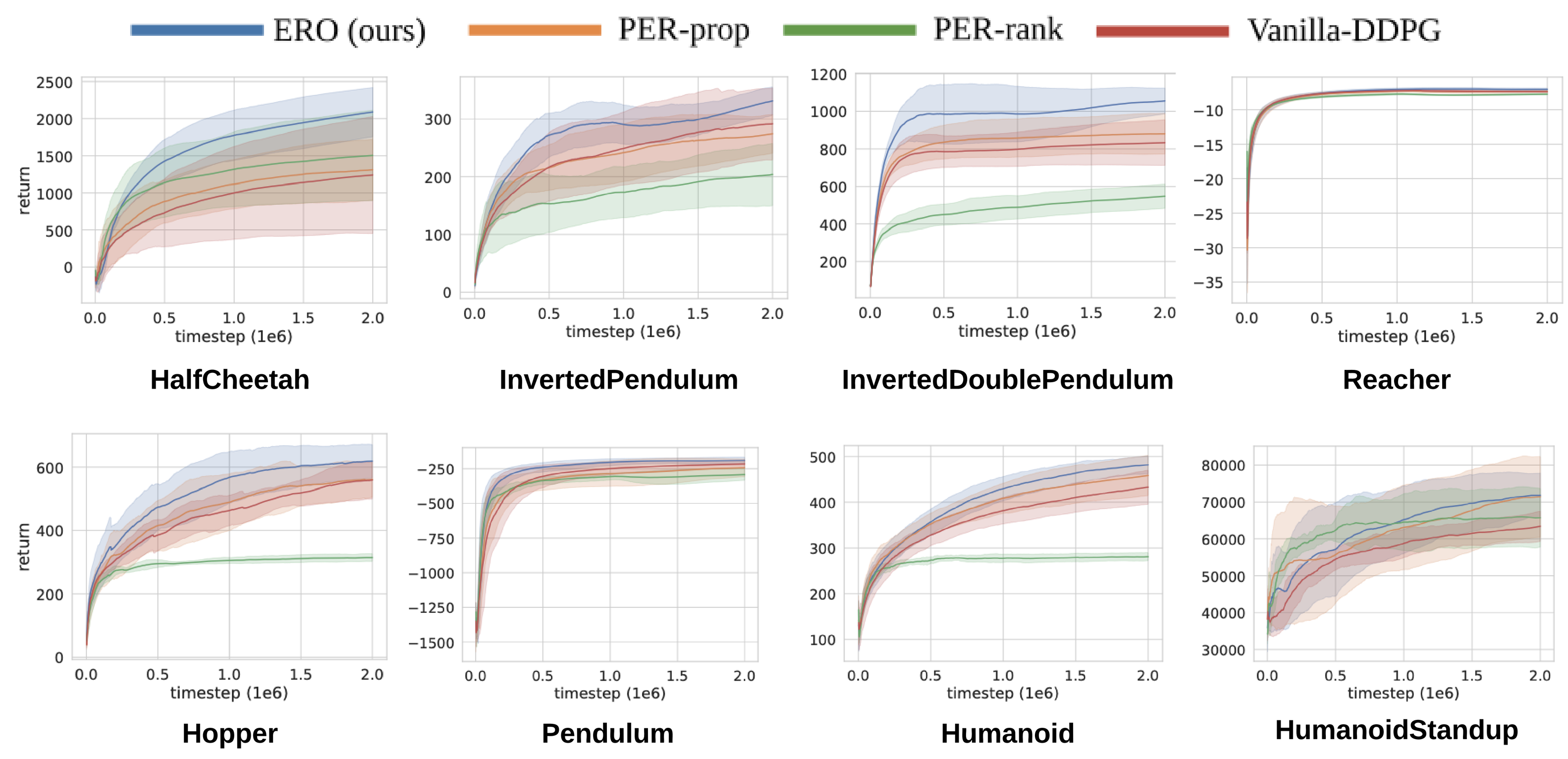
$$\begin{aligned} \nabla_{\theta^{\phi}}\mathcal{J} &= \nabla_{\theta^{\phi}}\mathbb{E}_{\mathbf{I}}[r^r] \\ &= \mathbb{E}_{\mathbf{I}}[r^r \, \nabla_{\theta^{\phi}} \log \mathcal{P}(\mathbf{I}|\phi)]. \end{aligned} \tag{4}$$

The resulting policy gradient can be written as:

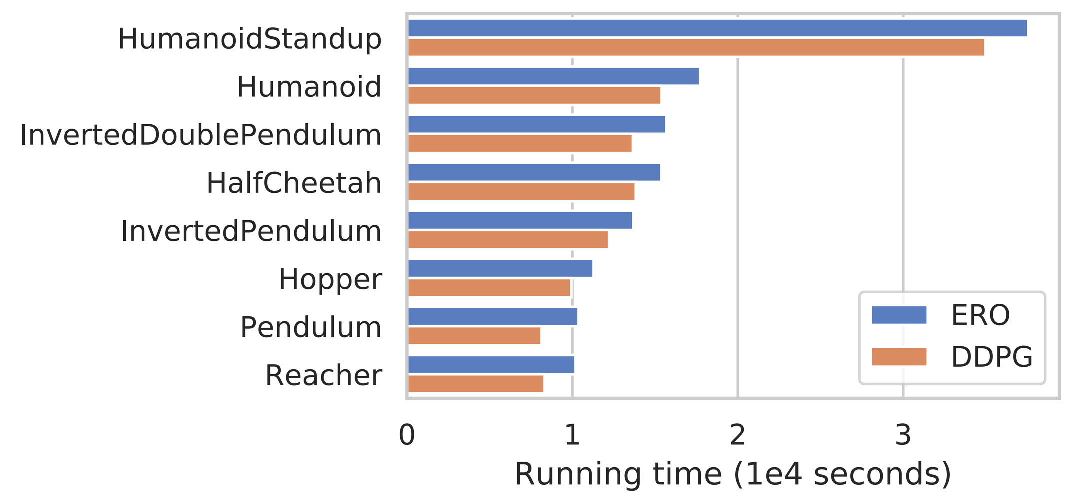
$$\nabla_{\theta^{\phi}}\mathcal{J} \approx r^r \sum_{i=1}^{N} \nabla_{\theta^{\phi}}[\mathbf{I}_i \log \phi + (1 - \mathbf{I}_i)\log(1-\phi)]. \tag{5}$$

Experiments

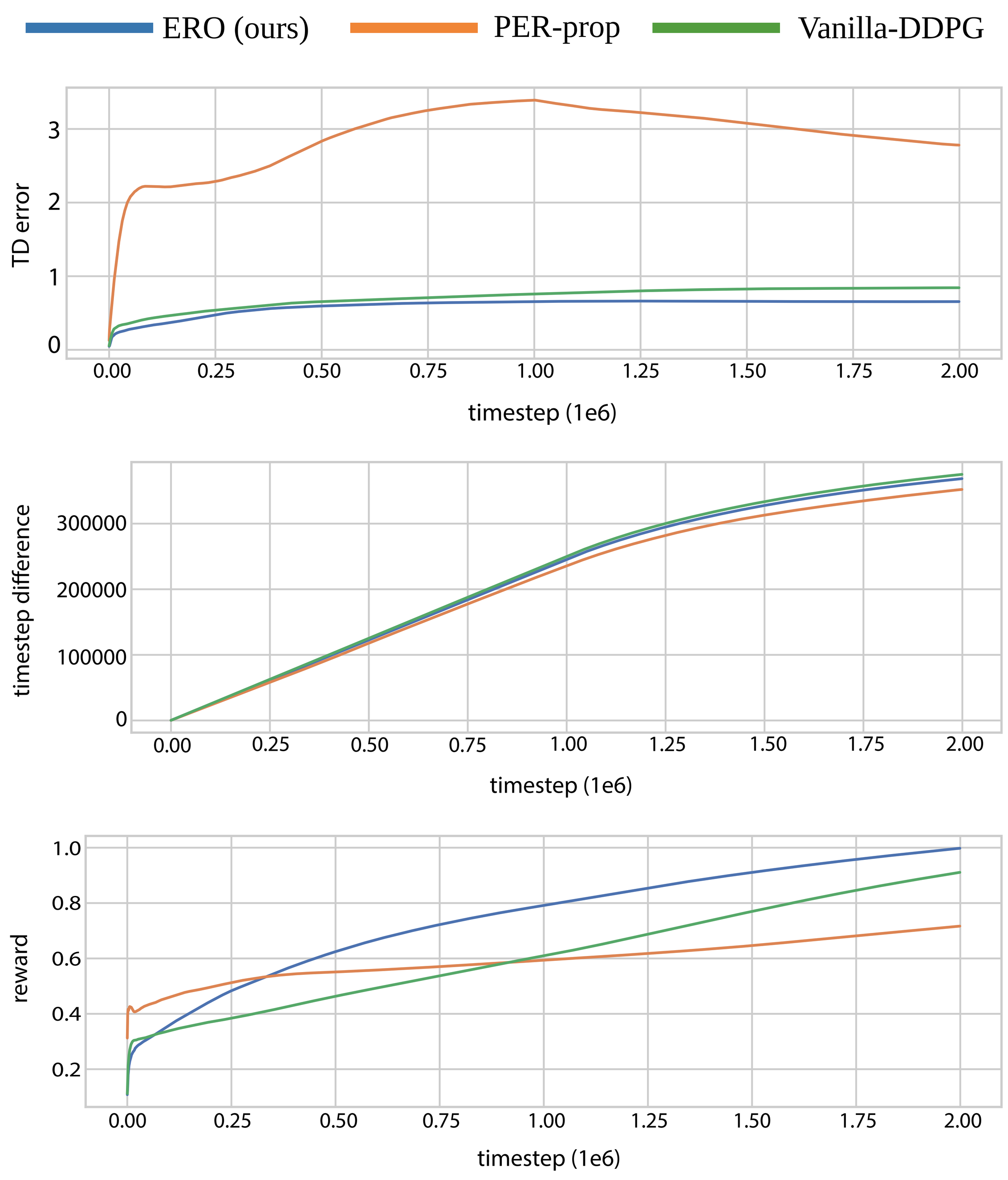
Performance Comparison



Efficiency Evaluation



Replay Policy Analysis



Observations

- The learned replay policy of ERO samples more transitions with low TD errors in HalfCheetah. More studies are needed to understand this aspect in the future work.
- ERO samples more recent transitions than Vanilla-DDPG. This suggests that recent transitions may be more helpful in this specific task.

Acknowledgements

The work is, in part, supported by National Science Foundation (#IIS-1718840 and #IIS-1750074).