A Meta-MDP Approach to Exploration for Lifelong Reinforcement Learning

Extended Abstract

Francisco M. Garcia
University of Massachusetts
Amherst, Massachusetts, USA
fmgarcia@cs.umass.edu

Philip S. Thomas
University of Massachusetts
Amherst, Massachusetts, USA
pthomas@cs.umass.edu

ABSTRACT

In this paper we consider the problem of how a reinforcement learning agent that is tasked with solving a sequence of reinforcement learning problems (Markov decision processes) can use knowledge acquired early in its lifetime to improve its ability to solve new problems. Specifically, we focus on the question of how the agent should *explore* when faced with a new environment. We show that the search for an optimal exploration strategy can be formulated as a reinforcement learning problem itself, albeit with a different timescale. We conclude with experiments that show the benefits of optimizing an exploration strategy using our proposed approach.

KEYWORDS

Reinforcement Learning; Hierarchical RL; Exploration

ACM Reference Format:
Francisco M. Garcia and Philip S. Thomas. 2019. A Meta-MDP Approach to Exploration for Lifelong Reinforcement Learning. In *Proc. of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019), Montreal, Canada, May 13–17, 2019,* IFAAMAS, 3 pages.

1 INTRODUCTION

One hallmark of human intelligence is our ability to leverage knowledge collected over our lifetimes when we face a new problem. When we first drive a new car, we do not re-learn from scratch how to drive a car. Instead, we leverage our experience driving to quickly adapt to the new car (its handling, control placement, etc.). Standard *reinforcement learning* (RL) methods lack this ability.; when faced with a new problem—a new *Markov decision process* (MDP)—they typically start from scratch, initially making decisions randomly to *explore* and learn about the current problem they face.

In this paper we focus on one aspect of lifelong learning: when faced with a sequence of MDPs sampled from a distribution over MDPs, how can a reinforcement learning agent learn an optimal policy for exploration? Specifically, we study the question of, given that an agent is going to explore, which action should it take?

After formally defining the problem of searching for an *optimal exploration policy*, we show that this problem can itself be modeled as an MDP. That is, the task of finding an optimal exploration strategy for a learning agent can be solved by another reinforcement learning agent that is solving a new *meta-MDP*: an MDP that operates at a different timescale, where one time step of the meta-MDP corresponds to an entire lifetime of the RL agent. This difference of timescales distinguishes our approach from previous meta-MDP methods for optimizing components of reinforcement learning algorithms, [2, 4, 5, 9, 10].

We contend that using random action selection during exploration (as is common when using Q-learning, [11], Sarsa, [8], and DQN, [6]) ignores useful information from the agent's experience with previous similar MDPs that could be leveraged to direct exploration. To address this problem, we separate the policies that define the agent's behavior into an exploration policy (which governs behavior when the agent is exploring) and an exploitation policy (which governs behavior when the agent is exploiting).

2 PROBLEM STATEMENT

We define the performance of the advisor's policy, μ, for a specific task $c \in C$ to be $\rho(\mu, c) = \mathbf{E}\left[\sum_{i=0}^{I}\sum_{t=0}^{T} R_t^i | \mu, c\right]$, where R_t^i is the reward at time step t during the i^{th} episode.

Let C be a random variable that denotes a task sampled from d_C. The goal of the advisor is to find an *optimal exploration policy*, μ^*, which we define to be any policy that satisfies:

$$\mu^* \in \underset{\mu}{\text{argmax}} \quad \mathbf{E}\left[\rho(\mu, C)\right]. \tag{1}$$

To optimize this objective, we formulate the problem of finding an optimal exploration policy as an RL problem where the advisor is itself an RL agent solving an MDP whose environment contains both the current task, c, and the agent solving the current task [1].

3 A GENERAL SOLUTION FRAMEWORK

Our framework can be viewed as a meta-MDP—an MDP within an MDP. From the point of view of the agent, the environment is the current task, c (an MDP). However, from the point of view of the advisor, the environment contains both the task, c, and the agent. At every time-step, the advisor selects an action U and the agent an action A. The selected actions go through a selection mechanism which executes action A with probability $1 - \epsilon_i$ and action U with probability ϵ_i at episode i. Figure 1 depicts the proposed framework with action A (exploitation) being selected. Even though one time step for the agent corresponds to one time step for the advisor, one episode for the advisor constitutes a lifetime of the agent (all of its interactions with a sampled task). From this perspective, wherein the advisor is merely another reinforcement learning algorithm, we can take advantage of the existing body of work in RL to optimize the exploration policy, μ.

[1] Full details and derivations of this formulation can be found at https://arxiv.org/abs/1902.00843

Proc. of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019), N. Agmon, M. E. Taylor, E. Elkind, M. Veloso (eds.), May 13–17, 2019, Montreal, Canada.

Problem Class	R	R+Advisor	PPO	PPO+Advisor	MAML
Pole-balance (d)	20.32 ± 3.15	28.52 ± 7.6	27.87 ± 6.17	**46.29 ± 6.30**	39.29 ± 5.74
Animat	−779.62 ± 110.28	**−387.27 ± 162.33**	−751.40 ± 68.73	−631.97 ± 155.5	−669.93 ± 92.32
Pole-balance (c)	—	—	29.95 ± 7.90	**438.13 ± 35.54**	267.76 ± 163.05
Hopper	—	—	13.82 ± 10.53	**164.43 ± 48.54**	39.41 ± 7.95
Ant	—	—	−42.75 ± 24.35	83.76 ± 20.41	**113.33 ± 64.48**

Table 1: Average performance on discrete and continuous control unseen tasks over the last 50 episodes. The use of the advisor leads to improved performance over random exploration.

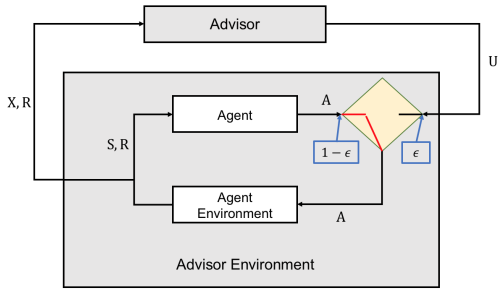


Figure 1: MDP view of interaction between the advisor and agent. At each time-step, the advisor selects an action U and the agent an action A. With probability ϵ the agent executes action U and with probability $1-\epsilon$ it executes action A.

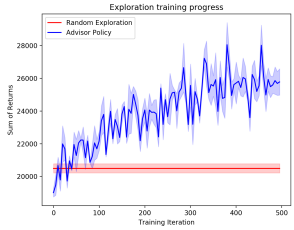
We experimented training the advisor policy using two different RL algorithms: REINFORCE, [12], and Proximal Policy Optimization (PPO), [7]. Pseudocode for an implementation of our framework using REINFORCE, where the meta-MDP is trained for I_{meta} episodes, is described in Algorithm 1. By updating μ every n steps, instead of the end of the lifetime of the agent, we can adapt the pseudocode to use PPO.

Algorithm 1 Agent + Advisor - REINFORCE

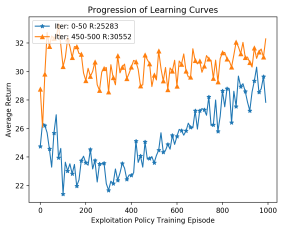
Initialize advisor policy μ randomly
for $i_{meta} = 0, 1, \ldots, I_{meta}$ **do**
  Sample task c from d_c
  for $i = 0, 1, \ldots, I$ **do**
    Initialize π to π_0
    $s_t \sim d_0^c$
    for $t = 0, 1, \ldots, T$ **do**
      $a_t \sim \begin{cases} \mu \text{ with probability } \epsilon_i \\ \pi \text{ with probability } (1-\epsilon_i) \end{cases}$
      take action a_t, observe s_t, r_t
    for $t = 0, 1, \ldots, T$ **do**
      update policy π using REINFORCE with s_t, a_t, r_t
  for $k = 0, 1, \ldots, IT$ **do**
    update policy μ using REINFORCE with s_k, a_k, r_k

4 EMPIRICAL RESULTS

We present experiments for discrete and continuous control tasks in the following problem classes: Pole-balancing [8], Animat [9], Hopper, and Ant [1]. We demonstrate that in practice the meta-MDP, M_{meta}, can be solved using existing RL methods, and the learned exploration policy leads to improved performance. We compare our framework to Model Agnostic Meta Learning (MAML) [3].



(a) Performance curves during training comparing advisor policy (blue) and random exploration policy (red).



(b) Average learning curves on training tasks over the first 50 advisor episodes (blue) and the last 50 advisor episodes (orange).

Figure 2: Advisor results on pole-balancing problem class.

Figure 2 helps us understand the behavior of our framework; these results were obtained on Cartpole. Figure 2a contrasts the cumulative return of an agent using the advisor for exploration (in blue) with the cumulative return obtained by an agent using ϵ-greedy random exploration (in red) during training over 6 training tasks. The exploitation policy, π, was trained using REINFORCE for I = 1,000 episodes and the exploration policy, μ, was trained using REINFORCE for 500 iterations. The horizontal axis corresponds to iterations—episodes for the advisor. The horizontal red line denotes an estimate (with standard error bar) of the expected return that an agent will obtain during its lifetime if it samples actions uniformly when exploring. The blue curve (with standard error bars from 15 trials) shows how the expected return that the agent will obtain during its lifetime changes as the advisor learns to improve its policy. Figure 2b shows the mean *learning curves* (episodes of an agent's lifetime on the horizontal axis and average return for each episode on the vertical axis) during the first and last 50 iterations. The mean return were 25,283 and 30,552 respectively.

Table 1 shows the average performance of the learned exploitation policy on novel tasks trained using different exploration strategies. The table compares the return of the best policy found by learning using REINFORCE (R) and PPO, with and without an advisor, and using MAML, given a maximum training time of 500 episodes.

5 CONCLUSION

In this work we developed a framework for leveraging experience to guide an agent's exploration in novel tasks, where the *advisor* learns the exploration policy used by the *agent* solving a task. We showed that a few sample tasks can be used to learn an exploration policy that the agent can use improve the speed of learning on novel tasks.

REFERENCES

[1] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. 2016. OpenAI Gym. (2016). arXiv:arXiv:1606.01540

[2] Fernando Fernandez and Manuela Veloso. 2006. Probabilistic Policy Reuse in a Reinforcement Learning Agent. In *Proceedings of the Fifth International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS '06)*. ACM, New York, NY, USA, 720–727. https://doi.org/10.1145/1160633.1160762

[3] Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. In *Proceedings of the 34th International Conference on Machine Learning (Proceedings of Machine Learning Research)*, Doina Precup and Yee Whye Teh (Eds.), Vol. 70. PMLR, International Convention Centre, Sydney, Australia, 1126–1135. http://proceedings.mlr.press/v70/finn17a.html

[4] Romain Laroche, Mehdi Fatemi, Harm van Seijen, and Joshua Romoff. 2017. Multi-Advisor Reinforcement Learning. (April 2017).

[5] Bingyao Liu, Satinder P. Singh, Richard L. Lewis, and Shiyin Qin. 2012. Optimal rewards in multiagent teams. In *2012 IEEE International Conference on Development and Learning and Epigenetic Robotics, ICDL-EPIROB 2012, San Diego, CA, USA, November 7-9, 2012*. 1–8. https://doi.org/10.1109/DevLrn.2012.6400862

[6] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. 2015. Human-level control through deep reinforcement learning. *Nature* 518, 7540 (Feb. 2015), 529–533.

[7] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal Policy Optimization Algorithms. *CoRR* abs/1707.06347 (2017). arXiv:1707.06347 http://arxiv.org/abs/1707.06347

[8] Richard S. Sutton and Andrew G. Barto. 1998. *Introduction to Reinforcement Learning* (1st ed.). MIT Press, Cambridge, MA, USA.

[9] Philip S. Thomas and Andrew G. Barto. 2011. Conjugate Markov Decision Processes. In *Proceedings of the 28th International Conference on Machine Learning, ICML 2011, Bellevue, Washington, USA, June 28 - July 2, 2011*. 137–144.

[10] Harm van Seijen, Mehdi Fatemi, Joshua Romoff, Romain Laroche, Tavian Barnes, and Jeffrey Tsang. 2017. Hybrid Reward Architecture for Reinforcement Learning.

[11] Christopher J. C. H. Watkins and Peter Dayan. 1992. Q-learning. In *Machine Learning*. 279–292.

[12] Ronald J. Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. In *Machine Learning*. 229–256.