Towards Active Imitation Learning

Chip Schaff*
TTI-Chicago

Falcon Z. Dai*
TTI-Chicago

Matthew R. Walter
TTI-Chicago

***Abstract*—Reinforcement learning has seen significant progress recently, solving complex tasks such as Atari games and Go. When combined with deep learning, model-free methods are able to learn directly from high-dimensional observations with a scalar reward. However, learning only from this often sparse reinforcement signal requires a sample complexity that is prohibitively high for direct application to robotic problems. Alternatively, imitation learning can significantly reduce the sample complexity by imitating the actions of a teacher. However, imitation learning can suffer from having suboptimal teachers and their supervision is usually expensive. In practice, we are often willing to solicit some (possibly suboptimal) supervision in exchange for shorter training time. In this work, we propose a novel framework for integrating *reinforcement and supervision* by introducing a *query action*. Our framework extends the reinforcement learning framework by allowing student agents to *actively seek* supervision from a teacher in the form of primitive actions. Furthermore, we add an explicit query cost when querying the teacher, which allows for a trade-off between sample complexity and the cost of teacher supervision. In addition, we propose a Q-learning solution based on the options framework.**

I. Introduction

Reinforcement learning (RL) and imitation learning (IL) are intimately related as they both try to find some "good" policy in a given interactive environment through learning, but with different feedback signals available. In RL, a policy is usually trained directly based upon reward feedback from the environment, i.e., reinforcement. IL considers domains for which the reward is unknown or otherwise difficult to specify, and exploits a teacher's actions in the same environment as supervision to train the "student" policy. While the overt goal is to *imitate* the given teacher's actions, the ultimate goal is to attain, if not surpass, the teacher's performance.[1] With this perspective, we inquire what we can gain given access to both the reward signal *and* a teacher's supervision in the form of primitive actions.

We propose a method based on the options framework [9] that extends the student's action set with a *query action*. This enables the student to actively seek supervision as well as estimate the value of querying the teacher in a given scenario. In the next section, we will describe our method in detail. Then, we will show describe some initial, proof-of-concept experiments. Finally, we will discuss ongoing work and directions for future work.

*These two authors contributed equally. The names were ordered randomly.

[1] In contrast, Judah et al. [5] treat imitation as the ultimate goal and instead use simple *manually constructed* reward functions to aid IL.

II. Method

Problem Formulation

Following Sutton and Barto [8], we represent a Markov decision process (MDP) as the tuple $(S, A, P_0, T, \gamma, R)$. Our goal is to learn a "student" policy $\pi_s : S \to A$ that attains high reward in this MDP. Suppose that, during training, we have access to a "teacher" policy $\pi_t : S \to A$. We can then extend this MDP with an option [9] $q = \langle S, \pi_t, \beta \rangle$ that queries the teacher, where S is the initiation set and $\beta : S \to [0, 1]$ determines the termination probability at each state. Additionally, we add a *query cost* $c \geq 0$ to penalize querying the teacher. Sutton et al. [9] show that the addition of this option creates a semi-Markov decision process (SMDP) and that the existence of an underlying MDP allows us to learn about multiple options from the execution of one. During training, this allows us to use the teacher's demonstrations to provide feedback to the student both on how it should act and how to best utilize the teacher. This formulation can be extended to include multiple teachers (each following different policies) with different query costs. This is done by adding an additional option for each teacher.

Intra-Option Q-Learning

If the query cost $c > 0$, then the SMDP and the original MDP share the same set of optimal policies. Therefore, it is sufficient to learn an optimal policy in the SMDP and use that policy in the original MDP. To do this, we propose to use intra-option Q-learning [9]. Intra-option Q-learning updates Q-values using the one-step temporal difference update:

$$Q(s_t, o) \leftarrow Q(s_t, o) + \alpha[r_t + \gamma U(s_{t+1}, o) - Q(s_t, o)] \quad (1)$$

where $U(s_t, o)$ is the state-option value function:

$$U(s, o) = (1 - \beta_o(s))Q(s, o) + \beta_o(s) \max_{o' \in O} Q(s, o') \quad (2)$$

This update is applied to all options (including primitive actions) consistent with the action taken at time t. Therefore, whenever the teacher is queried, this algorithm allows us to update our estimates of both the query option and the primitive actions taken by the teacher. In order to handle high dimensional state spaces, we use function approximation to model the student Q-function, i.e., the student Q-function $Q'(s, o; \theta)$ is parametrized by some θ. At test time, we use the same parameters θ and restrict the student policy to only consider primitive actions:

$$\pi_{\text{test}}(s) = \arg\max_{a \in A} Q'(s, a; \theta) \quad (3)$$

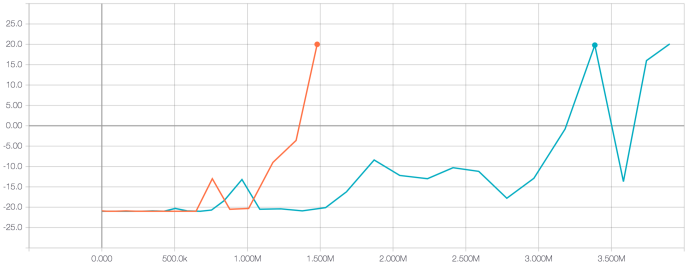

Fig. 1. Results on Pong comparing the fastest student to the fastest policy without access to a teacher. The orange curve shows the episodic rewards of the student when it is restricted to primitive actions. The turquoise curve shows the training for a Q-Learning agent with the same hyper-parameters, except it is not allowed to query a teacher. The dot on each curve represents the point at which the environment was solved.

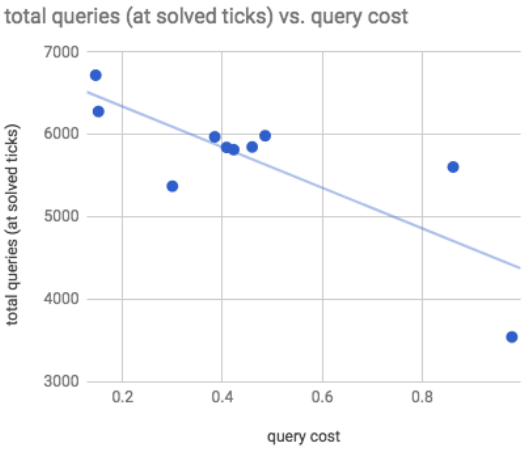

Fig. 2. Early results on Cart-pole problem. The number of total queries made when the task is considered solved negatively correlates with the query cost. Each dot corresponds to a different run with different query costs.

III. Preliminary Results

We evaluate our approach through a series of simulated experiments with a previously trained agent serving as the teacher. These experiments examine the efficiency of our method in terms of both *training speed* and the amount of teacher supervision needed, compared to the standard RL methods without supervision. Currently, we have evaluated the method on the cart-pole problem and the Atari game Pong [1], both using the OpenAI Gym [2].

In order to study the *training speed*, we record the amount of experience (measured by ticks) when the student first surpasses a high performance threshold without help from the teacher. In order to avoid high variance in this metric, we roll out 10 episodes and check the average performance against the threshold. We consider the problem solved when the average performance reaches 90% of the optimal reward. [2]

In both environments, we observed that the student was able to learn to achieve the optimal episodic rewards. In the cart-pole problem, where we were able to perform more experiments, most of the students stop querying and learn to solve the problem on their own. Furthermore, the students under a higher query cost are more likely to stop querying sooner (see Fig. 2), which is what we expected.

In the cart-pole problem, we observe that the average training speed of the student as measured by our metric is about 16% faster than a similar model trained without a teacher's supervision. However, the large standard deviations in both groups render the difference statistically insignificant. In Pong, however, we find that the average training speed of the student is 50% faster with teacher supervision. Even though this result is based on a small sample size of 5, every student was able to solve the environment faster than each baseline policy. On average, the Pong students queried the teacher 11000 times before solving the environment.

IV. Discussion

There has been considerable recent work that leverages teacher demonstrations as an initialization before training with reinforcement, for example, AlphaGo's policy was initialized by supervised learning over recorded human Go plays [7]. Hester et al. [4] further studied an initialization technique that relies only on an *offline* collection of recorded teacher demonstrations. In our problem setting, we assume that an *online* teacher can be queried by the student during training. But unlike DAGGER [6], we do not query the online teacher's decisions on all of the states encountered by the student. Instead, we let the student decide when to query. This allows for selective queries and possibly a smaller amount of supervision. Our proposed problem framework is closely related to the AskForHelp framework proposed by Clouse [3] which uses a fixed query criterion on top of the Q-learning framework.

[2]We recognize the obvious limitation of this metric's applicability in other environments where the upper bound of attainable performance is unclear.

V. Conclusion and Future Work

The initial results we obtained suggest that incorporating supervision with reinforcement in our proposed framework can help reduce training sample complexity. We also showed that the "student" agent can learn a reasonable query policy whose query frequency negatively correlates with query cost, allowing practitioners to trade-off between sample complexity and expert supervision.

Currently, we are continuing to experiment in more domains and eventually hope to apply this technique to complex robotics problems. We also aim to study in-depth the impact of providing different amounts supervision in response to a query action. So far, the impact of temporally extended sequences of supervised actions is inconclusive and we suspect it depends on the complexity of the task.

Besides improvements to our proposed method, we are working to provide a more meaningful comparison with several closely related methods (though not specifically designed to solve our proposed SMDP), such as warm start from imitating recorded teacher demonstrations [4] and using a fixed query policy [3].

REFERENCES

[1] Marc G Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. The arcade learning environment: An evaluation platform for general agents. *J. Artif. Intell. Res.(JAIR)*, 47:253–279, 2013.

[2] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. OpenAI Gym, 2016.

[3] Jeffery Allen Clouse. *On integrating apprentice learning and reinforcement learning*. PhD thesis, University of Massachusetts, Amherst, January 1996.

[4] Todd Hester, Matej Vecerik, Olivier Pietquin, Marc Lanctot, Tom Schaul, Bilal Piot, Andrew Sendonaris, Gabriel Dulac-Arnold, Ian Osband, John Agapiou, Joel Z. Leibo, and Audrunas Gruslys. Learning from demonstrations for real world reinforcement learning. *arXiv:1704.03732*, April 2017.

[5] Kshitij Judah, Alan Paul Fern, Prasad Tadepalli, and Robby Goetschalckx. Imitation learning with demonstrations and shaping rewards. In *Proc. Nat'l Conf. on Artificial Intelligence (AAAI)*, pages 1890–1896, 2014.

[6] Stphane Ross, Geoffrey J. Gordon, and Drew Bagnell. A Reduction of Imitation Learning and Structured Prediction to No-Regret Online Learning. In *Int'l Conf. on Artificial Intelligence and Statistics (AISTATS)*, volume 1, page 6, 2011.

[7] David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489, January 2016.

[8] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press Cambridge, 1998.

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