Agents Teaching Agents in Reinforcement Learning (Nectar Abstract)

Matthew E. Taylor[1] and Lisa Torrey[2]

[1] Washington State University, School of EECS, Pullman, WA, USA
eecs.wsu.edu/~taylorm/, mtaylor@eecs.wsu.edu
[2] St. Lawrence University, Math and Computer Sciences, Canton, NY, USA
myslu.stlawu.edu/~ltorrey/, ltorrey@stlawu.edu

1 Introduction

Using reinforcement learning [4] (RL), agents can autonomously learn a control policy to master sequential-decision tasks. Rather than always learning *tabula rasa*, our recent work [5, 7, 8] considers how an experienced RL agent, the *teacher*, can help another RL agent, the *student*, to learn. As a motivating example, consider a household robot that has learned to perform tasks in a household. When the consumer purchases a new robot, she would like the student robot to quickly learn to perform the same tasks as the teacher robot, even if the new robot has different state representation, learning method, or manufacturer. Our goals are to: 1) Allow the student to learn faster with the teacher than without it, 2) Allow the student and teacher to have different learning methods and knowledge representations, 3) Not limit the student's performance when the teacher is sub-optimal, 4) Not require a complex, shared language, and 5) Limit the amount of communication required between the agents.

Our approach was influenced by learning from demonstration [1] (LfD) and transfer learning [6] (TL). LfD methods typically do not achieve goals 3 and 5, limiting an agents' performance to that of the demonstrator, and requiring many trajectory demonstrations. The majority of TL methods assume that the trained agent knows the new agent's learning method or knowledge representation, failing to meet goal 2, and assumes direct access to the the "brain" of the student agent, failing goals 4 or 5.

We investigate how an RL agent can best teach another RL agent using a limited amount of advice, assuming that the teacher can observe the student's state and that the student can receive (and execute) action advice from the teacher. The teacher can give advice a fixed number of times, but cannot observe or change anything internal to the student. This paper presents three of our teaching algorithms and shows a selection of results in the Ms. Pac-Man domain, although our work has also evaluated our methods in the Mountain Car and StarCraft domains. A key insight is that the same amount of advice, given at different moments, can have different effects on student learning. Results show our teaching methods can achieve all five of the above goals.

2 Teaching on a Budget

In this setting, a teacher RL agent has learned a (potentially sub-optimal) policy π_T for a task. Using this fixed policy, it will assist a student agent. As the student learns using RL, the teacher will observe each state s the student encounters and each action a the student takes. The teacher has a fixed budget — it may suggest an action at most b times

for the student's current state using the teacher's policy: $\pi_t(s)$. Theoretically calculating how the teacher should best spend its advice is difficult except in the simplest of RL problems. We instead take an experimental approach to this question, proposing and testing several algorithms for deciding when to give advice.

Early Advising Students should benefit more from advice early on, when they know very little. Early Advising serves as a baseline where the teacher always provides advice for the first b states the student encounters.

Importance Advising When all states in a task are equally important, Early Advising should be an effective strategy. However, we hypothesized that some states are more important for learning than others, and saving advice for more important states would be a more effective strategy. In some situations, the wrong action could cause catastrophic failure, while in other situations, any action may be acceptable and none are disastrous. A teacher that is conscious of state importance could give advice only when it reaches some threshold th. We call this approach *Importance Advising* (Fig. 1, left).

When th is 0, this becomes equivalent to Early Advising, assuming importance values are non-negative. In this work, we consider teachers that learn Q-functions. If the teacher estimates the Q-values for all actions in s to be the same, it does not matter which action is selected.[3] However, if some actions have larger Q-values than others in s, the action selected matters and the state has some importance. We therefore define state importance as: $I(s) = \max_a Q(s,a) - \min_a Q(s,a)$.

Predictive Advising The final algorithm builds upon Importance Advising by attempting to provide advice only when 1) the teacher evaluates the current state to be important and 2) the teacher believes the student will execute a sub-optimal action. Although the teacher cannot access the student's policy, it may be able to infer the policy through observation — by observing s, a pairs the student has executed, the teacher can train a classifier to predict student actions, π_s.

If a teacher's action predictor performs perfectly, the teacher will never "waste" advice on a state where the student would execute the action that would have been advised. Inaccurate predictions may waste advice, or miss opportunities for the teacher to give useful advice. Our implementation used the *SVM-Light* software package [2] where the SVM used the teacher's state representation of observed states as input and observed student actions as classification labels. The teacher trains a new SVM after each episode using training examples from the previous episode.[4] This classification task is inherently challenging: a student's behavior is non-stationary and includes exploration, and differences between teacher and student state representations mean that the classifier's hypothesis space may not even be able to represent the student's policy.

3 Experiments in Ms. Pac Man

Our experiments use an implementation based on the Ms. Pac-Man vs. Ghosts [3] competition. Pac-Man can move in the four cardinal directions but only some of these ac-

[3] $I(s)$ is computed with the teacher's learned Q-function, leveraging the teacher's knowledge without requiring any knowledge of the student's policy or internal representation.

[4] Training the SVM required approximately 1 second — in other domains, it may be possible to update the SVM during episodes or use incremental update methods to improve performance.

procedure IMPORTANCEADVISE(π_t, b, th)
  for each student state s **do**
    if $b > 0$ and $I(s) \geq th$ **then**
      $b \leftarrow b - 1$
      Advise $\pi_t(s)$

procedure PREDICTIVEADVISE(π_t, b, th)
  for each student state s **do**
    if $b > 0$ and $I(s) \geq th$ and
      $\pi_s(s) \neq \pi_t(s)$ **then**
      $b \leftarrow b - 1$
      Advise $\pi_t(s)$

Fig. 1: Both teaching algorithms use a teacher's policy, the budget, and a threshold

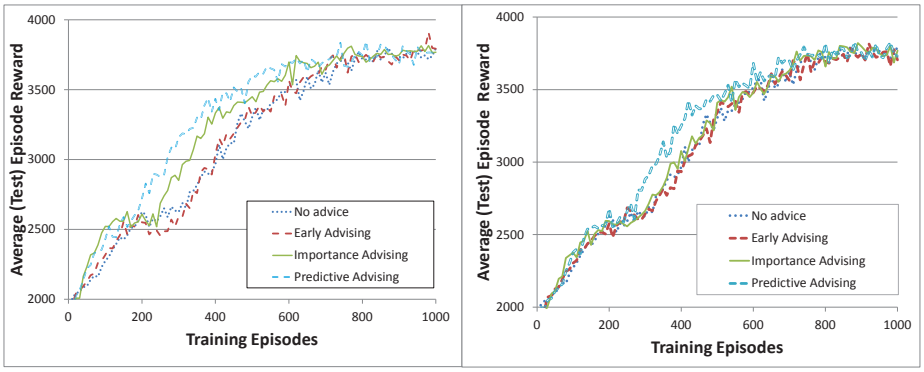

Fig. 2: Students with the high-asymptote feature learn from high-asymptote teachers (left) and low-asymptote teachers (right)

tions are available in many states of the maze. Points are awarded for eating the small food pellets, larger power pellets, or ghosts shortly after eating a power pellet. The episode ends if any ghost touches Pac-Man, Pac-Man eats all pellets, or after 2000 steps. Useful high-level features tend to describe distances, such as "the distance from Pac-Man to the nearest food pellet," detailed in the released code.[5]

We defined two feature sets. One representation uses 16 features; this "low-asymptote" feature set allows agents to average 2,250 points per episode after training. The other uses 7 heavily-engineered features; this "high-asymptote" feature set allows agents to average 3,380 points after training because they typically learn to eat at least one ghost. Teachers have an advice budget of 1000, roughly half the number of steps in a single well-played episode. The RL parameters that agents use for learning with the SARSA(λ) algorithm are $\epsilon = 0.05$, $\alpha = 0.001$, $\gamma = 0.999$, and $\lambda = 0.9$.

In Fig. 2, students in both figures use a high-asymptote feature set. In the left figure, the teacher also uses the high-asymptote feature set. In this setting, Early Advising fails to improve over no advice, but Importance Advising significantly outperforms both, and Predictive Advising significantly outperforms all methods. In the right figure, the teacher uses the low-asymptote feature set. In this case, only Predictive Advising significantly outperforms learning without advise.[6] Even though the teacher has a different representation of state and an average performance of only 2,250 points, it is able to significantly improve student learning performance.

[5] `http://www.eecs.wsu.edu/~taylorm/13ConnectionScience.html`

[6] In both experiments, the SVM embedded in the Predictive Advising achieves an 86% accuracy.

Teachers provide advice in only a small fraction of the training steps but can still significantly improve student learning. How quickly teachers provide advice is partly controlled by the importance threshold, th. Teachers in Fig. 2 (right) perform best by using their advice quickly before the students surpass them ($th = 50$). Teachers in Fig. 2 (left) perform better by using less frequent advice over longer periods ($th = 250$).

4 Conclusions and Future Work

As more problems become solvable by agent-based methods, it is important for agents to be able to work together, even if they have very different implementations. RL agents succeed at learning control policies for specific tasks, and allowing them to serve as teachers for these tasks can significantly improve the speed and applicability of RL for fielded agents. Our experimental results, a sample of which were presented in the previous section, lead us to the following conclusions about teaching with an advice budget. First, student learning can be improved with a small advice budget. Second, advice can have greater impact when it is spent on more important states. Third, when teachers can successfully predict student mistakes, they can use their advice budget more effectively. Fourth, students can benefit from advice even from teachers with less inherent ability, different representations, and different learning methods.

There are many exciting directions for future work. The concept of state importance could benefit from further investigation: there may exist better domain-specific ways to measure state importance or effective strategies for automatically selecting and adjusting importance thresholds. The teaching framework could be extended to include multiple teachers and/or students. Students currently always execute actions suggested by the teacher — in the future, the student could decide to ignore advice, or proactively ask the teacher for advice. We are interested in testing our method on other student learning methods, and modifying our methods to work other teacher learning methods. Finally, we are also excited about testing our teaching algorithms with human students.

Acknowledgements This work was supported in part by NSF IIS-1149917.

References

1. B. Argall, S. Chernova, M. Veloso, and B. Browning. A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, 57(5):469 – 483, 2009.
2. T. Joachims. Making large-scale SVM learning practical. In B. Scholkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*. MIT Press, 1999.
3. P. Rohlfshagen and S. M. Lucas. Ms Pac-Man versus Ghost Team CEC 2011 competition. In *Proc. of the IEEE Congress on Evolutionary Computation*, 2011.
4. R. S. Sutton and A. G. Barto. *Introduction to Reinforcement Learning*. MIT Press, 1998.
5. M. E. Taylor, N. Carboni, A. Fachantidis, I. Vlahavas, and L. Torrey. Reinforcement learning agents providing advice in complex video games. *Connection Science*, 26(1):45–63, 2014.
6. M. E. Taylor and P. Stone. Transfer learning for reinforcement learning domains: A survey. *Journal of Machine Learning Research*, 10(1):1633–1685, 2009.
7. L. Torrey and M. E. Taylor. Towards student/teacher learning in sequential decision tasks (poster). In *Proc. of the Int't. Conf. on Autonomous Agents and Multiagent Systems*, 2012.
8. L. Torrey and M. E. Taylor. Teaching on a budget: Agents advising agents in reinforcement learning. In *Proc. of the Int't. Conf. on Autonomous Agents and Multiagent Systems*, 2013.