Fast Customization Using Human-Feedback in Assistive Robotics

James Staley

1 Introduction

Robots and Assistive Agents that can learn from caregivers or users are much more likely to be used. An agent will leave the factory with a standardized set of abilities, but standard implementations will not handle every users needs. An agent that can alter its performance based on live feedback will be more useful and much more likely to gain acceptance into peoples lives.

Taking this one step further, an agent should be able to adjust its behavior in real-time – fast enough that the user can tell the robot is reacting to their feedback. Unfortunately, user-feedback tends to be sparse and inconsistent. Therefore, it may be helpful to spread the reward from user feedback throughout the tabular value function during planning steps to speed up learning.

2 Background Related Work

The reward signal is a defining aspect of reinforcement learning. An agent repeatedly explores a state-action space, building up a policy as it encounters reward. A trained agent uses its learned policy to take the action that maximized its expected future reward given the current state. Typically the reward is given by the environment, but recent work has focused on incorporating human-feedback into training.

Researchers are looking at several different methods for incorporating human-feedback. The learning system may treat human-feedback as an additional source of environmental reward in a process known as reward shaping [3]. In this case when the user gives positive or negative feedback it is simply weighted and summed with the environment reward for the credited transition. This is easy to implement and straightforward, but it doesn't treat human-feedback as anything special. Human-feedback can be considered to contain high-level information about the task, and so there may be some benefit in distinguishing it from environmental reward.

TAMER [2] ignores any environmental reward and uses human-feedback to determine a value function by updating either a table entry (in tabular learning) or a weights vector (in function approximation). This method leads to rapid

early learning that can be expanded upon through more human-feedback or other reinforcement learning techniques.

Policy Shaping [1] retains environmental reward, but keeps a separate policy based on received feedback and combines the two when deciding which actions to take. The learned feedback policy must make an assumption about the consistency and frequency of the user's feedback, but otherwise requires fewer parameters than other reinforcement algorithms with comparable performance.

Directly relevant to our use case, Pilarski et al. [4] train an agent to map electrical muscle signals to prosthetic arm motion using human-feedback via an Actor-Critic system. The Actor-Critic system uses an actor, which selects actions and shapes the policy, and a critic, which estimates the value function used to update the weight vector. They use tile coding to approximate S as a binary feature vector $x(s)$.

All of the above methods use simple binary feedback where the user can approve of, disapprove of, or make no comment on the behavior of the user.

3 Technical Approach / Methodology / Theoretical Framework

We plan to approach this problem using the simplest representation possible, and then scaling up complexity as is required. Initially we'll perform tabular Q-learning ()1) where our reward will be provided through human-feedback by a TAMER-like algorithm.

We'll use Dyna-Q [5] to speed up the effects of feedback on behavior. Dyna-Q keeps a model of our state-action transitions and simulated n action for each action our agent takes in the real world. This will more rapidly propagate the feedback we get throughout our table and enable a user to see the results of their feedback in real time. Using Dyna-Q may help us clarify whether user-feedback *should* be propagated throughout a table in the way reward is.

$$Q(S, A) \leftarrow Q(S, A) + \alpha[R + \gamma \max Q(S', A') - Q(S, A)] \tag{1}$$

3.1 Tasks

Subsections are useful for breaking down the problem into sub-parts. For example, you could break down the tasks for your project and list them one by one.

4 Evaluation

Describe how you will evaluate your approach/solution. What constitutes success? What metrics will you use? Do you have any preliminary hypothesis that you plan to test? Also, describe the RL domain or environment you plan to use.

5 Timeline and Individual Responsibilities

State the timeline in terms of weeks and milestones you want to achieve. If working on a team, state what the individual responsibilities are at this point (i.e., who is going to do what, these may of course change over the course of the project). [**?**].

References

[1] Shane Griffith, Kaushik Subramanian, Jonathan Scholz, Charles L Isbell, and Andrea L Thomaz. Policy Shaping: Integrating Human Feedback with Reinforcement Learning. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 2625–2633. Curran Associates, Inc., 2013.

[2] W. Bradley Knox and P. Stone. TAMER: Training an Agent Manually via Evaluative Reinforcement. In *2008 7th IEEE International Conference on Development and Learning*, pages 292–297, August 2008.

[3] Andrew Y Ng, Daishi Harada, and Stuart Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *ICML*, volume 99, pages 278–287, 1999.

[4] P. M. Pilarski, M. R. Dawson, T. Degris, F. Fahimi, J. P. Carey, and R. S. Sutton. Online human training of a myoelectric prosthesis controller via actor-critic reinforcement learning. In *2011 IEEE International Conference on Rehabilitation Robotics*, pages 1–7, June 2011.

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