Supporting End Users in Defining Reinforcement-Learning Problems for Human-Robot Interactions

Valerie Zhao
Department of Computer Science
University of Chicago
Chicago, IL 60637
vzhao@uchicago.edu

Michael L. Littman
Department of Computer Science
Brown University
Providence, RI 02912
michael_littman@brown.edu

Shan Lu
Department of Computer Science
University of Chicago
Chicago, IL 60637
shanlu@uchicago.edu

Sarah Sebo
Department of Computer Science
University of Chicago
Chicago, IL 60637
sarahsebo@uchicago.edu

Blase Ur
Department of Computer Science
University of Chicago
Chicago, IL 60637
blase@uchicago.edu

Abstract

Reinforcement learning (RL) can help agents learn complex tasks that would be hard to specify using standard imperative programming. However, end users may have trouble personalizing their technology using RL due to a lack of technical expertise. Prior work has explored means of supporting end users after a problem for the RL agent to solve has been defined. Little work, however, has explored how to support end users *when* defining this problem. We propose a tool to provide structured support for end users defining problems for RL agents. Through this tool, users can (i) directly and indirectly specify the problem as a Markov decision process (MDP); (ii) receive automatic suggestions on possible MDP changes that would enhance training time and accuracy; and (iii) revise the MDP after training the agent to solve it. We believe this work will help reduce barriers to using RL and contribute to the existing literature on designing human-in-the-loop systems.

Keywords: End-User Programming, Reinforcement Learning, Markov Decision Process, Human-Robot Interaction, Human-in-the-Loop

Acknowledgements

This material is based upon work supported by the National Science Foundation under Grants CCF-1837120 and CCF-1836948. We thank Yunfei Shen and Lilith Yu for their contributions to this work.

1 Introduction

Reinforcement learning (RL) is applicable to a variety of domains [Sutton and Barto, 1998, Barto et al., 2017, Kober et al., 2013], but end users without sufficient technical expertise may have trouble applying it to their own needs. They may face challenges such as determining how to train the RL agent efficiently and how to define and administer rewards. Existing work has attempted to support end users by incorporating user feedback through policy shaping, guided exploration, and augmented value functions [Arzate Cruz and Igarashi, 2020].

While most work supports end users in training the RL agent on a defined task, little work has attempted to address end-user challenges in *defining* that task. It may appear straightforward to define an RL task as a Markov decision process (MDP) with a set of states, a set of actions, and a set of rewards. However, users might find it difficult to anticipate the appropriate state space and action space. Specifying too many states or actions can cause the agent to learn over an excessively long period of time. Specifying too few, on the other hand, may cause the agent to learn a suboptimal policy.

We propose a tool for assisting end users in defining and refining RL problems. Our tool consists of a graphical interface that guides the user in defining RL problems both before and after training an agent. Using built-in primitives, users will first directly or indirectly specify the states, actions, and rewards of the MDP. The tool will generate recommendations for refining this MDP, which the user can accept or reject. The RL agent will then train on the resulting MDP. (For our tool, we assume a Q-learning agent.) The interface shows the policy at each timestep to help the user evaluate whether their MDP appropriately captures the intended task. When training terminates, the user can accept the final policy, revise it, or revise the MDP definition and try again.

Our work focuses on RL applications in Human-Robot Interaction (HRI), which concern social interactions between people and robots. In HRI tasks, robots must behave in a socially appropriate manner by adapting to personal preferences, cultural norms, and other context-specific factors [Andrist et al., 2015, Wang et al., 2010, Lee et al., 2012, Leyzberg et al., 2014]. One example in HRI is the issue of robot abuse. Multiple public incidents have occurred in which people purposefully blocked the robots' path or sensors, hurled obscenities against them, or resorted to violent behaviors like kicking and hitting [Nomura et al., 2015, Brščić et al., 2015, Salvini et al., 2010]. Robots designed to accomplish tasks unsupervised must somehow mitigate these incidents.

How can a user help the robot navigate complex social interactions, such as avoiding or pacifying incidents of robot abuse? Programming languages like Python for the Pepper robot[1] require substantial technical knowledge. End-user programming paradigms and systems, such as Choreographe [Pot et al., 2009], demands less technical knowledge but still requires the user to anticipate the ideal robot behavior under a specific circumstance for all possible circumstances. A similar problem arises for Learning from Demonstration (LfD).

In contrast, an RL approach can help the robot adapt to the many scenarios it will encounter without requiring the user to detail each possibility, making it especially well suited to those with minimal programming and domain expertise. Unfortunately, to leverage RL for this problem, the user still needs advanced knowledge of the environment. In our example, users may neglect to realize that states related to bystanders may affect the likelihood of robots being bullied, choosing instead to only consider states that relate to the bullies and the robot. Our tool seeks to address this gap, with a focus on HRI tasks but may also be extended to other applications.

2 Vision for the Interaction

We envision a multi-stage interaction between the user and the RL system. Via a graphical interface, the user will first specify the states, actions, and rewards of the MDP (Figure 1) by selecting from a set of system-defined choices (Figure 2). However, some users may find it difficult to think about RL tasks in the terms of MDP. The interface will offer these users the option to, instead, define the task by predicting parts of the optimal policy. They will describe these predictions in the form of IF-THEN (trigger-action) statements, and the system will extract a potential MDP based on these statements. For example, if the user writes "IF a person starts hitting the robot THEN the robot should run away," the system will infer that "whether there is a person hitting the robot" is a relevant state and "robot running away from people" is a relevant action for the MDP overall.

The system will review the MDP and recommend possible changes to improve the training efficiency and accuracy. These recommendations may include removing redundant state features and actions, or adding those that seem relevant to the task. The user accepts or rejects these recommendations based on their understanding of their intended task, producing a tentative MDP on which to train the robot.

Training the robot will produce information for debugging the MDP. During training, the interface will show the policy of the robot at each timestep. The user can monitor this to understand the trends in the robot's learning, such as how quickly they acquire the right behaviors. When the robot has completed learning, the interface visualizes the final policy.

[1] `https://www.softbankrobotics.com/emea/en/pepper`

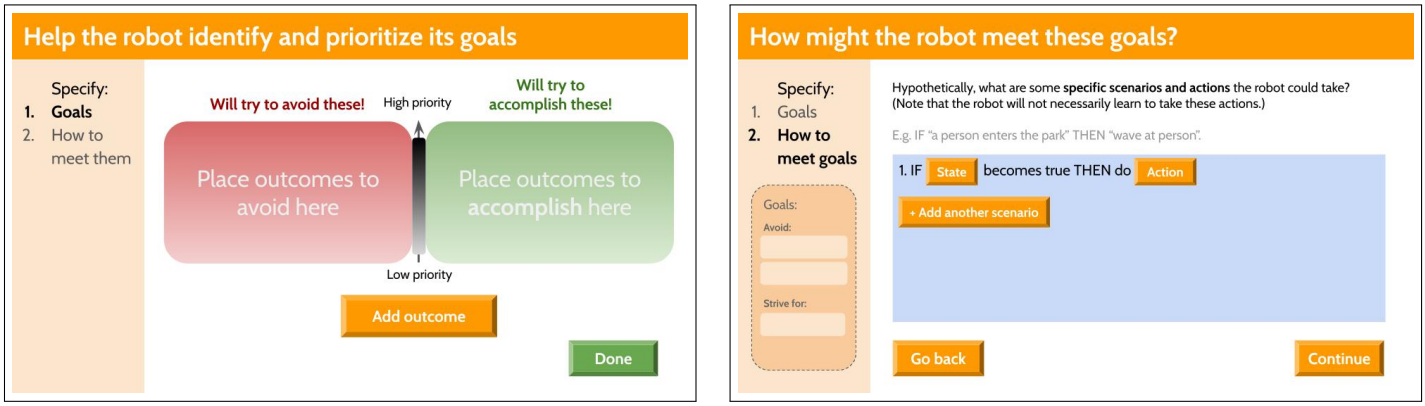

Figure 1: Design sketches showing selected stages of the user interaction with our graphical interface. Left: The user can specify outcomes and assign reward values as part of the MDP. Right: The user can specify states and actions as well as review recommendations from the system.

State Features

- **Person's Location**: their position and distance relative to the robot;
- **Person's Mood**: how happy, engaged, or frustrated they are;
- **Person's Actions**: whether they are looking at the robot or speaking, what they are saying, or other user-specified actions;
- **Miscellaneous**: what the robot is doing, is the environment noisy, is there trash around the robot.

(Robot's) Actions

- **Gaze**: look at a specific person, object, or somewhere else;
- **Speech**: say a specified phrase, stop speaking;
- **Indicate**: motion to people or objects;
- **Navigate**: start/stop traveling a specified distance toward a specified direction at a specified speed;
- **Miscellaneous**: rest, shut down.

Outcomes to Reward

- **Person's Mood**: whether they are happy, engaged, or frustrated;
- **Person's Speech**: whether they refer to the robot positively, whether they say certain phrases;
- **Person's Actions**: whether they acknowledge or ignore the robot; whether they respond as expected (e.g. whether they are following the robot's directions).

Figure 2: A work-in-progress list of interface primitives we plan to support, deduced from reviewing existing tasks studied in HRI literature (e.g. [Sauppé and Mutlu, 2014]). "Person" refers to a single individual interacting with the robot, such as a conversation partner.

The user may even choose to see the robot execute this policy, either in simulation or in a real-world deployment. The user can modify this final policy, or return to the beginning to revise the MDP and repeat the process.

To illustrate a use case of our proposed tool, we walk through the following example. Suppose a robot is deployed at a store and tasked with maximizing average customer satisfaction, as measured by an in-store exit survey. A user who is comfortable with the ideas of MDP might use our interface to specify state features related to the mood of the customers. They might specify actions such as greeting customers and approaching customers to offer assistance. They might consider each customer to be an episode, and the positive or negative survey rating of the customer to be the reward for that episode. A different user might choose to to define the tasks by writing IF-THEN statements such as "IF a customer begins to get frustrated THEN approach the customer to offer assistance," allowing the system to extract a similar MDP as above.

Based on the MDP, the tool might recommend additions to the states, actions, and rewards to account for potential robot abuse. For example, it might recommend the age of the customer as a state feature since children might be more unruly and more likely to hinder the robot, and the ability to run away as a possible action. The user decides to accept these changes, and then train the robot. When training has concluded, suppose the user sees that the final policy has the robot walk away from people intentionally blocking its path. The user wants to know if it will be more effective in raising customer moods and reducing antagonistic encounters if the robot instead asks the customer to politely move, which was not an action included in the original MDP. They can, therefore, return to the starting MDP, modify it to add this new action, and re-train the robot.

3 Development and Evaluation

Our development road map consists of low-fidelity prototyping, an implementation stage, and then a final user study evaluation. We will first hold remote, moderated sessions with participants of various levels of technical expertise to prototype our designs with them. After iterating on the designs based on participants' feedback, we will implement them. We evaluate the implemented system with a user study.

In the user study, participants will use our tool to train a robot for a set of HRI tasks. We may select tasks applicable to either the NAO[2] or the Stretch[3] robots. There may be a baseline condition in which participants use a simplified version of tool, without receiving automatic support in the form of MDP recommendations or policy visualizations. We will ask participants to rate the difficulty of the tasks, the convenience of the tool, and the confidence in their answers. We will administer the System Usability Scale and ask participants to describe their thought process during the experience.

4 Related Work

In this section, we review selected existing work on applying RL to HRI and on RL tools for end users.

4.1 HRI Applications for RL

Hemminghaus and Kopp [2017] explored how Q-learning can help a robot learn when and how to provide assistance to the user without overwhelming them or distracting them from the task at hand. Park et al. [2019] designed a robot for expanding childrens' language skills by using Q-learning to determine the complexity of the stories it tells to children. Ramírez et al. [2016] used inverse RL to train robots to approach humans in a socially appropriate manner. In these works, the authors defined the RL problem for the robot to solve, rather than asking novices to determine the problem, which our work will enable.

4.2 RL Support for End Users

A large body of work has investigated the potential for interactive RL, in which the user is able to adjust the reward function and state features during the agent's learning process [Amershi et al., 2014, Arzate Cruz and Igarashi, 2020]. Thomaz and Breazeal [2006] found that study participants used rewards to provide feedback on the robot's past actions and provide guidance on future actions. Incorporating these reward variants helped RL non-experts efficiently train a virtual robot to successfully bake a cake. In addition, Thomaz et al. [2006] found that study participants preferred giving positive rewards, and often adapted their reward behavior as they became more knowledgeable about the robot's training process. MacGlashan et al. [2017] showed that people construe the reward they deliver as being a critique relative to the agent's recent behavior. Judah et al. [2010] explored means of combining agent exploration with end-user critique after each session to label certain actions as "good" or "bad." These studies largely focus on end-user involvement during the training process. Our work instead enables users to explicitly define the problem being learned, thus allowing them to dictate the purpose of the training and to do so before training starts.

5 Conclusion

A large body of work has examined supporting end users and experts in improving their RL system after it is deployed, yet little has attempted to help at the very beginning of this lifecycle. We propose supporting end users in defining RL problems through structured templates, automatic recommendations, and iterative debugging. We will prototype these designs, implement them, and then evaluate the system by conducting a user study with HRI tasks.

References

Saleema Amershi, Maya Cakmak, William Bradley Knox, and Todd Kulesza. Power to the people: The role of humans in interactive machine learning. *AI Magazine*, 35(4), Dec. 2014. doi: 10.1609/aimag.v35i4.2513.

Sean Andrist, Micheline Ziadee, Halim Boukaram, Bilge Mutlu, and Majd Sakr. Effects of culture on the credibility of robot speech: A comparison between english and arabic. In *Proceedings of the Tenth Annual ACM/IEEE International Conference on Human-Robot Interaction*, HRI '15, 2015. doi: 10.1145/2696454.2696464.

Christian Arzate Cruz and Takeo Igarashi. *A Survey on Interactive Reinforcement Learning: Design Principles and Open Challenges*. 2020.

[2] https://www.softbankrobotics.com/emea/en/nao

[3] https://hello-robot.com/product

Andrew G Barto, Philip S Thomas, and Richard S Sutton. Some recent applications of reinforcement learning. In *Proceedings of the Eighteenth Yale Workshop on Adaptive and Learning Systems*, 2017.

Dražen Brščić, Hiroyuki Kidokoro, Yoshitaka Suehiro, and Takayuki Kanda. Escaping from children's abuse of social robots. In *Proceedings of the Tenth Annual ACM/IEEE International Conference on Human-Robot Interaction*, HRI '15, 2015. doi: 10.1145/2696454.2696468.

Jacqueline Hemminghaus and Stefan Kopp. Towards adaptive social behavior generation for assistive robots using reinforcement learning. In *Proceedings of the 2017 ACM/IEEE International Conference on Human-Robot Interaction*, HRI '17, 2017. doi: 10.1145/2909824.3020217.

Kshitij Judah, S. Roy, Alan Fern, and Thomas G. Dietterich. Reinforcement learning via practice and critique advice. In *AAAI*, 2010.

Jens Kober, J. Andrew Bagnell, and Jan Peters. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 32(11), 2013. doi: 10.1177/0278364913495721.

Min Kyung Lee, Jodi Forlizzi, Sara Kiesler, Paul Rybski, John Antanitis, and Sarun Savetsila. Personalization in hri: A longitudinal field experiment. In *Proceedings of the Seventh Annual ACM/IEEE International Conference on Human-Robot Interaction*, HRI '12, 2012. doi: 10.1145/2157689.2157804.

Daniel Leyzberg, Samuel Spaulding, and Brian Scassellati. Personalizing robot tutors to individuals' learning differences. In *Proceedings of the 2014 ACM/IEEE International Conference on Human-Robot Interaction*, HRI '14, 2014. doi: 10.1145/2559636.2559671.

James MacGlashan, Mark K Ho, Robert Loftin, Bei Peng, Guan Wang, David L. Roberts, Matthew E. Taylor, and Michael L. Littman. Interactive learning from policy-dependent human feedback. In *Proceedings of the Thirty-Fourth International Conference on Machine Learning*, 2017.

Tatsuya Nomura, Takayuki Uratani, Takayuki Kanda, Kazutaka Matsumoto, Hiroyuki Kidokoro, Yoshitaka Suehiro, and Sachie Yamada. Why do children abuse robots? In *Proceedings of the Tenth Annual ACM/IEEE International Conference on Human-Robot Interaction Extended Abstracts*, HRI'15 Extended Abstracts, 2015. doi: 10.1145/2701973.2701977.

Hae Won Park, Ishaan Grover, Samuel Spaulding, Louis Gomez, and Cynthia Breazeal. A model-free affective reinforcement learning approach to personalization of an autonomous social robot companion for early literacy education. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'19/IAAI'19/EAAI'19, 2019. doi: 10.1609/aaai.v33i01.3301687.

E. Pot, J. Monceaux, R. Gelin, and B. Maisonnier. Choregraphe: a graphical tool for humanoid robot programming. In *RO-MAN 2009 - The 18th IEEE International Symposium on Robot and Human Interactive Communication*, 2009. doi: 10.1109/ROMAN.2009.5326209.

Omar A. Islas Ramírez, Harmish Khambhaita, Raja Chatila, Mohamed Chetouani, and Rachid Alami. Robots learning how and where to approach people. In *2016 25th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, 2016. doi: 10.1109/ROMAN.2016.7745154.

P. Salvini, G. Ciaravella, W. Yu, G. Ferri, A. Manzi, B. Mazzolai, C. Laschi, S.R. Oh, and P. Dario. How safe are service robots in urban environments? bullying a robot. In *19th International Symposium in Robot and Human Interactive Communication*, 2010. doi: 10.1109/ROMAN.2010.5654677.

Allison Sauppé and Bilge Mutlu. Robot deictics: How gesture and context shape referential communication. In *Proceedings of the 2014 ACM/IEEE International Conference on Human-Robot Interaction*, HRI '14, 2014. doi: 10.1145/2559636.2559657.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. 1998.

Andrea L. Thomaz and Cynthia Breazeal. Reinforcement learning with human teachers: Evidence of feedback and guidance with implications for learning performance. In *Proceedings of the 21st National Conference on Artificial Intelligence - Volume 1*, AAAI'06, 2006.

Andrea L. Thomaz, Guy Hoffman, and Cynthia Breazeal. Reinforcement learning with human teachers: Understanding how people want to teach robots. In *ROMAN 2006 - The 15th IEEE International Symposium on Robot and Human Interactive Communication*, 2006. doi: 10.1109/ROMAN.2006.314459.

Lin Wang, Pei-Luen Patrick Rau, Vanessa Evers, Benjamin Krisper Robinson, and Pamela Hinds. When in rome: The role of culture & context in adherence to robot recommendations. In *Proceedings of the 5th ACM/IEEE International Conference on Human-Robot Interaction*, HRI '10, 2010.