Neural Computing and Applications (2023) 35:23429–23439
https://doi.org/10.1007/s00521-021-06375-y

S.I.: HUMAN-IN-THE-LOOP MACHINE LEARNING AND ITS APPLICATIONS

Improving reinforcement learning with human assistance: an argument for human subject studies with HIPPO Gym

Matthew E. Taylor[1,3] · Nicholas Nissen[1] · Yuan Wang[1] · Neda Navidi[2]

Received: 3 February 2021 / Accepted: 26 July 2021 / Published online: 19 September 2021

Abstract

Reinforcement learning (RL) is a popular machine learning paradigm for game playing, robotics control, and other sequential decision tasks. However, RL agents often have long learning times with high data requirements because they begin by acting randomly. In order to better learn in complex tasks, we argue that an external teacher can often significantly help the RL agent learn. OpenAI Gym is a common framework for RL research, including a large number of standard environments and agents, making RL research significantly more accessible. This article introduces our new open-source RL framework, the Human Input Parsing Platform for Openai Gym (HIPPO Gym), and the design decisions that went into its creation. The goal of this platform is to facilitate human-RL research, making human-in-the-loop RL more accessible, including learning from demonstrations, learning from feedback, or curriculum learning. In addition, all experiments can be conducted over the internet without any additional software needed on the client's computer, making experiments at scale significantly easier.

Keywords Reinforcement learning · Human-in-the-loop AI · Human-AI interaction · Human subject studies · Crowdsourcing · Open-source software

1 Introduction

Reinforcement learning (RL) is a type of machine learning that lets virtual or physical agents learn through experience, often finding novel solutions to difficult problems and exceeding human performance. RL has had many exciting successes, including video game playing [35], robotics [32], stock market trading [7], and data center optimization [13]. Unfortunately, there are still relatively few real-world, deployed, RL success stories. One reason is that learning a policy can be very sample inefficient (e.g., Open AI Five used 180 years of simulated training data per day via massive parallelism on many servers [40]). One reason for this is that RL has traditionally focused on how agents can learn from the ground up.[1] While such research is

This work has taken place in the Intelligent Robot Learning Laboratory at the University of Alberta, which is supported in part by research grants from the Alberta Machine Intelligence Institute (Amii); CIFAR; a Canada CIFAR AI Chair, Amii; and NSERC.

✉ Matthew E. Taylor
matthew.e.taylor@ualberta.ca

Nicholas Nissen
nnissen@ualberta.ca

Yuan Wang
wang17@ualberta.ca

Neda Navidi
neda.navidi@lassena.etsmtl.ca

[1] Department of Computing Science, University of Alberta, Edmonton, Canada

[2] École de technologie supérieure (ETS), Montreal, Canada

[3] Alberta Machine Intelligence Institute (AMII), University of Alberta, Edmonton, Canada

[1] For example, Sutton and Barto's influential RL textbook [52] does not mention human interaction, transfer learning, etc., eschewing external knowledge sources.

absolutely important, we argue that we need to also better allow RL agents to learn from others, whether programs, agents, or humans.[2] Cheating should be encouraged![3]

Rather than the typical RL setting, in order to reduce the potentially substantial costs of environmental interactions and compute (in terms of time, money, wear and tear on the robot, etc.) and to provide better initial performance, we consider how an RL *student* can receive help from a *teacher* via additional guidance. Our long-term interest in such research is to enable RL to be successfully deployed in more real-world scenarios by focusing exploration and jumpstarting initial behavior to quickly reach high quality policies. For example, our Human-Agent Transfer (HAT) algorithm [55] used 3 min of human guidance to save 7 h of agent learning time in a simulated soccer environment. While such initial successes like HAT are encouraging, many questions must be addressed before these techniques can reliably improve RL performance. Such approaches will allow existing programs and humans to provide guidance to an RL agent, significantly improving RL algorithms so that they can learn better performance faster, relative to (1) learning without external guidance and (2) existing human/agent guidance algorithms.

This article has two goals. First, to highlight open and exciting problems, relative to existing work, and to encourage additional research in this area. Second, to introduce an open-source software platform that enables human-in-the-loop RL experiments to easily scale to hundreds or thousands of users.

Section 2 provides a very brief introduction to reinforcement learning. Section 3 discusses different methods for allowing a student agent to learn how to complete sequential decision tasks with a human or agent teacher. Section 4 motivates why additional research on human-in-the-loop RL is critical, due in part to the large number of open questions. Section 5 describes our novel framework, argues why it is an ideal platform for enabling such experiments, and provides two examples of how the framework could be used. Section 6 concludes (Fig. 1).

[2] RL agents could theoretically treat all input as sensory input, considering it part of the environment. However, it is more practical to program in the ability to leverage advice, rather than requiring the agent to learn the about the special significance, and interpretation, of advice.

[3] Of course, there are also good reasons not to include external information. For instance, it may be much more fruitful to spend time and resources developing better algorithms that can directly benefit from Moore's law and its analog to computational improvements [51]. While this may indeed be a better approach in the long run, we argue that including these kinds of biases can help agents solve difficult RL problems today, without waiting for more powerful algorithms that have yet to be developed.

2 Reinforcement learning background

Reinforcement learning considers the problem of how an agent should act in an environment over time to maximize a reward signal (in expectation). We can formalize the interaction of an agent with its environment as a Markov Decision Process (MDP).

An MDP M is a 5-tuple $(\mathcal{S}, \mathcal{A}, p, r, \gamma)$, where $\mathcal{S}$ is the set of states in the environment, $\mathcal{A}$ is the set of actions the agent can execute, $p(s'|s,a)$ is the transition function that gives the probability of reaching state s' from s after taking action a, $r(s)$ is the reward function that gives the immediate reward for reaching state s, and γ is a (0, 1] discount factor.

At each discrete time step t, the agent uses its current state to select an action according to its *policy* $\pi(s)$. The goal is to learn to approach or reach an optimal policy, $\pi^\star$, which maximizes the expected discounted sum of rewards from now until the end of an episode at time T (or ∞ in the non-episodic case):

$$\mathbb{E}\left[\sum_{i=t}^{T} \gamma^{i-t} r(s_t)\right]$$

There are many ways to try to learn to reach, or approximate, $\pi^\star$, including model-free methods that learn how to act in the environment and model-learning methods that can incorporate planning. One common approach is to not learn π directly, but to instead learn an action-value function that estimates how good a given action will be in some state when following the current policy:

$$q_\pi(s_t, a_t) = \sum_{s_{t+1}, r_t} p(s_{t+1}, r_t | s_t, a_t)\left[r_t + \gamma \max_{a_{t+1}} q_\pi(s_{t+1}, a_{t+1})\right]$$

Eventually, q-values should converge toward $q_{\pi^\star}$, at which point the agent would have learned the optimal policy, $\pi^\star$.

Most real-world domains do not have only tabular states or actions—to learn in continuous states spaces and/or with continuous action spaces, some type of function approximation is necessary. Currently (deep) neural networks are often used. For an introduction to such learning methods, please see [52].

3 Current speedup approaches

There are many existing approaches leveraging knowledge—for instance, even using batch or offline RL can be considered "existing knowledge." For an overview, please see our recent survey [6]. In this article, we specifically focus on a student-teacher framework, where the teacher

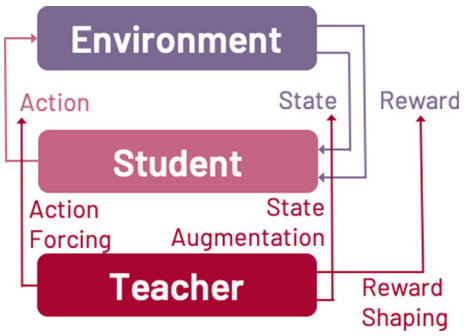

Fig. 1 A student agent can both learn by interacting with the environment, and directly or indirectly from a teacher. Examples of assistance include suggesting or forcing the student to execute an action, adding additional information to the student's state, and creating a more informative reward signal

can be a human, an RL agent, or a program, and the student is an RL agent.

The goals for such an approach can be to

1. allow an RL student to improve its learning performance (relative to learning without guidance from a teacher);
2. ensure that the student's final performance is not harmed by a suboptimal teacher;
3. minimize the cognitive load or stress on the human teacher;
4. minimize the amount of advice needed from the teacher; and
5. make the best use of whatever advice is provided by the teacher.

A teacher could proactively provide advice [53] because it knows better than the student, a student could ask for advice [12] because it knows when it is confused, or a combination of both could happen simultaneously [2, 11]. Furthermore, the advice could be provided up front (e.g., a human records a number of demonstrations for an agent [3]) or could be provided over time (e.g., a demonstrator could provide new labels over time [45]). The guidance could be free or costly, and unlimited or finite.

This section briefly reviews a selection of existing approaches where an RL student can improve its learning performance. Generally, we assume a teacher's goals are aligned with the student and its performance is (at least initially) better than the student (Fig. 2).

3.1 Programmatic teachers

If the teacher is an RL agent, transfer learning [54] can often be used to directly bring the teacher's "brain" (e.g., its q-values) possibly with some adaptation, into the student. But, in many cases this is infeasible because the teacher's knowledge is not directly accessible or is incompatible with the student's knowledge representation.

More flexible methods like action advising allow a teacher to tell the student what action to perform on the current timestep. As long as the teacher can suggest an action, it does not matter whether the agent uses a neural network, a table, a PID controller, or even a carefully hand-coded policy. The advice could be provided if the student is uncertain and it asks the teacher for help [12], the teacher may proactively provide guidance when it thinks the student is about to make a large mistake [53], or a combination of the two [2]. There is typically a fixed teaching budget (e.g., a constraint) or a set teaching cost (e.g., a multi-objective or optimal stopping problem). As long as the student eventually stops receiving advice, the optimal policy is guaranteed not to change [58]. Existing methods often use heuristics to decide when to request/provide guidance, but a teacher could also learn how to teach [16], or agents could both learn when to ask for, and provide, guidance [39, 48] simultaneously.

3.2 Human teachers

In the case of a human teacher, a transfer learning approach that directly uses the teacher's learned policy or q-values, is no longer available. But methods like action advising still apply. Other methods, such as leveraging demonstrations, can be used by programmatic teachers but are even more common with human teachers. For instance, demonstrations can help initialize a neural network [15, 20] that an RL agent uses for representation learning and/or to help learn a policy. Demonstrations could also be used to train a classifier to estimate what action the demonstrator would take in any state [55, 57]. Another type of guidance is human feedback—the participant could give qualitative feedback ("good robot" vs. "bad robot") that the agent could directly learn from [24, 28, 30], or the agent could learn from a combination of this feedback and the environmental reward [8, 25]. To help the student learn, a human could also provide: a curriculum of tasks of increasing difficulty for the agent to help it learn [41], a shaping reward [4, 37], natural language advice [29], salient regions of the state space [18], and advice as statements in logic [31] or code [44].

4 Selected open problems

While the previous section gave a brief overview to existing approaches, there are still many open questions. This section highlights questions related to different types of teachers and related to evaluation.

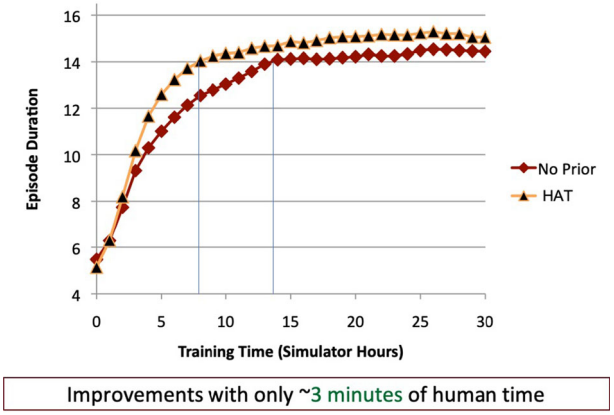

Fig. 2 An agent could learn using reinforcement learning alone, or it could leverage just 3 min of human demonstrations to significantly improve the performance, e.g., the time needed to reach a performance of 14, by using the HAT algorithm [55]

4.1 Teacher-dependent approaches

This section considers open problems that depend on the type of teacher: both human and programmatic teachers, teachers that are agents, and teachers that are human.

4.1.1 Human or programmatic teachers

both can allow an RL student to decide when or where it should ask for advice. For instance, assuming advice is not free or infinite, value of information (VOI) estimates could help determine when an agent should ask for advice. This VOI estimate will not only depend on the student's current performance and uncertainty, but also on the estimated quality of the teacher's advice, and the cost for this advice. It is not clear yet when it is better for the student to request guidance (because it knows what it does not know) or better for the teacher to provide proactive guidance (because it understands the task better). While this research is easier to conduct with programmatic teachers because data are more plentiful, it is ultimately an even more important for a human teacher, because humans will provide relatively less advice.

Existing work (including our own) almost exclusively focuses on demonstrating that a particular method can help improve student learning, rather than understanding where and why different kinds of guidance work best. For instance, a teacher providing an action suggestion may be particularly useful if the agent has a large action space. In contrast, providing reward feedback may be preferable if the teacher is unable to make fast, low-level action decisions and the environmental reward signal is sparse.

While most current work focuses on a single student and a single teacher, other combinations are possible. For instance, if there are multiple teachers, a student could decide which of multiple teachers to query based on their abilities [27]. Or, in contrast, multiple students could have access to a single teacher, and the students could coordinate among themselves to decide which one asks for guidance [9].

4.1.2 Agent teachers

must consider when and where to proactively provide advice. In addition to heuristic and learned methods, there could also be advanced modeling methods. In addition to the teacher estimating the student's policy or the student's performance, a deeper level of understanding could allow it to provide more targeted and impactful advice. In the limit, if the teacher fully knew the environment, the student's prior knowledge, and the student's learning algorithm, it could treat teaching as a planning problem [43] where it figured out the optimal set of advice to provide to find the most efficient learning outcome.

4.1.3 Human teachers

introduce many additional challenges. For instance, we now worry about trying to keep them engaged [26], how they might naturally teach [23] or prefer to teach, or when one method is perceived to be more difficult to teach (e.g., via the NASA TLX [19]).

The participant's background may have a direct impact on the usefulness of their guidance or their comfort providing such guidance. For instance, our prior work [36] found a statistically significant (with large effect size) correlation between gender, task framing, and the participant's self-reported interest in a robotic task. We had also found that there was a correlation between teleoperating a UAV with gaming experience [47]. Other work on robot's learning from demonstration via keyboard interaction [50] showed that non-roboticists interacted significantly differently from roboticists, which affected the performance of some algorithms—we want to make sure our methods work for many different kinds of people, not just researchers in an AI lab! We recommend recruiting and studding interactions with diverse set of participants (e.g., different technology exposure, gender, age, education, video game experience, views on AI, and AI backgrounds).

We should better understand how the participant's proficiency in a task, the participant's understanding of the task, and whether the agent is explainable or interpretable [21, 42] affects the quality of the participant's advice, and the resultant quality of the student's learning.

Many studies on human-in-the-loop RL (again, including some of our own), leverage partially trained agents as a stand-in for human advice. However, it is not clear how such automated teachers differ from actual humans. It is

likely that there are differences such that a method tuned on automated teachers should be modified for human teachers, and vice versa. Similarly, if we were able to create more realistic teaching agents, the difference between automated teachers and human teachers could be reduced.

Other future studies could also consider questions such as:

- How do be best explain to human participants how to provide the most useful feedback for a particular type of learning algorithm?
- How do we understand what type of modality a human participant would prefer to use when teaching, and why?
- Can we account for biases in how people naturally teach to learn better from them?[4]

4.2 Where and how to evaluate

This section considers the difference between evaluation methodologies when the teacher is programmatic or is human.

4.2.1 Evaluation with programmatic teachers

is relatively easy because data are plentiful. A teacher, whether a control method, a hard-coded program, or a RL agent, can run indefinitely in simulation to generate data, helping us understand where and how different kinds of teacher and student approaches do or do not work. Such methods are excellent for producing statistical significance and would apply in real-world settings where such a teacher exists. However, it is not currently clear if and when experiments conducted with programmatic teachers will directly apply to human teachers.

4.2.2 Evaluation with human teachers

is again more difficult. While there are some guides toward how to start testing human-AI [1] or human-ML [33] algorithms, few machine learning researchers have extensive human subject study experience. This can require non-trivial ramp-up times for researchers.

Some types of human subject experiments need very specific hardware, such as for visual emotion recognition [10], gaze tracking [46], or EEG control [22]. However, we believe that many of these types of human-in-the-loop RL experiments can be conducted over the web with a standard computer. We recently created the Human Input Parsing Platform for Openai Gym (HIPPO Gym, Fig. 3) project [38], which has been released as open source.[5] This framework allows easy development and deployment of human subject studies, allowing participants to interact with OpenAI Gym environments like Atari [5] and MuJuCo [56] over the internet. HIPPO Gym currently supports demonstrations, feedback, mouse clicks (e.g., identifying student mistakes [14]), and task speed adjustment. It can run on standalone servers or be easily integrated into Amazon's Web Service and Amazon's Mechanical Turk. We will continue to develop this codebase to allow for additional kinds of interactions and include implemented learning algorithms for different types of guidance. Our hope is that better standardization among human-in-the-loop RL experiments and experimenters would make this research more accessible and replicable.

By designing and running many human subject studies with hundreds of participants, we will better understand what types of guidance are more or less useful. For instance, a teacher providing an action suggestion may be particularly useful if the agent has a large action space, while providing reward feedback may be preferable if the teacher is unable to make fast, low-level decisions. A related goal is to discover general guidelines about when a human teacher would prefer to provide one type of guidance by asking participants to interact with a student in multiple ways. This approach allows us to not only quantitatively measure the impact of student learning, but also to elicit human teacher preferences (e.g., interviews and a 5-point Likert scale questions) and measure if these correlate with perceived task difficulty (e.g., the NASA TLX [19]).

5 HIPPO Gym

We have designed, implemented, tested, and released the HIPPO Gym to meet the *desiderata* discussed in previous sections for conducting experiments with human teachers.

5.1 Design principles

HIPPO Gym is built around three core features: modularity, so that different functions and features can be reused; ease of implementation, so that an RL researcher can quickly get up to speed without a deep understanding of web programming; and inclusion of many examples, so

[4] For example, in our past work [41], we biased our curriculum learning algorithm to better take advantage of a pattern we found in human participants. Similarly, in a game theory setting, we know human participants are not perfectly rational, and leveraging theories like quantal response equilibrium [34] and the anchoring effect [17] can help better predict and understand human behavior.

[5] An overview can also be found at https://hippogym.irll.net/.

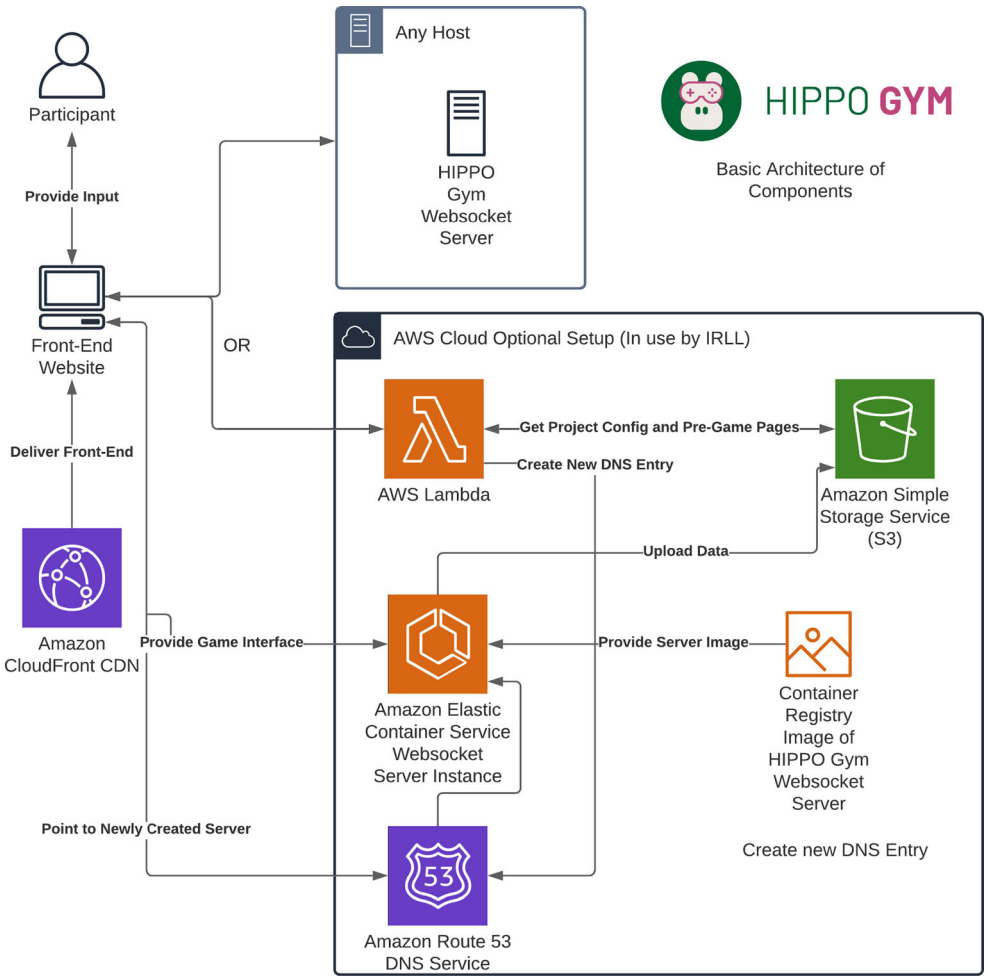

Fig. 3 Overview of the component structure of HIPPO Gym. Note that the AWS Infrastructure is optional and can be replaced by any server setup

that the researcher can begin quickly understanding the benefits and capabilities of the framework without a large up-front time investment.

HIPPO Gym consists of three primary components: a front-end website for collecting human inputs; a websocket server for connecting the website to an agent and environment; and the cloud infrastructure for deploying experiments. Each component is independent and in practice a researcher may only need to change the websocket server code to integrate their agent/environment in order to run an experiment (see Fig. 3).

Ease of use by experimenters (not network engineers) was a critical factor in the development, common languages (python and json) were chosen due to the general familiarity with these tools within the research community. Additionally, the user-facing website is statically deployed and maintained so that any researcher anywhere can use this component with no effort on their part (unless they would prefer to host on their own webserver). Finally, the cloud infrastructure is set up on Amazon Web Services (AWS) so that any researcher with appropriate permissions deploy a research project with only a single command and a configuration file, without requiring significant infrastructure. Instructions and code for using and understanding this infrastructure are available to anyone, so that with a small effort this rapid deployment can be recreated for any group of researchers. However, it is also not mandatory as deployed infrastructure could be as simple as a server sitting on a researcher's desk.

5.2 Code structure

This section outlines the structure of the released HIPPO Gym code.

5.2.1 Front-end website

Once human participants are recruited (via email, Amazon's Mechanical Turk, etc.), they can access the front-end user interface. The IRL Lab hosted version lives at https://

irll.net and is configurable through the use of URL query strings and websocket messages. The functional components of the page (buttons, inputs, and information displays) are determined by the websocket message from the server—a researcher can determine how this web page looks and functions without ever changing its code. Additionally, the project and user IDs, as well as server location, debugging features, and even look and feel via css are controlled via URL query strings. By changing the link to the page, a researcher can control all of these aspects. There is little need for a researcher to do any browser-specific coding.

Examples of the front end for Breakout and Lunar Lander are in Fig. 4 and Ms. Pac-Man is shown in Fig. 5.

5.2.2 Back-end websocket server

This layer provides communication between the user interface (UI) in the browser and the agent/environment. At its core, this component sends UI feature instructions to the front end, along with the frame image for the current step in the environment, and then receives and parses input messages from the browser to provide the relevant information to the experimenter's code. There are also functions for saving data and (if appropriate) uploading that data into cloud storage for later use.

The back end is written in python and uses json for websocket messaging. This combination is not the most efficient and in fact limits concurrent users on a single server. However, based on the nature of our targeted research projects, these limitations are far outweighed by the familiarity of researchers with these technologies allowing for much faster development and iteration of experiments.

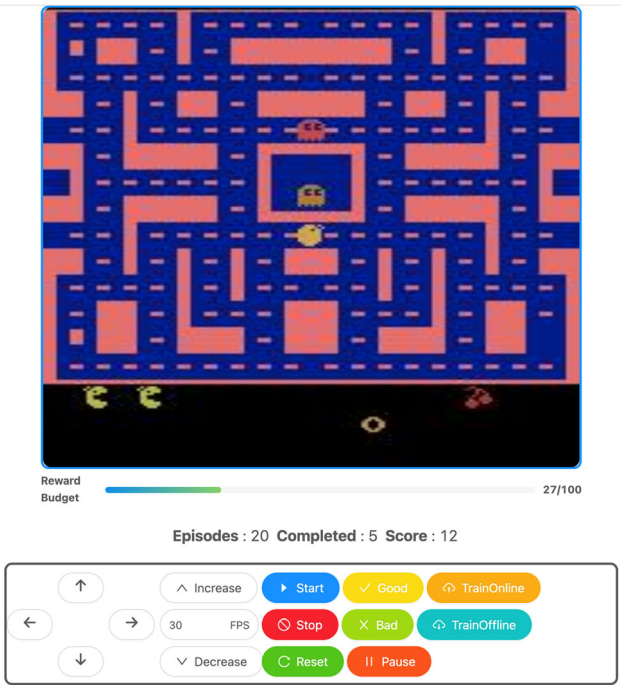

Fig. 5 This screenshot shows the training Ms. Pac-Man in HIPPO Gym by using a set number of good/bad feedbacks and demonstrations. Note additional options, which can be easily (de)activated, include increase speed, decrease speed, start, stop, pause, reset, train offline, and train online

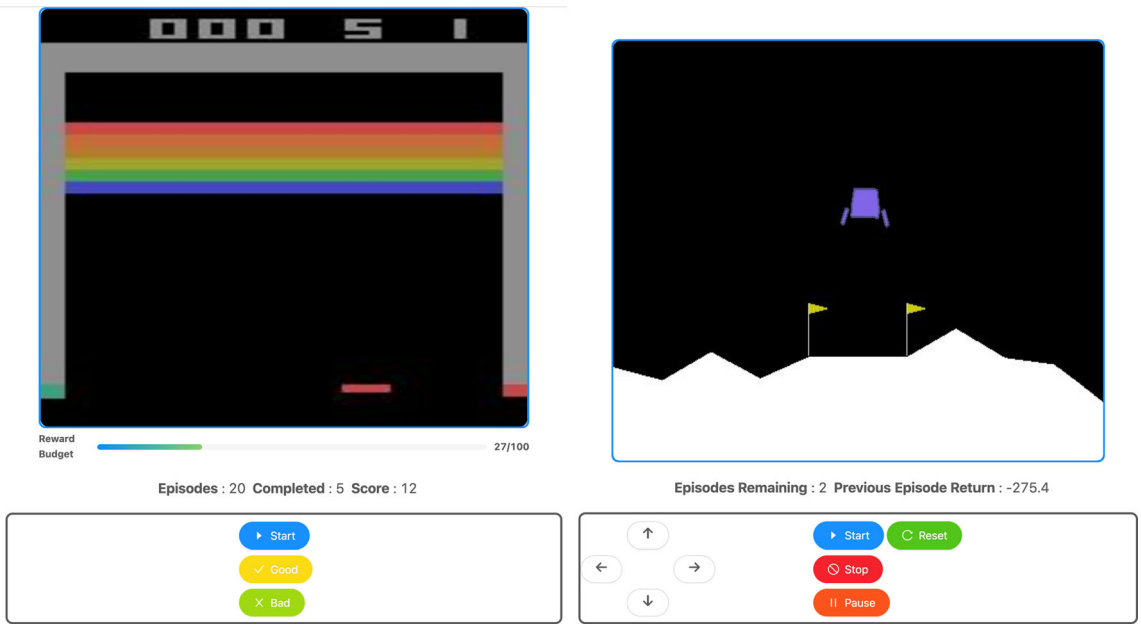

Fig. 4 In HIPPO Gym, breakout could be trained with positive and negative human feedback (left) and Lunar Lander could be trained by giving demonstrations (right)

5.2.3 Infrastructure

Infrastructure for any complex project distributed over the web can become quite complex. Given that researcher's time is best spent doing research and not managing infrastructure, we aimed to remove as much infrastructure work as possible. The front-end component is statically deployed on a content delivery network (CDN) and managed by the IRL Lab. Any researcher may use this deployed version for their experiments, removing an entire piece of infrastructure from their process. The front-end code is also open sourced and available through GitHub should anyone want to host their own version or should the IRL Lab hosted version become unavailable.

Server infrastructure becomes more difficult to abstract away. At the base level, HIPPO Gym can be run on any machine, optionally via containerization, by passing either the IP address or the domain name of the machine via query string to the front end. This is especially useful during development and testing of an experiment, and may work in production for small-scale experiments, or experiments that require a very specific hardware configuration. In general, however, this is not the best setup for production deployment of an experiment. Therefore, we created a distributed system of AWS services to provide all of the necessary infrastructure (DNS, on-demand containerized servers, long-term data storage, SSL certificates, etc.) to researchers with a simple configuration file and a single command. This allows our group to deploy and update experiments without touching any infrastructure while still benefiting from all the advantages of AWS cloud services. Should another group or individual wish to replicate the setup for themselves, all the required code and setup instructions are available via GitHub.

Functionally, the infrastructure works as such: the front end is statically hosted with the CloudFront CDN. When a user lands on the page, information is read from the query string of the link that brought them to the page and the correct project is loaded, or the 'game' page starts immediately pointing to the given server address. A loaded project may have defined pre-game pages including questionnaires and consent forms. During the pre-game phase the AWS Elastic Container Service is used to start a server for this individual user, the servers are all ephemeral, created when there is a user and destroyed after the user session is complete in order to substantially reduce costs. In order to support SSL, a DNS entry is made in Route53 (the AWS DNS Service), which will point to the just-created server. The front end will then connect to the new server via websocket. The server will pass information about the required UI inputs and they will be loaded along with the first frame of the game. The user then participates in the game in the manner intended by the researcher. When the trial completes the user is shown the next page, often a thank you page, but possibly a redirect elsewhere, further instructions, or even another game page. Once the server is no longer required, the saved data from the trial is uploaded to S3 (AWS Simple Storage Service), which provides an easily accessible long-term file storage, and the server is then destroyed. If a user abandons a trial before completing an experiment, the server has a fail-safe timeout, after which it will be destroyed. This system is extremely cost effective because the infrastructure only has cost when there is a live participant—there is no substantial idle time.

5.3 Current Abilities

Currently, HIPPO Gym works well with discrete action space environments from OpenAI Gym including Classic Control and Box2D environments. Continuous action space environments are compatible but require some additional configuration on the part of the researcher.

Available human inputs (all optional) include: all directions, fire, start, pause, stop (end trial), reset (end episode), increase/decrease frame rate, positive feedback (good), negative feedback (bad). It is also possible for a researcher to define other inputs for which they wish to define an action. Users are also able to click on the game window, recording the x and y coordinates, which may be used for identifying points of interest or identifying errors [14].

Feedback or information can be passed to the user. This information will typically include a score and progress information, but a budget bar is also available that shows how many points of feedback have been given out of a set maximum.

An optional debug mode facilitates development by showing all of the websocket messages in real time.

The base repository for HIPPO Gym includes example integration of both TAMER [24] and COACH [30] algorithms using tile coding for the Mountain Car environment.

5.4 Example results: learning from demonstration

Smart and Kaelbling [49] showed how human demonstrations can be used to bootstrap learning. In the first phase of the two-phase approach, the human controls an agent while the agent performs a single Q-learning update for each $\langle s, a, r, s\prime \rangle$ tuple. In phase 2, the RL agent learning on its own using Q-learning. One of the authors used HIPPO Gym to play Lunarlander-v2 in OpenAI Gym for 120 episodes, and then allowed the agent to learn on its own using DQN. During both phases, we record the current policy and then test it after learning.

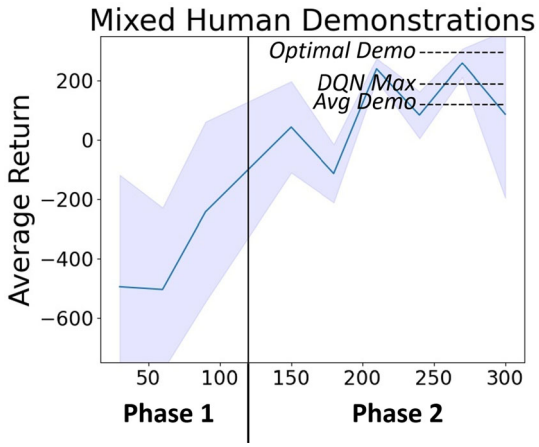

Fig. 6 Human demonstrations are provided in phase 1 and then the agent learns autonomously in phase 2. Although the performance of the demonstrations is far from optimal, the deep Q-learning agent is able to use these demonstrations to quickly achieve a near-optimal performance in less than half the time of an agent without such demonstrations

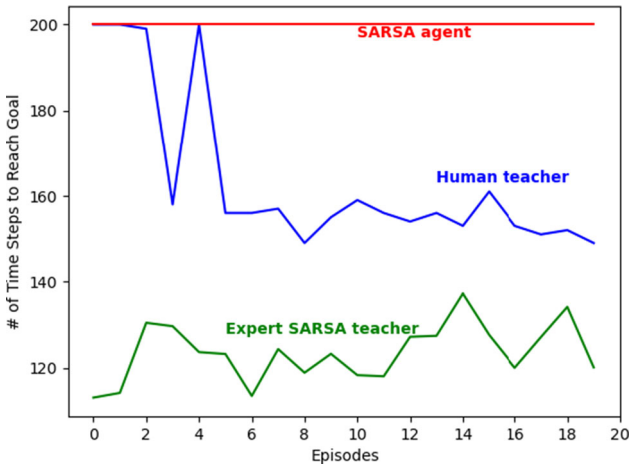

Fig. 7 A member of the IRL lab used positive and negative feedback to train an agent to learn lunar lander with no environmental reward

Figure 6 shows results using a mix of human demonstrations, where the human sometimes successfully navigates the lander to the goal state, but sometimes fails, and almost never optimally. The shaded area represents the standard deviation learning over 10 independent trials on the same demonstration data. DQN reaches an average performance of 187 after 300 independent learning trials, where as these results show learning from demonstration significantly reduce the time needed by the agent to learn independently, similar to Smart and Kaelbling.

5.5 Example results: learning from human feedback

The TAMER algorithm [24] can also be used for an agent to learn in Lunar Lander. In this case, we remove the environmental reward and the agent may only learn to try to maximize human feedback. On every timestep, the human can provide no feedback, positive feedback (good), or negative feedback (bad).

Figure 7 shows TAMER updating weights in a neural network to try to maximize the human's feedback. The human can provide feedback for the first 5 episodes, after which the agent must perform on its own. While the agent performs lower than 200 after the human leaves, it is still quite good compared to training autonomously for only 20 episodes (where an autonomously learning agent would typically have a reward of roughly -200). For comparison, a trained Sarsa agent could also give feedback to the TAMER agent at random intervals, but for the same total amount of feedback as the human. In this case, the TAMER agent does not see a significant drop when the Sarsa agent stops providing feedback to the TAMER agent. However, final performance is not as high when using an agent as a teacher, suggesting that the human teacher is able to provide feedback at more important times in the TAMER agent's training.

5.6 Future enhancements

In the future, in addition to increased documentation and inevitable bug fixes, we plan the following enhancements:

- Implement more OpenAI Gym compatible environments such as Mar.io and Minecraft-type environments.
- Allow the user to fast forward and rewind in order to provide feedback.
- Provide code supporting additional algorithms that can learn from human teachers (with or without an additional environmental reward) out of the box.
- Introduce more generalized agent examples that are not specific to any particular function approximator.

6 Conclusion

This article has summarized some current approaches that allow RL students to learn from human or agent teachers, as well as important open questions. It also introduced HIPPO Gym, an open-source framework for human subject studies in human-RL collaboration. While this article does not provide novel scientific hypotheses or contribute new algorithms, we hope that this collection of open questions

and this new tool can significantly improve our community to conduct such research. For example, there have been multiple studies where a "human" teacher is simply a fixed agent, but we argue that "real human teachers" are more realistic to many final use cases, and may provide different insights than agent teachers. Our hope is that by outlining and motivating these questions, and providing a way for researchers to more quickly conduct human experiments, this article will enable a new generation of researchers to begin studying these important questions at the intersection of machine learning and human subjects.

Acknowledgements All authors contributed to writing this article. Taylor conceptualized and directed the project, secured funding, and was the primary author on this paper. Nissen drove back-end code development, testing, and documentation; lead the integration with Amazon Web Services; and took lead on user testing. Wang drove the front-end code development, testing, and documentation. Navidi provided the initial approach for interacting with OpenAI agents over a website, as well as assisted with code reviews. We appreciate help and feedback from other students in the IRL Lab, including Calarina Muslimani, Rohan Nuttall, and Volodymyr Tkachuk.

Declarations

Conflicts of interest The authors have no conflicts of interest to declare that are relevant to the content of this article.

References

1. Amershi S, Weld D, Vorvoreanu M, Fourney A, Nushi B, Collisson P, Suh J, Iqbal S, Bennett PN, Inkpen K, Teevan J, Kikin-Gil R, Horvitz E (2019) Guidelines for human-AI interaction. In: Proceedings of the 2019 CHI conference on human factors in computing systems, CHI'19, pp 1–13. Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/3290605.3300233
2. Amir O, Kamar E, Kolobov A, Grosz B (2016) Interactive teaching strategies for agent training. In: Proceedings of international joint conference on artificial intelligence (IJCAI)
3. Argall BD, Chernova S, Veloso M, Browning B (2009) A survey of robot learning from demonstration. Robot Autonom Syst 57(5):469–483. https://doi.org/10.1016/j.robot.2008.10.024
4. Behboudian P, Satsangi Y, Taylor ME, Harutyunyan A, Bowling M (2020) Useful policy invariant shaping from arbitrary advice. In: Proceedings of the adaptive and learning agents workshop at the AAMAS-20 conference
5. Bellemare MG, Naddaf Y, Veness J, Bowling M (2013) The arcade learning environment: an evaluation platform for general agents. J Artif Intell Res 47(1):253–279
6. Bignold A, Cruz F, Taylor ME, Brys T, Dazeley R, Vamplew P, Foale C (2020) A conceptual framework for externally-influenced agents: an assisted reinforcement learning review. arXiv preprint 2007.01544
7. Burhani H, Ding GW, Hernandez-Leal P, Prince S, Shi D, Szeto S (2020) Aiden—reinforcement learning for order execution. https://www.borealisai.com/en/blog/aiden-reinforcement-learning-for-order-execution/. Accessed 1 Feb 2021
8. Cederborg T, Grover I, Isbell C, Thomaz A (2015) Policy shaping with human teachers. In: International joint conference on artificial intelligence (IJCAI)
9. Chernova S, Veloso MM (2010) Confidence-based multi-robot learning from demonstration. Int J Soc Robot 2(2):195–215. https://doi.org/10.1007/s12369-010-0060-0
10. Cui Y, Zhang Q, Allievi A, Stone P, Niekum S, Knox WB (2020) The empathic framework for task learning from implicit human feedback. In: arXiv 2009.13649
11. Da Silva FL, Warnell G, Costa AHR, Stone P (2020) Agents teaching agents: a survey on inter-agent transfer learning. AAMAS 34(9)
12. Da Silva FL, Hernandez-Leal P, Kartal B, Taylor ME (2020) Uncertainty-aware action advising for deep reinforcement learning agents. In: Proceedings of AAAI conference on artificial intelligence
13. DeepMind AI reduces Google data centre cooling bill by 40%. https://deepmind.com/blog/article/deepmind-ai-reduces-google-data-centre-cooling-bill-40 (2016). Accessed 1 Oct 2020
14. de la Cruz Jr GV, Peng B, Lasecki WS, Taylor ME (2015) Towards integrating real-time crowd advice with reinforcement learning. In: The 20th ACM conference on intelligent user interfaces (IUI). https://doi.org/10.1145/2732158.2732180
15. de la Cruz Jr GV, Du Y, Taylor ME (2019) Pre-training with non-expert human demonstration for deep reinforcement learning. Knowl Eng Rev 34. https://doi.org/10.1017/S0269888919000055
16. Fachantidis A, Taylor M, Vlahavas I (2017) Learning to teach reinforcement learning agents. Mach Learn Knowl Extract 1. https://doi.org/10.3390/make1010002
17. Furnham A, Boo H (2011) A literature review of the anchoring effect. J Socio Econ 40:35–42. https://doi.org/10.1016/j.socec.2010.10.008
18. Guan L, Verma M, Guo S, Zhang R, Kambhampati S (2020) Explanation augmented feedback in human-in-the-loop reinforcement learning. In: arXiv 2006.14804
19. Hart SG, Staveland LE (1988) Development of NASA-TLX (task load index): results of empirical and theoretical research. Hum Ment Workl 1(3):139–183
20. Hester T, Vecerík M, Pietquin O, Lanctot M, Schaul T, Piot B, Horgan D, Quan J, Sendonaris A, Osband I, Dulac-Arnold G, Agapiou JP, Leibo JZ, Gruslys A (2018) Deep Q-learning from demonstrations. In: Proceedings of AAAI conference on artificial intelligence
21. Heuillet A, Couthouis F, Díaz-Rodríguez N (2020) Explainability in deep reinforcement learning. In: arXiv 2008.06693
22. Iturrate I, Montesano L, Minguez J (2010) Robot reinforcement learning using EEG-based reward signals. In: 2010 IEEE international conference on robotics and automation, pp 4822–4829. https://doi.org/10.1109/ROBOT.2010.5509734
23. Knox WB, Glass BD, Love BC, Maddox WT, Stone P (2012) How humans teach agents: a new experimental perspective. Int J Soc Robot 4:409–421. https://doi.org/10.1007/s12369-012-0163-x
24. Knox WB, Stone P (2009) Interactively shaping agents via human reinforcement: the TAMER framework. In: Proceedings of the international conference on knowledge capture (KCap)
25. Knox WB, Stone P (2012) Reinforcement learning from simultaneous human and MDP reward. In: Proceedings of the international conference on autonomous agents and multi-agent systems (AAMAS)
26. Li G, Hung H, Whiteson S, Knox WB (2013) Using informative behavior to increase engagement in the tamer framework. In: Gini ML, Shehory O, Ito T, Jonker CM (eds) International conference on autonomous agents and multi-agent systems, AAMAS'13, Saint Paul, MN, USA, May 6-10, 2013, pp 909–916. IFAAMAS. http://dl.acm.org/citation.cfm?id=2485064
27. Li M, Wei Y, Kudenko D (2019) Two-level q-learning: learning from conflict demonstrations. Knowl Eng Rev 34:e14. https://doi.org/10.1017/S0269888919000092

28. Loftin R, Peng B, MacGlashan J, Littman ML, Taylor ME, Huang J, Roberts DL (2015) Learning behaviors via human-delivered discrete feedback: modeling implicit feedback strategies to speed up learning. J Autonom Agents Multi Agent Syst, pp 1–30. https://doi.org/10.1007/s10458-015-9283-7
29. Luketina J, Nardelli N, Farquhar G, Foerster JN, Andreas J, Grefenstette E, Whiteson S, Rocktäschel T (2019) A survey of reinforcement learning informed by natural language. In: Proceedings of the international joint conference on artificial intelligence (IJCAI)
30. MacGlashan J, Ho M, Loftin R, Peng B, Wang G, Roberts DL, Taylor ME, Littman ML (2017) Interactive learning from policy-dependent human feedback. In: Proceedings of ICML
31. Maclin R, Shavlik J (1996) Creating advice-taking reinforcement learners. Mach Learn 22:251–281. https://doi.org/10.1023/A:1018020625251
32. Mahmood AR, Korenkevych D, Vasan G, Ma W, Bergstra J (2018) Benchmarking reinforcement learning algorithms on real-world robots. In: 2nd annual conference on robot learning (CoRL)
33. Mathewson KW (2019) A human-centered approach to interactive machine learning. In: arXiv 1905.06289
34. McKelvey RD, Palfrey TR (1995) Quantal response equilibria for normal form games. Games Econ Behav 10(1):6–38. https://doi.org/10.1006/game.1995.1023
35. Mnih V, Kavukcuoglu K, Silver D, Rusu AA, Veness J, Bellemare MG, Graves A, Riedmiller M, Fidjeland AK, Ostrovski G, Petersen S, Beattie C, Sadik A, Antonoglou I, King H, Kumaran D, Wierstra D, Legg S, Hassabis D (2015) Human-level control through deep reinforcement learning. Nature 518(7540):529–533. https://doi.org/10.1038/nature14236
36. Morton S, Kmec J, Taylor, ME (2019) It's what you call it: gendered framing and women's and men's interest in a robotics instruction task. Int J Gender Sci Technol 11(2)
37. Ng AY, Harada D, Russell S (1999) Policy invariance under reward transformations: theory and application to reward shaping. In: Proceedings of the international conference on machine learning (ICML)
38. Nissen N, Wang Y, Navi N, Taylor ME (2020) Human input parsing platform for openai gym (HIPPO Gym). https://github.com/IRLL/HIPPO_Gym
39. Omidshafiei S, Kim D, Liu M, Tesauro G, Riemer M, Amato C, Campbell M, How JP (2019) Learning to teach in cooperative multiagent reinforcement learning. In: Proceedings of the AAAI conference on artificial intelligence
40. Open AI Five. https://blog.openai.com/openai-five (2018). Accessed 7 Sept 2018
41. Peng B, MacGlashan J, Loftin R, Littman ML, Roberts DL, Taylor ME (2018) Curriculum design for machine learners in sequential decision tasks. IEEE Trans Emerg Top Comput Intell 2:268–277. https://doi.org/10.1109/TETCI.2018.2829980
42. Puiutta E, Veith EM (2020) Explainable reinforcement learning: a survey. In: arXiv 2005.06247
43. Rabinovich Z, Dufton L, Larson K, Jennings N (2010) Cultivating desired behaviour: policy teaching via environment-dynamics tweaks. In: The 9th international conference on autonomous agents and multiagent systems, Toronto, Canada, pp 1097–1104
44. Rosenfeld A, Cohen M, Taylor ME, Kraus S (2018) Leveraging human knowledge in tabular reinforcement learning: a study of human subjects. Knowl Eng Rev 33. https://doi.org/10.1017/S0269888918000206
45. Ross S, Gordon GJ, Bagnell D (2011) A reduction of imitation learning and structured prediction to no-regret online learning. In: Gordon GJ, Dunson DB, Dudík M (eds) Proceedings of the fourteenth international conference on artificial intelligence and statistics, AISTATS 2011, Fort Lauderdale, USA, April 11–13, 2011, JMLR proceedings, vol 15, pp 627–635. JMLR.org. http://proceedings.mlr.press/v15/ross11a/ross11a.pdf
46. Saran A, Zhang R, Short ES, Niekum S (2020) Efficiently guiding imitation learning algorithms with human gaze. In: arXiv 2002.12500
47. Scott M, Peng B, Chili M, Nigam T, Pascual F, Matuszek C, Taylor ME (2015) On the ability to provide demonstrations on a UAS: observing 90 untrained participants abusing a flying robot. In: Proceedings of the AAAI fall symposium on artificial intelligence and human–robot interaction (AI-HRI)
48. Silva FLD, Warnell G, Costa AHR, Stone P (2019) Agents teaching agents: a survey on inter-agent transfer learning. Autonom Agents Multi Agent Syst
49. Smart WD, Kaelbling LP (2002) Effective reinforcement learning for mobile robots. In: Proceedings of the 2002 IEEE international conference on robotics and automation, ICRA 2002, May 11–15, 2002, Washington, DC, USA. IEEE, pp 3404–3410. https://doi.org/10.1109/ROBOT.2002.1014237
50. Suay HB, Toris R, Chernova S (2012) A practical comparison of three robot learning from demonstration algorithm. Int J Soc Robot 4(4):319–330. https://doi.org/10.1007/s12369-012-0158-7
51. Sutton RS (2019) The bitter lesson. http://www.incompleteideas.net/IncIdeas/BitterLesson.html. Accessed 1 Feb 2021
52. Sutton RS, Barto AG (2018) Reinforcement learning: an introduction. MIT Press, Cambridge
53. Taylor ME, Carboni N, Fachantidis A, Vlahavas I, Torrey L (2014) Reinforcement learning agents providing advice in complex video games. Connect Sci 26(1):45–63. https://doi.org/10.1080/09540091.2014.885279
54. Taylor ME, Stone P (2009) Transfer learning for reinforcement learning domains: a survey. J Mach Learn Res 10(1):1633–1685
55. Taylor ME, Suay HB, Chernova S (2011) Integrating reinforcement learning with human demonstrations of varying ability. In: Proceedings of the international conference on autonomous agents and mulit agent systems (AAMAS)
56. Todorov E, Erez T, Tassa Y (2012) Mujoco: a physics engine for model-based control. In: Proceedings of the international conference on intelligent robots and systems (IROS)
57. Wang Z, Taylor ME (2017) Improving reinforcement learning with confidence-based demonstrations. In: Proceedings of the 26th international conference on artificial intelligence (IJCAI)
58. Zhan Y, Bou Ammar H, Taylor ME (2016) Theoretically-grounded policy advice from multiple teachers in reinforcement learning settings with applications to negative transfer. In: Proceedings of the 25th international conference on artificial intelligence (IJCAI)