NiceWebRL: a Python library for human subject experiments with reinforcement learning environments

Anonymous Author(s)
Affiliation
Address
email

Abstract

We present NiceWebRL, a research tool that enables researchers to use machine reinforcement learning (RL) environments for online human subject experiments. NiceWebRL is a Python library that allows any Jax-based environment to be transformed into an online interface, supporting both single-agent and multi-agent environments. As such, NiceWebRL enables AI researchers to compare their algorithms to human performance, cognitive scientists to test ML algorithms as theories for human cognition, and multi-agent researchers to develop algorithms for human-AI collaboration. We showcase NiceWebRL with 3 case studies that demonstrate its potential to help develop Human-like AI, Human-compatible AI, and Human-assistive AI. In the first case study (Human-like AI), NiceWebRL enables the development of a novel RL model of cognition. Here, NiceWebRL facilitates testing this model against human participants in both a grid world and Craftax, a 2D Minecraft domain. In our second case study (Human-compatible AI), NiceWebRL enables the development of a novel multi-agent RL algorithm that can generalize to human partners in the Overcooked domain. Finally, in our third case study (Human-assistive AI), we show how NiceWebRL can allow researchers to study how an LLM can assist humans on complex tasks in XLand-Minigrid, an environment with millions of hierarchical tasks. The library is available at https://anonymous.4open.science/r/nicewebrl-28BF.

1 Introduction

The last 20 years have seen a whirlwind of progress in Machine Learning (ML). Reinforcement Learning (RL) agents have achieved superhuman performance on complex games such as Go [55]; computer vision systems can now process complex scenes [64, 36, 49]; and large language models (LLMs) increasingly act as our coding assistants and thought partners [2, 9].

This progress motivates many researchers to study modern Artificial Intelligence (AI) agents in the context of human behavior. Some ML researchers aim to improve AI systems by comparing them to humans, since humans can still provide an upper bound on the performance our systems can hope to achieve [45, 56]. For example, Minecraft [25, 27] remains a challenging exploration problem for machines but a fun exploration adventure for people [31, 19]. In cognitive science, there is increased interest in asking whether these machines are human-like [39, 63, 12]. Even if they are not, cognitive scientists are interested in using them as the basis for building human-like machines [39, 14]. In multi-agent RL, many researchers are interested in whether these agents can act as adaptive partners to humans across the wide range of social settings they might be deployed in [11, 50]. This is increasingly relevant with LLMs, as they possess superhuman knowledge and are improving in their "reasoning" abilities [23]. A natural question is how well they can combine their prior knowledge with environment perception to assist us in completing complex tasks. Collectively, these advances could have a wide impact—ranging from robotics [58] to education [32] to healthcare [54]. Clearly, many are interested in building human-like, human-compatible, and human-assistive AI.

Submitted to 39th Conference on Neural Information Processing Systems (NeurIPS 2025). Do not distribute.

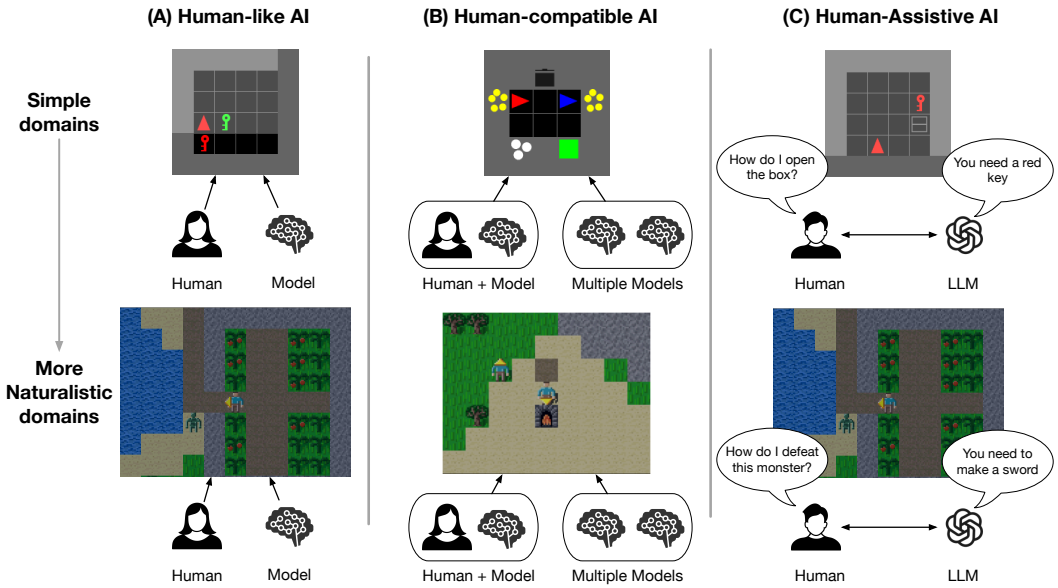

Figure 1: **NiceWebRL is a meta-environment that enables the use of Jax-based environments to develop Human-like, Human-compatible, and Human-assistive AI.** (A) Researchers can compare how humans and AI complete tasks to evaluate if AI behaves in human-like ways. (B) They can study how AI coordinates with humans during task completion to assess if the AI has learned human-compatible social behaviors. (C) They can also integrate Large Language Models into their experiments to evaluate how effectively they combine prior knowledge with environmental perception to assist participants. Importantly, NiceWebRL enables findings to generalize across potentially more complex domains.

Despite this interest in human-centered AI development, pursuing research that integrates human subject experiments with modern ML libraries is currently a cumbersome process. To run experiments with many participants, researchers leverage the internet to get large sample sizes [24]. Thus, most infrastructure is written in the web's programming language: JavaScript [20, 30, 24, 16]. Machine learning code, on the other hand, relies heavily on Python for model development [1, 47, 8] and variants of C for developing fast environments for simulation [37, 60]. Leveraging ML models or environments for human subject experiments currently requires setting up domain-specific server-client configurations that integrate Javascript, Python, and sometimes C. Doing this for each domain makes the process even more cumbersome.

To address this challenge, we present NiceWebRL: a research tool that lets researchers leverage ML environments for human subject experiments (see Figure 1). Integrating Python with JavaScript requires maintaining a connection between a remote Python-based server and a local Javascript-based client. This distance can cause latency issues when running online experiments. To circumvent this challenge, NiceWebRL exploits Jax [8]—a high-performance numerical computing library—to precompute and cache environment dynamics for arbitrary Jax-based environments. NiceWebRL then acts as a meta-environment for researchers to use arbitrary Jax-based environments in their human subject experiments. Critically, NiceWebRL allows researchers to program experiments entirely in Python by integrating with NiceGUI[1]—a library that enables web developers to specify advanced Graphical User Interface (GUI) components entirely in Python.

Our contributions are as follows. (1) We present NiceWebRL, a research tool that enables the use of Jax-based virtual environments for both developing artificial agents and for running human subject experiments (§4). (2) We present 3 case studies that demonstrate how NiceWebRL can support the development of Human-like AI (§5.1), Human-compatible AI (§5.2), and Human-assistive AI (§5.3). (3) Our codebase, `https://anonymous.4open.science/r/nicewebrl-28BF`, comes with several functional example folders using NiceWebRL across these 3 scenarios.

[1] `https://github.com/zauberzeug/nicegui/`

2 Related Work

NiceWebRL is a meta-environment for leveraging Jax-based virtual environments in online human subject experiments. It facilitates the development of Human-like AI, Human-compatible AI, and Human-assistive AI. There is a rich literature on these topics. Researchers have created desiderata for measuring how "Human-like" AI agents are [63, 63]; they have developed benchmarks that test for "human-like" abilities [65, 52]; and books [50] and articles [15] have been written about how to enable human-compatible AI. There is a rich literature on how general AIs can assist humans [26, 53] and a growing literature on how LLMs can assist humans on tasks [42, 61]. Our focus is on enabling researchers to run human subject experiments with modern ML models and environments. Below we review the most relevant literature.

Running human subject experiments. JavaScript is the only programming language that can be run on modern web browsers. As a consequence, most tools for human subject experiments target JavaScript-based development. For example, Labvanced [20], lab.js [30], and Psiturk [24] all provide GUI interfaces for designing JavaScript-based web experiments. While there are Python-based tools for developing human subject experiments like Psychopy [48], they only permit local experiments and an accompanying JavaScript library must be used for writing web experiments. JsPsych [16] is a JavaScript library that facilitates programmatic definitions of experiments like NiceWebRL. While each are useful, none provide utility for leveraging modern ML models and environments (written in Python) for online human subject experiments. This is precisely the gap that NiceWebRL aims to fill.

Comparing natural intelligence and artificial intelligence in virtual environments. Many of the efforts that compare natural and artificial intelligence rely on the Unity game engine, which allows for the development of 3D environments with realistic physics and sensory observations [60]. Cobel-RL developed a 3D environment for studying neuroscience questions around spatial navigation with Deep learning based RL models (i.e. Deep RL models) [18]. The Animal-AI environment was developed to study whether Deep RL models displayed cognitive abilities associated with animals [5]. Most similar to NiceWebRL is PsychLab which explicitly aims to compare Humans to Deep RL models across several classic experimental paradigms like visual search, multiple object tracking, and random dot motion discrimination [40]. However, the following challenges its adoption. (1) Its API is in Lua, which limits accessibility, whereas NiceWebRL is purely in Python, (2) it lacks precise response time measurements which NiceWebRL provides, (3) its reliance on Unity makes it challenging to run online experiments. While NiceWebRL relies on Python, it is able to run online but maintain fast environment performance thanks to its reliance on Jax.

3 Background

The Jax ecosystem. Jax [8] is a Python library for high-performance numerical computing and machine learning with NumPy-like syntax. It follows a functional programming paradigm where functions are stateless, only defining input transformations. It achieves fast computation through JIT compilation and includes tools for tracking random number generators, enhancing reproducibility. Most relevant to NiceWebRL, there is a growing set of environments including XLand-Minigrid, which has millions of hierarchical tasks for studying long-horizon generalization [46]; JaxMARL, which has multi-agent environments for studying coordination [51]; Pgx, which has planning environments for studying reasoning [38]; Craftax, a large procedurally-generated open-world environment that enables studying exploration and generalization [43]; Kinetix, a procedurally-generated physics-based environment for studying physical reasoning [44]; and JaxGCRL, a goal-conditioned robotics environment for studying 3D manipulation tasks [7]. NiceWebRL can be used with all of these environments.

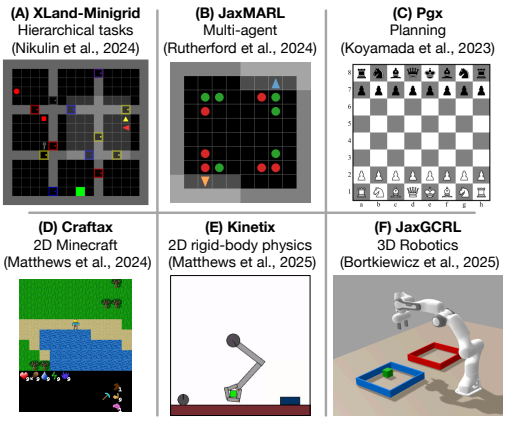

Figure 2: **Examples of Jax-based environments in the Jax ecosystem.** All of these can be leveraged with NiceWebRL to develop Human-like, Human-compatible, or Human-assistive AI.

Python-based web development requires maintaining a Python-based web server that communicates with clients that operate in JavaScript. We build on NiceGUI which comes with tools for handling many concurrent client connections asynchronously. When sending large data packets between a client and a server, web socket connections are needed for real-time communication. NiceGUI's web socket implementation facilitates setting up persistent connections by having web sockets automatically reopen when connections close unexpectedly. This is key to having seamless human experiments with Python-based environment backends. Finally, and equally important for online experiments, NiceGUI enables researchers to use Python to build responsive GUI components without any JavaScript knowledge. We provide examples of GUI components provided by NiceGUI in Figure 3. All of these can be used with NiceWebRL.

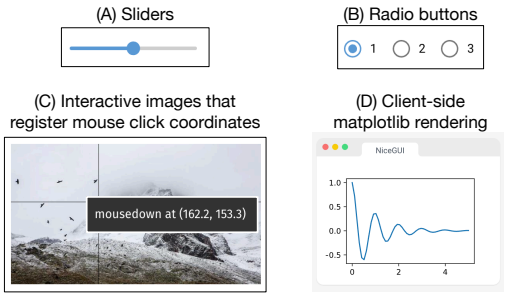

Figure 3: **Examples of advanced GUI capabilities provided by NiceGUI that can be leveraged with NiceWebRL.** Researchers can create (A) sliders for reporting numerical scores (B) radio buttons for selecting choices. (C) Beyond recording key strokes, researchers can record actions as (x, y)-coordinate selections on images or (D) leverage familiar plotting tools such as matplotlib for presenting graphs to participants.

4 NiceWebRL

NiceWebRL is a Python library that leverages Jax and NiceGUI to create online interfaces for human interaction with Jax-based virtual environments. The fundamental structural unit in NiceWebRL is the stage object, which represents distinct phases of a web experiment. Stages can display information, collect feedback via forms, or enable environment interaction. Researchers define experiments by sequencing different stage types. There are three main stage types: instruction stages, feedback stages, and environment interaction stages. For example, a researcher could have an instruction stage, followed by training and evaluation environment interaction stages. We describe our setup more formally in §A and provide more details on and examples of stage types in §B.

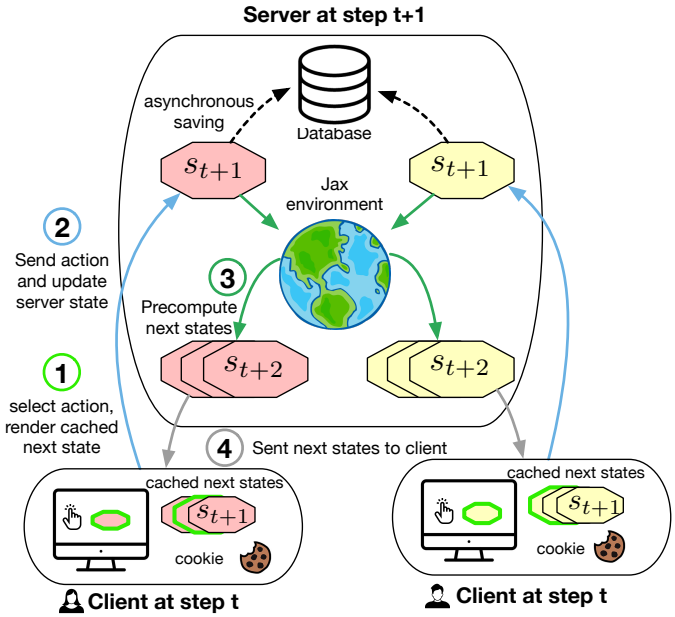

Figure 4: **How NiceWebRL leverages Jax to enable a single server to interact with multiple users**. Jax's functional paradigm prohibits inter-user state interference since each user has isolated environment states. Jax enables fast parallel computation of future states. Caching these states client side reduces latency.

Interacting with arbitrary environments on a remote server. The key innovation of NiceWebRL is the `EnvStage` object, which takes a Jax-based environment as input and allows participants to interact with the environment until some criteria is met (e.g. a minimum number of successful episodes). To obtain different environment behaviors across different parts of an experiment, `EnvStage` objects take in "environment parameters" (e.g. a fixed task distribution or friction coefficient) to define different dynamics based on a user's experiment stage. `EnvStage` leverages a functional paradigm to decouple (1) the function that defines how the environment will reset and evolve from (2) information about the participant's state and from the environment parameters. This

design allows a single compiled environment to be used across different experiment stages by different participants with their own independent environment state. We present an overview in Figure 4.

Reducing latency when presenting observations generated by a remote server. NiceWebRL precomputes potential next states to reduce latency. When a stage initializes interaction with a participant ($t = 0$), the server computes the initial state s_0 and observation o_0, then immediately computes all possible next states $\{s_1\}$ and observations $\{o_1\}$ for each potential action a. Both the initial observation and all potential next observations are sent to the client before any user interaction occurs. When the observation renders, time t_1 is recorded. When the participant selects an action a_0, time t_2 is recorded. The client immediately renders the corresponding precomputed observation o_1 and sends (a_0, t_1, t_2) to the server. The server then selects the appropriate state from its cache, computes the next set of potential states and observations for all possible actions, and sends these to the client for the next interaction. This parallel computation of all potential next states and observations, enabled by JAX, reduces network latency and helps provide immediate visual feedback to participants. We summarize this server-client Human-environment interaction protocol in Algorithm 8.

Data management and persistence. Having a persistent participant state across page reloads or web socket connection issues is key to a fluid experiment. We maintain this in two ways. First, we leverage NiceGUI to track unique identifier information held in browser session cookies. To track a participant's experiment progress, we exploit Jax's functional paradigm to track a user's environment state and current random number generator. We serialize these objects and store them for every environment-interaction in a SQL database. Whenever a connection is reset, we identify a user and reload their information for fluid re-engagement with the experiment.

Reducing latency when serving multiple clients. When serving multiple clients, performing I/O operations for persistence can be resource intensive and block other operations. To mitigate this, we save data asynchronously with a queue-based strategy that leverages stochastic exponential backoffs whenever a save fails. Resaving at exponentially increasing time-periods (with some noise) helps prevent collisions if multiple participants try to save concurrently. This is important for having a responsive UI when the server experiences many parallel participants.

Human-AI coordination. When a human coordinates with an artificial agent, we can have the agent be a part of the environment. When the environment steps, it not only computes the state and observation, it also computes the action that the artificial agents would compute for that state and uses that to predict all possible next states for the participant. Thanks to Jax, environments and learning-based agents can be compiled into one function, reducing latency from this extra computation.

AI-assisted task completion. This is a single-agent setting where an LLM has access to either the environment state s_t or environment observation o_t. Two options for leveraging LLMs exist. One option is to interact with the LLM via API calls. A second option is to use a local LLM. We provide examples of each in our examples folder.

5 Case studies

We present three case studies that display how NiceWebRL can help in the development of human-like AI, human-compatible, and human-assistive AI. In §5.1 and §5.2, we'll highlight how NiceWebRL contributed to new insights that span AI and cognitive science research on human-like and human-compatible AI models. We will first describe the experimental results of prior work and then discuss how this is made possible by NiceWebRL. In §5.3, we will present a proof-of-concept for how NiceWebRL can support research on LLM-based human-assistive AI.

5.1 Case study 1: Developing Human-like AI with NiceWebRL

Developing a novel Deep RL cognitive science model with NiceWebRL [14]. A central question in cognitive science is how people represent the environment to enable generalization to new tasks. Successor features (SFs) are a mechanism for how an agent can cache expectations of what it will see when pursuing a policy [3]. Recent ML research has shown that SFs enable agents to repurpose policies for new tasks [4, 13]. Later, cognitive scientists showed that SFs also explained how people reuse prior policies for new tasks [57]. However, the behavioral work was done in a small grid-world with 13 states. Carvalho et al. [14] studied whether SFs could explain how humans reuse behaviors for new tasks in 2 more complex domains: a maze gridworld (Figure 5 A) and Craftax [43], a 2D minecraft domain (Figure 5 D). Across both domains, they set up training tasks where a test object

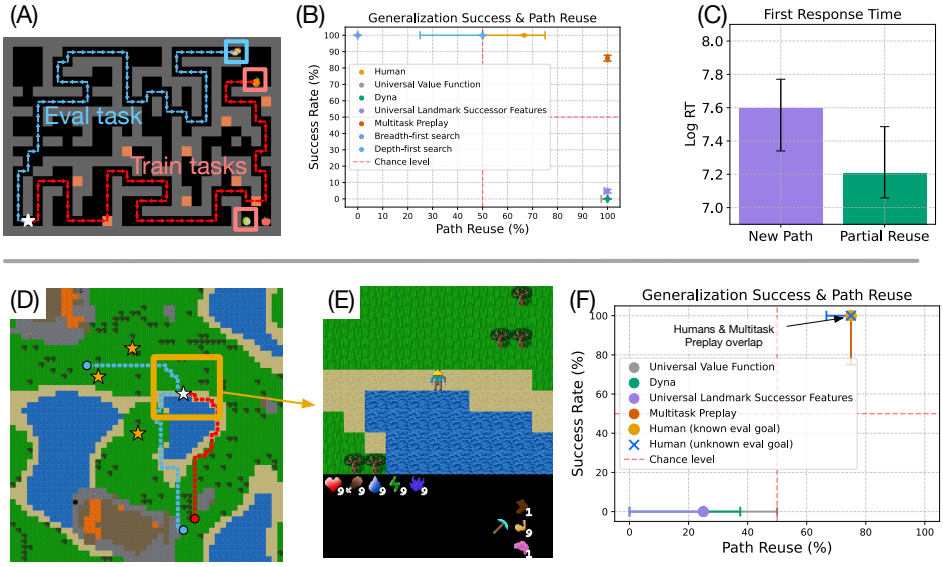

Figure 5: **Case study 1: NiceWebRL enabled the development of a novel Deep RL cognitive model that generalizes to new tasks with the same qualitative behaviors as Humans across multiple domains**. (A, D): a gridworld and 2D minecraft environment that both Human participants and Deep RL models learned in. (B, F): behavioral results studying the same phenomena across the two domains. NiceWebRL enabled developing a Deep RL cognitive model that could both (1) generalize to novel test goals in ways not permitted by previous methods, while (2) doing so with a similar suboptimal path reuse strategy that humans tend to exhibit. Figures reproduced from [14].

was visible from along the optimal training paths (e.g. top-right corner of Figure 5 A). SFs could not generalize here. People could; however, when people reused a training path to a novel goal, their response times suggested that they were using a caching-based solution rather than something more flexible—but expensive—like planning at decision-time (Figure 5 B-C). They developed an algorithm termed Multitask Preplay which *preemptively* learns solutions for unpursued tasks nearby training tasks by augmenting experience replay with small amounts of counterfactual simulation. They found this algorithm both better accounted for the response times people exhibited and better predicted how they would reuse prior behaviors. These results generalized to Craftax, where participants and models had to navigate from partial observations of a large world with many objects (Figure 5 E-F).

Role of NiceWebRL. First, NiceWebRL enabled comparing human behavior to advanced Deep RL algorithms including Successor Features. Second, NiceWebRL enabled using the same infrastructure to study both Deep RL algorithms and human behavior in two domains of increasing complexity: a gridworld and Craftax. This helped to ensure that findings were generalizable. Finally, NiceWebRL enabled the measurement of response times. This helped to adjudicate between theories that predict more or less computation at decision-time.

5.2 Case study 2: Developing Human-compatible AI with NiceWebRL

Developing a novel Multi-agent reinforcement learning (MARL) algorithm for coordinating with humans using NiceWebRL [33]. One central question in MARL is how we can develop MARL agents that can generalize to human partners without human training data. One current benchmark for human-compatible AI is the Overcooked domain [11] where agents must coordinate on basic cooking tasks. The state-of-the art algorithm is "Efficient End-to-End Training" [E3T; 62], a "Self Play" algorithm that plays with—and tries to predict the actions of—a noisy variant of itself. Jha et al. [33] developed a novel algorithm, Cross-Environment Cooperation (CEC), where an agent plays only against itself but across millions of different procedurally-generated environments (Figure 6 A). They found that while E3T [62] was able to "succeed" on more episodes when collaborating with humans (Figure 6 B), humans gave that agent a lower rating than CEC (Figure 6 C). The authors asked participants questions about their subjective experience using a Likert scale [41] and found that CEC was rated as more "adaptive" and "human-like"—despite succeeding *less* than E3T. When the authors analyzed game trajectories, they found CEC would collide less with humans across environments.

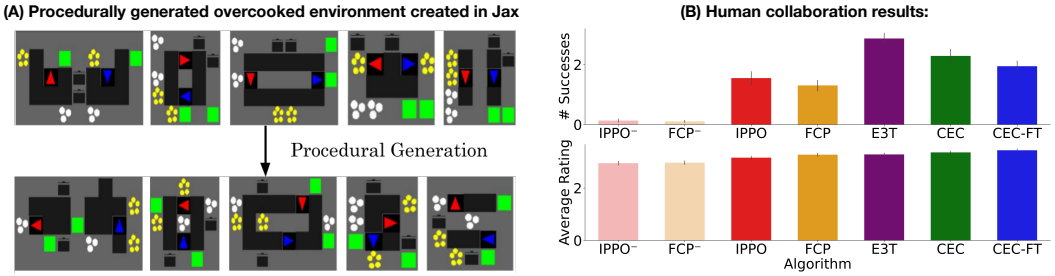

Figure 6: **Case study 2: NiceWebRL enabled the development of a novel MARL algorithm that is more compatible with novel human partners.** (A) A procedurally-generated environment used to design a novel MARL algorithm: Cross-Environment Cooperation (CEC). Prior work had agents learn with diverse sets of agents. CEC has a *single agent* play itself across millions of procedurally-generated environments. (B) While CEC *succeeded less* than other methods when collaborating with humans (top), it succeeded in ways that were *most favorable* to humans (bottom). Analysis suggested showed prior agents succeeded in less collaborative ways. Figures reproduced from [33].

Role of NiceWebRL. First, NiceWebRL enabled comparing multiple MARL algorithms in their ability to generalize to human partners. Second, NiceWebRL enabled the researchers to collect feedback from participants after every environment interaction stage using the "Feedback" stage object available in NiceWebRL coupled with NiceGUI's data collection GUI elements. This provided an easy way to get feedback from participants while agent-interaction data was fresh in their memories. Third, NiceWebRL stores all environment interactions so participant episodes could be analyzed post-hoc. This enabled the researchers to analyze trajectories by participants and agents to determine what qualitative behaviors (such as colliding) were different between the different MARL algorithms.

5.3 Case study 3: Developing Human-assistive AI with NiceWebRL

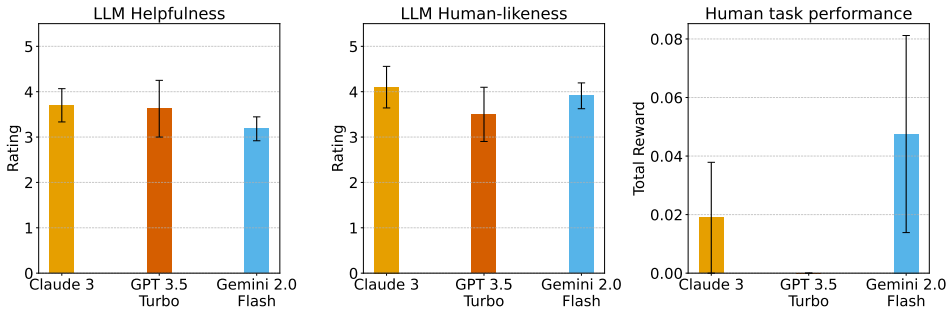

Figure 7: **Case study 3: Proof-of-concept experiment showing that NiceWebRL enables comparing how different LLMs can assist people in completing tasks.** We had Claude 3 Opus, GPT 3.5 Turbo, or Gemini 2.0 Flash act as assistants for people completing tasks in the XLand-Minigrid domain [46]. Each plot is showing the mean and standard error for 10 subjects per model.

Developing an LLM-assistant for sequential-decision making tasks in a virtual environment. We created a simple proof-of-concept experiment where people had to interact with tasks from the Xland-Minigrid domain [46]. To require assistance, they were given no task information but could ask an (anonymous) LLM assistant for help. At the beginning of the experiment, users were randomly assigned either Claude 3 Opus, GPT 3.5 Turbo, or Gemini 2.0 Flash. We set up our server so that it would interact with the LLMs via API calls[2]. Importantly, the LLM assistants were given text descriptions of the ground truth environment state including information on: (1) the goal of the episode (2) the locations and identities of all objects in the environment (3) the rules of the environment (e.g. how objects interact when combined). In principle, this can enable them to help users figure out the goal to maximize task reward. In this proof-of-concept, each participant completed 3 episodes, where a new task was sampled per episode. After all 3 episodes, participants

[2]we provide an example of how to set up a web experiment with a local LLM in our examples folder

answered two questions on a 5-point scale, "How helpful was the AI?" and "How human-like was the AI?". We collect data from 30 participants via CloudResearch. We describe details around recruiting participants in §D. We show results in Figure 7. For this proof-of-concept, we used older models with cheap API calls. We don't expect that these results are representative of what is possible with frontier models.

Role of NiceWebRL. NiceWebRL enabled using an existing ML domain to develop an experiment that studied how an LLM could assist people on long-horizon tasks. Additionally, the Feedback Stage object in NiceWebRL enabled collecting feedback from participants about the LLMs at the end of the experiment. This could also be done after every episode or during episodes.

6 Discussion and conclusion

We have presented NiceWebRL, a library for writing human subject experiments that leverage machine learning models and environments. Importantly, by integrating with NiceGUI, experiments with sophisticated GUI components can be written entirely in Python. NiceWebRL exploits Jax's compilation features and functional programming paradigm to reduce latency and enable multiple clients to interact with a single backend server. We demonstrated the utility of NiceWebRL with three case studies spanning both single-agent and multi-agent settings across 4 domains: a custom gridworld, Craftax [43], Overcooked [51], and Xland-Minigrid [46]. In the first case study, we showed that NiceWebRL could be used to measure human task performance and response times when developing a cognitive science model that could predict human behavior across two domains. In the second case study, we showed that NiceWebRL could be used to compare different MARL algorithms in their ability to generalize to humans without human training data. Here, we showed that NiceWebRL's stage objects facilitated qualitative and quantitative analysis studying why different algorithms were rated to be better collaborators by human participants. In our final case study, we showed that one could also use NiceWebRL to run experiments that measure how well LLMs can assist people on sequential decision-making tasks under asymmetric information constraints.

Limitations. While we've demonstrated NiceWebRL's utility for human subject experiments, many improvement avenues remain. We don't currently leverage Jax-based environment's ability to allow gradients to pass through their computational graph. Unsupervised environment design [17] can exploit this to automatically generate environments with different properties. Like vision researchers use gradient descent to generate stimuli for humans [21] and monkeys [59], NiceWebRL may be able to automatically generate environments for different target experimental conditions. Another limitation is that NiceWebRL currently only supports multi-agent domains with 2 agents. Future work can look to design an n-dimensional generalizations of the `EnvStage`object. Finally, while we precompute all next states to reduce latency, this may become prohibitive for large or continuous action spaces. Future work can integrate policy learning within the environment to select likely human actions for precomputing next states. Jax enables this policy to be incorporated into the environment's computational graph, which minimizes its computational cost.

NiceWebRL's reliance on Jax allows for a rich set of tools to improve future experiments. Jax has a growing ecosystem of libraries spanning probabilistic programming [6], Bayesian inference [10], LLM development [22], and general neural network development [29, 35]. This enables researchers from various disciplines to leverage NiceWebRL for human subject experiments regardless of their preferred modeling approach. Thus, we are optimistic that NiceWebRL can serve as a useful tool for future research developing Human-like, Human-compatible, and Human-assistive AI.

References

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. {TensorFlow}: a system for {Large-Scale} machine learning. In *12th USENIX symposium on operating systems design and implementation (OSDI 16)*, pages 265–283, 2016.

[2] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[3] André Barreto, Will Dabney, Rémi Munos, Jonathan J Hunt, Tom Schaul, Hado P van Hasselt, and David Silver. Successor features for transfer in reinforcement learning. *Advances in Neural Information Processing Systems*, 30, 2017.

[4] Andre Barreto, Diana Borsa, John Quan, Tom Schaul, David Silver, Matteo Hessel, Daniel Mankowitz, Augustin Zidek, and Remi Munos. Transfer in deep reinforcement learning using successor features and generalised policy improvement. In *International Conference on Machine Learning*, pages 501–510. PMLR, 2018.

[5] Benjamin Beyret, José Hernández-Orallo, Lucy Cheke, Marta Halina, Murray Shanahan, and Matthew Crosby. The animal-ai environment: Training and testing animal-like artificial cognition. *arXiv preprint arXiv:1909.07483*, 2019.

[6] Eli Bingham, Jonathan P. Chen, Martin Jankowiak, Fritz Obermeyer, Neeraj Pradhan, Theofanis Karaletsos, Rohit Singh, Paul A. Szerlip, Paul Horsfall, and Noah D. Goodman. Pyro: Deep universal probabilistic programming. *J. Mach. Learn. Res.*, 20:28:1–28:6, 2019. URL http://jmlr.org/papers/v20/18-403.html.

[7] Michał Bortkiewicz, Władek Pałucki, Vivek Myers, Tadeusz Dziarmaga, Tomasz Arczewski, Łukasz Kuci'nski, and Benjamin Eysenbach. Accelerating goal-conditioned RL algorithms and research. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/pdf/2408.11052.

[8] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL http://github.com/jax-ml/jax.

[9] Sébastien Bubeck, Varun Chadrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, et al. Sparks of artificial general intelligence: Early experiments with gpt-4, 2023.

[10] Alberto Cabezas, Adrien Corenflos, Junpeng Lao, and Rémi Louf. Blackjax: Composable Bayesian inference in JAX, 2024.

[11] Micah Carroll, Rohin Shah, Mark K Ho, Tom Griffiths, Sanjit Seshia, Pieter Abbeel, and Anca Dragan. On the utility of learning about humans for human-ai coordination. *Advances in neural information processing systems*, 32, 2019.

[12] Wilka Carvalho and Andrew Lampinen. Naturalistic computational cognitive science: Towards generalizable models and theories that capture the full range of natural behavior. *arXiv preprint arXiv:2502.20349*, 2025.

[13] Wilka Carvalho, Andre Saraiva, Angelos Filos, Andrew Lampinen, Loic Matthey, Richard L Lewis, Honglak Lee, Satinder Singh, Danilo Jimenez Rezende, and Daniel Zoran. Combining behaviors with the successor features keyboard. *Advances in Neural Information Processing Systems*, 36, 2024.

[14] Wilka Carvalho, Sam Hall-McMaster, Honglak Lee, and Sam Gershman. Preemptive solving of future problems: Multitask preplay in humans and machines. *arXiv preprint*, 2025.

[15] Katherine M Collins, Ilia Sucholutsky, Umang Bhatt, Kartik Chandra, Lionel Wong, Mina Lee, Cedegao E Zhang, Tan Zhi-Xuan, Mark Ho, Vikash Mansinghka, et al. Building machines that learn and think with people. *Nature human behaviour*, 8(10):1851–1863, 2024.

[16] Joshua R De Leeuw. jspsych: A javascript library for creating behavioral experiments in a web browser. *Behavior research methods*, 47:1–12, 2015.

[17] Michael Dennis, Natasha Jaques, Eugene Vinitsky, Alexandre Bayen, Stuart Russell, Andrew Critch, and Sergey Levine. Emergent complexity and zero-shot transfer via unsupervised environment design. *Advances in neural information processing systems*, 33:13049–13061, 2020.

[18] Nicolas Diekmann, Sandhiya Vijayabaskaran, Xiangshuai Zeng, David Kappel, Matheus Chaves Menezes, and Sen Cheng. Cobel-rl: A neuroscience-oriented simulation framework for complex behavior and learning. *Frontiers in Neuroinformatics*, 17:1134405, 2023.

[19] Yuqing Du, Eliza Kosoy, Alyssa Dayan, Maria Rufova, Pieter Abbeel, and Alison Gopnik. What can ai learn from human exploration? intrinsically-motivated humans and agents in open-world exploration. In *Neurips 2023 workshop: Information-theoretic principles in cognitive systems*, 2023.

[20] Holger Finger, Caspar Goeke, Dorena Diekamp, Kai Standvoß, and Peter König. Labvanced: a unified javascript framework for online studies. In *International conference on computational social science (cologne)*, pages 1–3. University of Osnabrück Cologne, 2017.

[21] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. In *International conference on learning representations*, 2018.

[22] Xinyang Geng. Easylm: A simple and scalable training framework for large language models, 2023. URL https://github.com/young-geng/EasyLM.

[23] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

[24] Todd M Gureckis, Jay Martin, John McDonnell, Alexander S Rich, Doug Markant, Anna Coenen, David Halpern, Jessica B Hamrick, and Patricia Chan. psiturk: An open-source framework for conducting replicable behavioral experiments online. *Behavior research methods*, 48:829–842, 2016.

[25] William H Guss, Brandon Houghton, Nicholay Topin, Phillip Wang, Cayden Codel, Manuela Veloso, and Ruslan Salakhutdinov. Minerl: A large-scale dataset of minecraft demonstrations. *arXiv preprint arXiv:1907.13440*, 2019.

[26] Dylan Hadfield-Menell, Stuart J Russell, Pieter Abbeel, and Anca Dragan. Cooperative inverse reinforcement learning. *Advances in neural information processing systems*, 29, 2016.

[27] Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models, 2023. *URL https://arxiv. org/abs/2301.04104*, 2023.

[28] Assaf Hallak, Dotan Di Castro, and Shie Mannor. Contextual markov decision processes. *arXiv preprint arXiv:1502.02259*, 2015.

[29] Jonathan Heek, Anselm Levskaya, Avital Oliver, Marvin Ritter, Bertrand Rondepierre, Andreas Steiner, and Marc van Zee. Flax: A neural network library and ecosystem for JAX, 2024. URL http://github.com/google/flax.

[30] Felix Henninger, Yury Shevchenko, Ulf K Mertens, Pascal J Kieslich, and Benjamin E Hilbig. lab. js: A free, open, online study builder. *Behavior Research Methods*, pages 1–18, 2021.

[31] Larissa Hjorth, Ingrid Richardson, Hugh Davies, and William Balmford. *Exploring Minecraft: Ethnographies of play and creativity*. Springer Nature, 2021.

[32] Wayne Holmes and Ilkka Tuomi. State of the art and practice in ai in education. *European journal of education*, 57(4):542–570, 2022.

[33] Kunal Jha, Wilka Carvalho, Yancheng Liang, Simon S Du, Max Kleiman-Weiner, and Natasha Jaques. Cross-environment cooperation enables zero-shot multi-agent coordination. In *International Conference on Machine Learning (ICML)*, 2025.

[34] Leslie Pack Kaelbling, Michael L Littman, and Anthony R Cassandra. Planning and acting in partially observable stochastic domains. *Artificial intelligence*, 101(1-2):99–134, 1998.

[35] Patrick Kidger and Cristian Garcia. Equinox: neural networks in JAX via callable PyTrees and filtered transformations. *Differentiable Programming workshop at Neural Information Processing Systems 2021*, 2021.

[36] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4015–4026, 2023.

[37] Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Matt Deitke, Kiana Ehsani, Daniel Gordon, Yuke Zhu, et al. Ai2-thor: An interactive 3d environment for visual ai. *arXiv preprint arXiv:1712.05474*, 2017.

[38] Sotetsu Koyamada, Shinri Okano, Soichiro Nishimori, Yu Murata, Keigo Habara, Haruka Kita, and Shin Ishii. PGX: Hardware-accelerated parallel game simulators for reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[39] Brenden M Lake, Tomer D Ullman, Joshua B Tenenbaum, and Samuel J Gershman. Building machines that learn and think like people. *Behavioral and brain sciences*, 40:e253, 2017.

[40] Joel Z Leibo, Cyprien de Masson d'Autume, Daniel Zoran, David Amos, Charles Beattie, Keith Anderson, Antonio García Castañeda, Manuel Sanchez, Simon Green, Audrunas Gruslys, et al. Psychlab: a psychology laboratory for deep reinforcement learning agents. *arXiv preprint arXiv:1801.08116*, 2018.

[41] Rensis Likert. A technique for the measurement of attitudes. *Archives of psychology*, 1932.

[42] Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, et al. Agentbench: Evaluating llms as agents. *arXiv preprint arXiv:2308.03688*, 2023.

[43] Michael Matthews, Michael Beukman, Benjamin Ellis, Mikayel Samvelyan, Matthew Jackson, Samuel Coward, and Jakob Foerster. Craftax: A lightning-fast benchmark for open-ended reinforcement learning. In *International Conference on Machine Learning (ICML)*, 2024.

[44] Michael Matthews, Michael Beukman, Chris Lu, and Jakob Foerster. Kinetix: Investigating the training of general agents through open-ended physics-based control tasks. In *International Conference on Learning Representations*, 2025.

[45] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

[46] Alexander Nikulin, Vladislav Kurenkov, Ilya Zisman, Artem Agarkov, Viacheslav Sinii, and Sergey Kolesnikov. XLand-minigrid: Scalable meta-reinforcement learning environments in JAX. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

[47] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Nicolas Gribonval, Zachary Devito, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems*, pages 8024–8035, 2019.

[48] Jonathan W Peirce. Psychopy—psychophysics software in python. *Journal of neuroscience methods*, 162(1-2):8–13, 2007.

[49] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021.

[50] Stuart Russell. Human-compatible artificial intelligence., 2022.

[51] Alexander Rutherford, Benjamin Ellis, Matteo Gallici, Jonathan Cook, Andrei Lupu, Gardar Ingvarsson, Timon Willi, Akbir Khan, Christian Schroeder de Witt, Alexandra Souly, Saptarashmi Bandyopadhyay, Mikayel Samvelyan, Minqi Jiang, Robert Tjarko Lange, Shimon Whiteson, Bruno Lacerda, Nick Hawes, Tim Rocktaschel, Chris Lu, and Jakob Nicolaus Foerster. Jax-MARL: Multi-agent RL environments in JAX. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

[52] Melanie Sclar, Sachin Kumar, Peter West, Alane Suhr, Yejin Choi, and Yulia Tsvetkov. Minding language models'(lack of) theory of mind: A plug-and-play multi-character belief tracker. *arXiv preprint arXiv:2306.00924*, 2023.

[53] Rohin Shah, Pedro Freire, Neel Alex, Rachel Freedman, Dmitrii Krasheninnikov, Lawrence Chan, Michael D Dennis, Pieter Abbeel, Anca Dragan, and Stuart Russell. Benefits of assistance over reward learning. *Arxiv*, 2020.

[54] Mohammed Yousef Shaheen. Applications of artificial intelligence (ai) in healthcare: A review. *ScienceOpen Preprints*, 2021.

[55] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *nature*, 550(7676):354–359, 2017.

[56] Adaptive Agent Team, Jakob Bauer, Kate Baumli, Satinder Baveja, Feryal Behbahani, Avishkar Bhoopchand, Nathalie Bradley-Schmieg, Michael Chang, Natalie Clay, Adrian Collister, et al. Human-timescale adaptation in an open-ended task space. *arXiv preprint arXiv:2301.07608*, 2023.

[57] Momchil S Tomov, Eric Schulz, and Samuel J Gershman. Multi-task reinforcement learning in humans. *Nature Human Behaviour*, 5(6):764–773, 2021.

[58] Sai H Vemprala, Rogerio Bonatti, Arthur Bucker, and Ashish Kapoor. Chatgpt for robotics: Design principles and model abilities. *Ieee Access*, 2024.

[59] Binxu Wang and Carlos R Ponce. Tuning landscapes of the ventral stream. *Cell Reports*, 41(6), 2022.

[60] Tom Ward, Andrew Bolt, Nik Hemmings, Simon Carter, Manuel Sanchez, Ricardo Barreira, Seb Noury, Keith Anderson, Jay Lemmon, Jonathan Coe, et al. Using unity to help solve intelligence. *arXiv preprint arXiv:2011.09294*, 2020.

[61] Yue Wu, Xuan Tang, Tom M Mitchell, and Yuanzhi Li. Smartplay: A benchmark for llms as intelligent agents. *arXiv preprint arXiv:2310.01557*, 2023.

[62] Xue Yan, Jiaxian Guo, Xingzhou Lou, Jun Wang, Haifeng Zhang, and Yali Du. An efficient end-to-end training approach for zero-shot human-ai coordination. *Advances in Neural Information Processing Systems*, 36:2636–2658, 2023.

[63] Lance Ying, Katherine M Collins, Lionel Wong, Ilia Sucholutsky, Ryan Liu, Adrian Weller, Tianmin Shu, Thomas L Griffiths, and Joshua B Tenenbaum. On benchmarking human-like intelligence in machines. *arXiv preprint arXiv:2502.20502*, 2025.

[64] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to prompt for vision-language models. *International Journal of Computer Vision*, 130(9):2337–2348, 2022.

[65] Pei Zhou, Aman Madaan, Srividya Pranavi Potharaju, Aditya Gupta, Kevin R McKee, Ari Holtzman, Jay Pujara, Xiang Ren, Swaroop Mishra, Aida Nematzadeh, et al. How far are large language models from agents with theory-of-mind? *arXiv preprint arXiv:2310.03051*, 2023.