STASIS: REINFORCEMENT LEARNING SIMULATORS FOR HUMAN-CENTRIC REAL-WORLD ENVIRONMENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

We introduce Stasis, a suite of reinforcement learning (RL) environments that aim to maintain realism for human-centric agents operating in real-world environments. Through representation learning and alignment with real-world offline data, Stasis allows RL systems to be trained in offline environments with tunable characteristics, such as observability, heterogeneity and levels of missing data. The resulting RL agents are capable of maintaining a level of performance and robustness that is comparable to agents trained in real-world online environments, while avoiding the high cost and risk associated with making mistakes during online training. We provide examples of two environments that will be part of Stasis and discuss its implications for the deployment of RL-based systems in sensitive and high-risk sectors.

1 INTRODUCTION

Reinforcement Learning (RL) is becoming increasingly popular for a variety of tasks ranging from robotic control and autonomous driving to artificial intelligence in the gaming domain. Despite its potential, the lack of realistic simulators for RL agents operating in the real-world is a major limitation for the development of RL agents. Current simulators lack the capability to model real-world applications of RL. This includes missing key components such as heterogeneity within the environment, specifically within the reward function, and observability, as all real-world environment are inherently perceived as partially observable. Furthermore, these simulators often lack the ability to handle missing data, either irregularly sampled data or observations missing at random (due design of device) and missing not-at-random (due to outcomes). Lastly, they lack the ability to generate observed data, as simulators should be thought of as a generative model of the real-world, where we want to generate samples close to the observed data.

The lack of realistic simulators for RL agents hinders the development of agents that can be successfully deployed to real-world tasks. The high cost and risk associated with making mistakes during online training makes it an important problem to address. To this end, we introduce Stasis, a suite of RL environments that aim to maintain realism for human-centric agents operating in real-world environments. Through representation learning and alignment with real-world offline data, Stasis allows RL systems to be trained in offline environments with tunable characteristics, such as observability, heterogeneity and levels of missing data. The resulting RL agents are capable of maintaining a level of performance and robustness that is comparable to agents trained in real-world online environments, while avoiding the high cost and risk associated with making mistakes during online training.

Related Work. The most similar work to the one we present here is the Gymnasium, formerly known as OpenAI gym as seen in Brockman et al. (2016), and the Safety Gym by Ray et al. (2019). Both of these are suites of environments where RL agents can be trained without requiring real-world deployment. However, they both place emphasis on robotics and control, with Safety Gym making use of MuJoCo (Todorov et al., 2012) with a focus on constrained RL. The Stasis library which we introduce here will focus on open problems related to RL in healthcare, including partial observability, heterogeneity, missing data and make use of labelled real-world data through offline RL (Levine et al., 2020).

2 UNDERLYING FRAMEWORK & CONSIDERATIONS

On-going efforts to build simulated environments for benchmarking conventional and emerging RL algorithms center on emulating the realism and practicality of real world settings. Evaluating algorithms in this fashion affords practitioners the ability to rigorously examine the suitability of an algorithm before initial deployment in the real world. In this paper, we identify four *core* themes that are important for representing *human-centric* real-world settings. Specifically, settings where the decision-making policy directly interacts with a human, or provides actions for a human to execute within their own environment.

Observability. Observability determines the full-range of information from the environment available to the agent for decision-making. In real-world settings, environments are typically partially observable; the agent only has access to a limited view of the current state of the environment (Littman, 2009). This can make it difficult to learn an optimal policy, as the agent may be missing important information or have to rely on incomplete observations to determine its actions. Therefore, a well-designed observability mechanism that captures the relevant information is critical for learning a good policy. However, increasing observability can also lead to higher computational and memory requirements, making it important to strike a balance between having enough information to make informed decisions and keeping the complexity manageable.

Heterogeneity. In real-world settings, the reward signal may vary between agents operating within a single environment. As such, learning a single policy that aims to optimize the reward for all agents is often difficult, leading to sub-optimal performance for select agents (Chen et al., 2022; Jin et al., 2022). Generally, this can result in a situation where some agents learn different, unintended behaviors. In multi-agent systems, this can lead to a lack of coordination, potentially hampering the functioning of the overall system. In applications such as healthcare where data from heterogeneous subjects are often used to make decisions for single subject, failing to account for heterogeneity in the reward signal can lead to an alignment problem, severely impacting the relevancy of the learned policy. Mitigating these challenges may require algorithms to leverage techniques from areas of research such as multi-agent reinforcement learning, or to directly modify the reward functions to account for the heterogeneity between agents.

Missing Data. The effectiveness of a policy in reinforcement learning is closely tied to the quality and quantity of data used to train the model. If the agent encounters missing data, such as incomplete or unavailable state or reward information, it may be unable to accurately estimate the value of different actions, leading to suboptimal decisions (Awan et al., 2022; Lizotte et al., 2008; Shortreed et al., 2011). Missing data can happen due to irregular sampling, where data is missing at random, which can occur due to various factors such as technical failures or data collection constraints. Additionally, data may be missing not at random, such as when specific actions or states are more likely to be absent. In healthcare application, this can be due to phenomena such as self-selection bias, where participants in the study exercise control over whether or not they participate in the study or how much data they provide. Therefore, it is essential to consider the consequences of missing data and address it using techniques such as imputation, data augmentation, or other advanced methods for handling missing data in reinforcement learning (Shortreed et al., 2011; Awan et al., 2022).

Offline Data. Previously collected experiential data from agents interacting within a given environment can be used to enhance the robustness and reliability of the simulated environment. We envision that offline data can be used to improve the following aspects of the simulated environment: (1) *state representation* – offline data can be used to provide more accurate state representations for the agent, including information about the environment, objects, and other agents (Zang et al., 2022); (2) *model dynamics* – the interactions between objects and agents in the environment can be modeled more accurately using offline data, allowing for a more realistic representation of the environment's dynamics Kidambi et al. (2020). Lastly, in most real-world environment, decision-making policies are rarely trained from scratch, rather offline data is commonly used to learn policies that achieve an acceptable level of performance (Levine et al., 2020). As such, incorporating available offline data into simulated environments allows for a pre-training phase, where policies are first initialized using offline data before being deployed within the environment.

3 THE SIMULATOR

The structure of the simulator is similar to the structure of the Gymnasium API (Brockman et al., 2016). To initialize an environment you pull it from the library's collection and then proceed to train your agent using the simulated environment. Each environment has the same structure regarding methods, in order for the users to be able to switch among environments and train on different scenarios with ease.

The difference to the Gymnasium API is that the environments also share parameters related to problems you will find in real world applications, in order to make the environments more realistic and thus the agents more robust to real world data. When initializing an environment, you will be able to specify the complexity of the problem, but also some parameters that are important in a healthcare context (Awrahman et al., 2022) and which are problematic in the collection and curation of healthcare data (Pezoulas et al., 2019). The shared parameters, when it is possible for an environment, will be able to tune aspects such as the heterogeneity of the simulation (Angus & Chang, 2021), incorporate missing data or have partial observability and add stochasticity or noise to the simulation. This can look different for every environment, but the purpose of the parameters is shared.

3.1 HEALTHY TRAVELING SALESMAN

The first environment, which is already implemented, is a simulated weighted travelling salesman problem (Lu et al., 2020). The environment is initialized by providing a set of coordinates that have to be visited each once and an exposure type (e.g. greenspace or bars) that has to be maximized or minimized. Then the environment will collect information on the possible routes you can take to visit each coordinate using the OpenRouteService API (Neis & Zipf, 2008) and the exposures around the locations of interest using the Overpass API (Olbricht, 2015) which both use data collected from OpenStreetMap or OSM for short (OpenStreetMap contributors, 2017). In the current implementation one of the coordinates to be provided is the starting location and the rest of the coordinates are the ones that are visited and then the agent needs to return to the starting point. The task is thus finding the optimized circle in a graph and complexity of the problem is increased by simply adding more coordinates.

The reward for each action is a weighted average between the distance covered and the time spent at exposure at each route, which is also tunable at input depending on what the agent should focus on optimizing. The possible exposure information are limited only by the possible types of locations you can collect from OSM. In terms of the parameters mentioned before on making the simulations more realistic to collected data, these are still a work in progress, but possible concepts discussed include modifying heterogeneity by adding different constraints in the possible actions of different simulated users. Some users have trouble moving large distances or want to avoid certain trigger areas, which can be encompassed in the reward function. In terms of missing data and stochasticity we can add parameters that tune the amount of information and noise we see in the possible routes. They can also be encompassed in a way that matches what we see from GPS collecting devices like smartphones and smartwatches, where missing data are not missing at random, but there are certain time-periods that continuously data are not being collected by the smart devices (Barnett & Onnela, 2018).

The following figures 1 and 2 are examples of rendering of the environment, where the green areas are greenspace locations (Novack et al., 2018) collected by the Overpass API, the blue arrows are the coordinates of the locations to be visited and the red home arrow indicates the coordinates of the starting and ending location. This is the visualization after an episode has been run using a Deep-QN agent (Mnih et al., 2013) trained on maximizing greenspace exposure on a set of 8 coordinates in Boston, MA (Figure 1) and a set of 6 coordinates in Bronx, NY (Figure 2).

A big advantage of this environment is that all the information in the simulation are collected from the real world. Researchers that want to use this environment and possess offline GPS data can also encompass them to enrich the information in the environment and make it even more realistic (Gur et al., 2022). Using GPS trajectories, we can collect information on areas that people want to avoid, areas with more traffic or areas which can be traversed faster and encompass this information in the reward function of the environment. The GPS data can also be used to gain information of people's

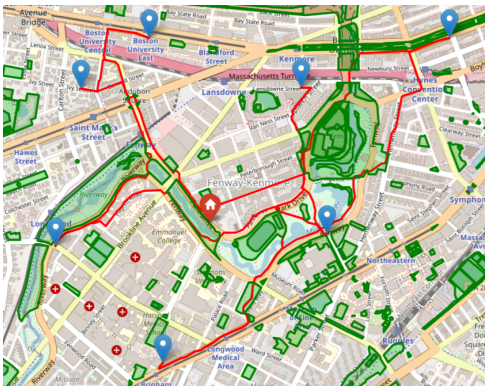

Figure 1: Map Environment rendering (Boston, MA)

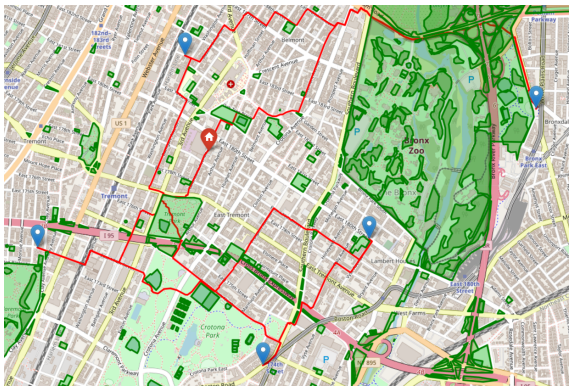

Figure 2: Map Environment rendering (Bronx, NY)

home and work locations or locations they like to visit frequently, thus making the environment adjust to a specific person's patterns and visit locations.

3.2 Resource Allocation in Clinical Settings

The third environment model in the Stasis library demonstrates a common problem found in clinical settings: dynamic resource allocation. Allocating resources efficiently and operationally requires decision making on a case-by-case basis, considering both the resources available and the individual conditions of multiple patients, which are not necessarily a function of only the resources themselves.

This environment's properties can be highly complex due to the sophistication of modern clinical settings. However, for its first iteration, it will be limited to a general setting. The action space will involve selecting resources from a given set, where each resource allocation carries a cost and takes up the resource for a certain period of time. We will not know the exact cost or duration, as these vary depending on the resource in question. For example, it takes less time to record blood pressure than to perform a blood test, as some people need extra preparation time for invasive procedures. The state space will include the set of available resources, their characteristics, the occupancy of the clinical section, the time and date, other features that help forecast future occupancy, and relevant patient features, outcomes of used resources, and further diagnosis.

The main goal of this framework is to understand the relationship between resources and patient outcomes. Domain expertise can be incorporated into the simulator by adjusting the outcomes from different resources and by using imitation learning or feedback. This allows the agent to be grounded in best practices, while giving it the opportunity to explore different strategies, in the safe, non-destructive environment of the simulator.

4 Future Directions

As the initiative grows, it will be important to focus on community building. This can be accomplished by creating a leaderboard, hosting workshops, and adding existing standalone environments. Such efforts will help to bring more researchers and practitioners to the platform, creating a strong community. Another focus of the library will be taking advantage of existing data to build more realistic environments. By leveraging existing offline data, the library could potentially use algorithms such as pretraining or initialization phases to further refine the environment and help it to behave in the most realistic way possible. Finally, there should be an active goal to make the environments relevant and useful in a medical or clinical context. To do this, researchers and developers will seek to collaborate with medical professionals to ensure the simulators are based on real world observations and are as accurate as possible. By doing so, Stasis can become a valuable tool for medical professionals.

REFERENCES

Derek C. Angus and Chung-Chou H. Chang. Heterogeneity of Treatment Effect. *JAMA*, 326(22): 2312, 12 2021. doi: 10.1001/jama.2021.20552. URL http://dx.doi.org/10.1001/jama.2021.20552.

Saqib Ejaz Awan, Mohammed Bennamoun, Ferdous Sohel, Frank Sanfilippo, and Girish Dwivedi. A reinforcement learning-based approach for imputing missing data. *Neural Computing and Applications*, 34(12):9701–9716, 6 2022. ISSN 14333058. doi: 10.1007/S00521-022-06958-3/TABLES/13. URL https://link.springer.com/article/10.1007/s00521-022-06958-3.

Banan Jamil Awrahman, Chia Aziz Fatah, and Mzhda Yasin Hamaamin. A Review of the Role and Challenges of Big Data in Healthcare Informatics and Analytics. *Computational Intelligence and Neuroscience*, 2022:1–10, 9 2022. doi: 10.1155/2022/5317760. URL http://dx.doi.org/10.1155/2022/5317760.

Ian Barnett and Jukka-Pekka Onnela. Inferring mobility measures from GPS traces with missing data. *Biostatistics*, 21(2):e98–e112, 10 2018. doi: 10.1093/biostatistics/kxy059. URL http://dx.doi.org/10.1093/biostatistics/kxy059.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. OpenAI Gym. *arXiv: Learning*, 6 2016. URL https://www.arxiv.org/pdf/1606.01540.

Elynn Y. Chen, Rui Song, and Michael I. Jordan. Reinforcement Learning with Heterogeneous Data: Estimation and Inference. 1 2022. doi: 10.48550/arxiv.2202.00088. URL https://arxiv.org/abs/2202.00088v1.

Izzeddin Gur, Ofir Nachum, and Aleksandra Faust. Targeted environment design from offline data, 2022. URL https://openreview.net/forum?id=Is5Hpwg2R-h.

Hao Jin, Yang Peng, Wenhao Yang, Shusen Wang, and Zhihua Zhang. Federated Reinforcement Learning with Environment Heterogeneity, 5 2022. ISSN 2640-3498. URL https://proceedings.mlr.press/v151/jin22a.html.

Rahul Kidambi, Aravind Rajeswaran, Praneeth Netrapalli, and Thorsten Joachims. MOReL : Model-Based Offline Reinforcement Learning. *Advances in Neural Information Processing Systems*, 2020-Decem, 5 2020. ISSN 10495258. doi: 10.48550/arxiv.2005.05951. URL https://arxiv.org/abs/2005.05951v3.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

Michael L. Littman. A tutorial on partially observable Markov decision processes. *Journal of Mathematical Psychology*, 53(3):119–125, 6 2009. ISSN 0022-2496. doi: 10.1016/J.JMP.2009.01.005.

Daniel J. Lizotte, Lacey Gunter, Eric B. Laber, and Susan A. Murphy. Missing data and uncertainty in batch reinforcement learning. 2008.

Hao Lu, Xingwen Zhang, and Shuang Yang. A Learning-based Iterative Method for Solving Vehicle Routing Problems. *International Conference on Learning Representations*, 4 2020. URL https://www.openreview.net/pdf?id=BJe1334YDH.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing Atari with Deep Reinforcement Learning. *arXiv: Learning*, 12 2013. URL http://cs.nyu.edu/~koray/publis/mnih-atari-2013.pdf.

P Neis and A Zipf. OpenRouteService.org – Combining Open Standards and Open Geodata. *The State of the Map. 2nd Open Street Maps Conference, Limerik. Ireland.*, 2008.

Tessio Novack, Zhiyong Wang, and Alexander Zipf. A System for Generating Customized Pleasant Pedestrian Routes Based on OpenStreetMap Data. *Sensors*, 18(11):3794, 11 2018. doi: 10.3390/s18113794. URL http://dx.doi.org/10.3390/s18113794.

Roland M. Olbricht. Data Retrieval for Small Spatial Regions in OpenStreetMap. *Lecture Notes in Geoinformation and Cartography*, pp. 101–122, 2015. doi: 10.1007/978-3-319-14280-7\{_}6. URL http://dx.doi.org/10.1007/978-3-319-14280-7_6.

OpenStreetMap contributors. Planet dump retrieved from https://planet.osm.org . https://www.openstreetmap.org, 2017.

Vasileios C. Pezoulas, Konstantina D. Kourou, Fanis Kalatzis, Themis P. Exarchos, Aliki Venetsanopoulou, Evi Zampeli, Saviana Gandolfo, Fotini Skopouli, Salvatore De Vita, Athanasios G. Tzioufas, and Dimitrios I. Fotiadis. Medical data quality assessment: On the development of an automated framework for medical data curation. *Computers in Biology and Medicine*, 107: 270–283, 4 2019. doi: 10.1016/j.compbiomed.2019.03.001. URL http://dx.doi.org/10.1016/j.compbiomed.2019.03.001.

Alex Ray, Joshua Achiam, and Dario Amodei. Benchmarking safe exploration in deep reinforcement learning. *arXiv preprint arXiv:1910.01708*, 7(1):2, 2019.

Susan M Shortreed, Eric Laber, Daniel J Lizotte, · T Scott Stroup, Joelle Pineau, Susan A Murphy, S M Shortreed, J Pineau, T S Stroup, E Laber, · D J Lizotte, S A Murphy, and D J Lizotte. Informing sequential clinical decision-making through reinforcement learning: an empirical study. *Mach Learn*, 84:109–136, 2011. doi: 10.1007/s10994-010-5229-0.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pp. 5026–5033. IEEE, 2012.

Hongyu Zang, Xin Li, Jie Yu, Chen Liu, Riashat Islam, Remi Tachet Des Combes, and Romain Laroche. Behavior Prior Representation learning for Offline Reinforcement Learning. 11 2022. doi: 10.48550/arxiv.2211.00863. URL https://arxiv.org/abs/2211.00863v2.