EuroVis Workshop on Visual Analytics (2022)
M. Angelini and J. Bernard (Editors)

A Comprehensive Workflow for Effective Imitation and Reinforcement Learning with Visual Analytics

Yannick Metz[1], Udo Schlegel[1] , Mennatallah El-Assady[2], Daniel Seebacher[1], Daniel Keim[1]

[1]University of Konstanz, Germany [2]ETH Zürich, Switzerland

Abstract

Multiple challenges hinder the application of reinforcement learning algorithms in experimental and real-world use cases even with recent successes in such areas. Such challenges occur at different stages of the development and deployment of such models. While reinforcement learning workflows share similarities with machine learning approaches, we argue that distinct challenges can be tackled and overcome using visual analytic concepts. Thus, we propose a comprehensive workflow for reinforcement learning and present an implementation of our workflow incorporating visual analytic concepts integrating tailored views and visualizations for different stages and tasks of the workflow.

CCS Concepts

• ***Human-centered computing*** → *Visual analytics;* • ***Computing methodologies*** → *Reinforcement learning;*

1. Introduction

Recently, there have been notable examples of the capabilities of Reinforcement Learning (RL) in diverse fields like robotics [NL19], Physics [MGL21] or even video-compression [M*22]. Despite these successes, the application and evaluation of recent deep reinforcement and imitation learning techniques in real-world scenarios are still limited. Existing research almost exclusively focuses on synthetic benchmarks and use cases [BNVB13]. We argue that the usage and evaluation in realistic scenarios is a mandatory step in assessing the capabilities of current approaches and identifying existing weaknesses and possibilities for further development. In this paper, we present a visual analytics workflow and an instantiation of the approach that facilitates the application of state-of-the-art algorithms to various scenarios. Our presented approach is designed to specifically support domain experts, with basic knowledge of core concepts in reinforcement learning, who are interested in applying RL algorithms to domain-specific sequential-decision making tasks. The goal is to enable the effective application of their knowledge to (1) design agents and simulation environments including reward functions, and (2) a detailed assessment of trained agents' capabilities in terms of performance, robustness, and traceability. A structured and well-defined approach can also help to critically investigate and combat some fundamental difficulties of reinforcement learning like *brittleness*, *generalization* to new tasks and environments, and issues of *reproducibility* [DLM*20, HIB*17, ZBP18].

Outside of reinforcement and imitation learning, there exists a wide range of workflows and interactive visual analytics (VA) tools for the training and evaluation of ML models [ACD*15, LSL*16, LWLZ17, ERT*18, ALA*18, SKKC19, SSSE20]. Compared to other fields of machine learning, there has been less work on applying visual analytics in the space of reinforcement and especially imitation learning. A large number of necessary decisions and the existence of interconnected tasks make the application of interactive machine learning, with close coupling of model and human, especially valuable for reinforcement learning.
Existing work such as *DQNViz* by Wang et al. [WGSY19] enables the analysis of spatial behavior patterns of agents in Atari environments like Breakout (see Arcade Learning Environment [BNVB13]) using visual analytics. He et al. present *DynamicsExplorer* [HLv*20] to evaluate and diagnose a trained policy in a robotics use case, which incorporates views to track the trajectories of a ball in the maze during episodes. The application enables the inspection of the effect of real-world conditions for trained agents. Saldanha et al. [SPBA19] showcase an application that supports data scientists during experimentation by increasing situational awareness. Key elements are thumbnails summarizing agent performance during episodes, and specialized views to understand the connection between particular hyperparameter settings and training performance.

Compared to the existing approaches, we (1) Extend the existing frameworks to encompass a holistic view of the relevant stages of the reinforcement learning process instead of just sub-tasks; (2) present a generic, easily adaptable application, which can be instantiated to specific use cases; (3) explicitly consider imitation learning, due to the frequent use in conjunction with reinforcement learning; (4) apply our framework in a novel, custom real-world use case instead of an existing benchmark environment.

2. A Workflow for User-Guided RL Experiments

To the best of our knowledge, there has not been a comprehensive workflow for experimentation and application of reinforcement

DOI: 10.2312/eurova.20221074

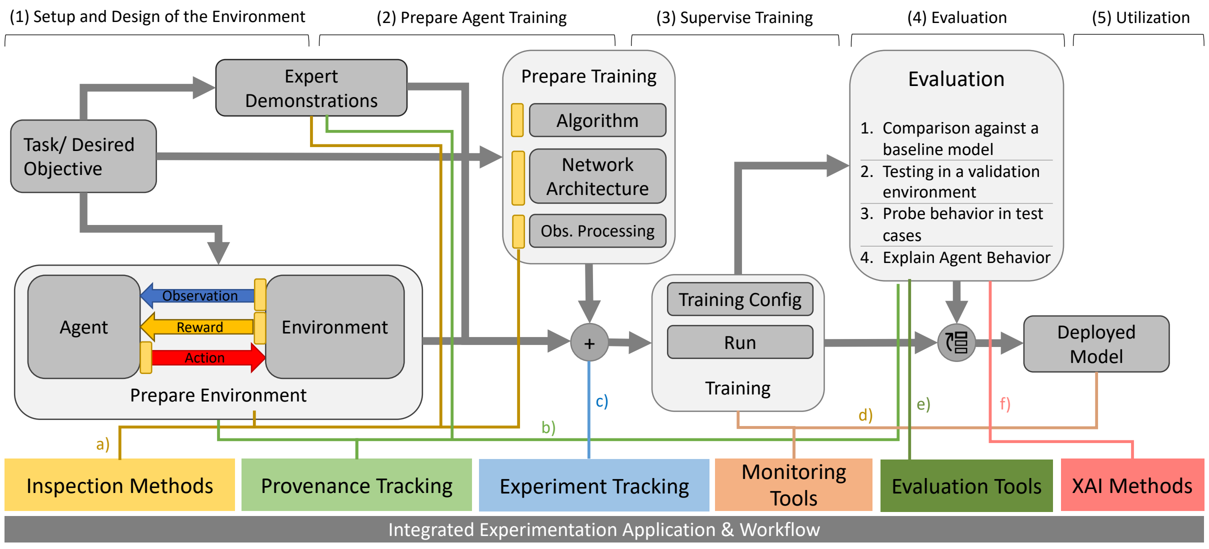

Figure 1: *Overview of the **RIVA** (**R**einforcement and **I**mitation Learning with **V**isual **A**nalytics) workflow. **RIVA** is an integrated experimentation workflow and application, that provides a range of tools to support all major critical steps: a) Inspecting Observations, Actions, and Rewards and ensuring matching values between simulation, expert demonstrations, and architectures, b) Provenance Tracking of interesting states to enable targeted case-based evaluation, c) Tracking of parameters and settings to ensure reproducibility and understand the effect of design decision, d) Interactively monitor training and final performance beyond reward, e) enable effective evaluation by integrating multiple evaluation tools, e) Explain behavior by natively integrating XAI methods like input attribution techniques*

learning tightly incorporating users. This leaves both researchers and practitioners to loosely defined best practices. In the following chapter, we outline a conceptual workflow for developers and researchers, which we base on guides, projects [Ach18, Irp18], and popular open-source libraries [RHE*19, HRE*18]. We follow the terminology used, e.g., in the *Gym* package [BCP*16]. As a starting point, we consider the fundamental workflow from Sacha et al. [SKKC19] that is aimed at generic ML tasks:

A. Prepare-Data: Data selection, cleaning, and transformations; detection of faulty or missing data
B. Prepare-Learning: Specification of an initial model, preparation of training, selection of algorithms, and training parameters
C. Model-Learning: Training of the actual model, monitoring, and supervision
D. Evaluate-Model: Apply the model to testing data, selecting and analyzing quality metrics, understanding the model

We are interested in highlighting steps and tasks that are specific and critical to reinforcement and imitation learning, and which have not been captured previously by more generic workflows. Figure 1 summarizes our proposed workflow described in this chapter. We further present more details on the relevant stages:

Setup and Design of the Environment Instead of preparing a dataset, the first step for reinforcement learning is specifying the environment. This is also necessary for many variants of imitation learning, that require access to an underlying environment for training and/or evaluation. Such an environment specification can be seen as related to the task of data cleaning and transformation, specifically feature engineering, found in supervised learning. This stage includes four/five distinct steps:

1. Designing the *Environment*: Designing, Implementing, and Debugging: Designing either a simulation environment or interface to the real world. The environment should model the desired problem as accurately as necessary to solve the specified task.
2. *Defining the observation space*: Define the agent's interface to perceive the environment and act in it. The observation space can consist of arbitrary numeric input (e.g., images, signals).
3. *Defining the action space*: The possible interactions of an agent with the environment. The specific problem sometimes restricts the action space, i.e., the available actions are determined by the agent and the task to solve. Generally, we have significant freedom in how to encode the actions (e.g., categorical or continuous, flat or complex nested actions).

4.* *Providing the dataset of expert demonstrations*: A step unique to imitation learning is the provision of a dataset for the agent. It is helpful to understand what is contained in the dataset, e.g., to identify if there is sufficient variety in the dataset to capture the desired task. The user, therefore, should be able to query and visualize the trajectory data contained in the dataset.

For the spaces, possible errors include incorrect values, cut-off points (e.g., premature episode termination), faulty scaling or processing of values. In imitation learning, defining the observation space corresponds to the shape of the data fed to the algorithm during training. Ensuring the consistency of these interfaces is a major difficulty, so we propose versatile **inspection methods** to analyze value ranges, dimensions, or effects of actions. Such inspection possibilities are particularly useful for custom environments subjected to rapid prototyping and evaluation.
In reinforcement learning, there is another crucial step:

5.* *Designing the reward function*: Reward design is both an essential and difficult step for reinforcement learning. A scalar reward

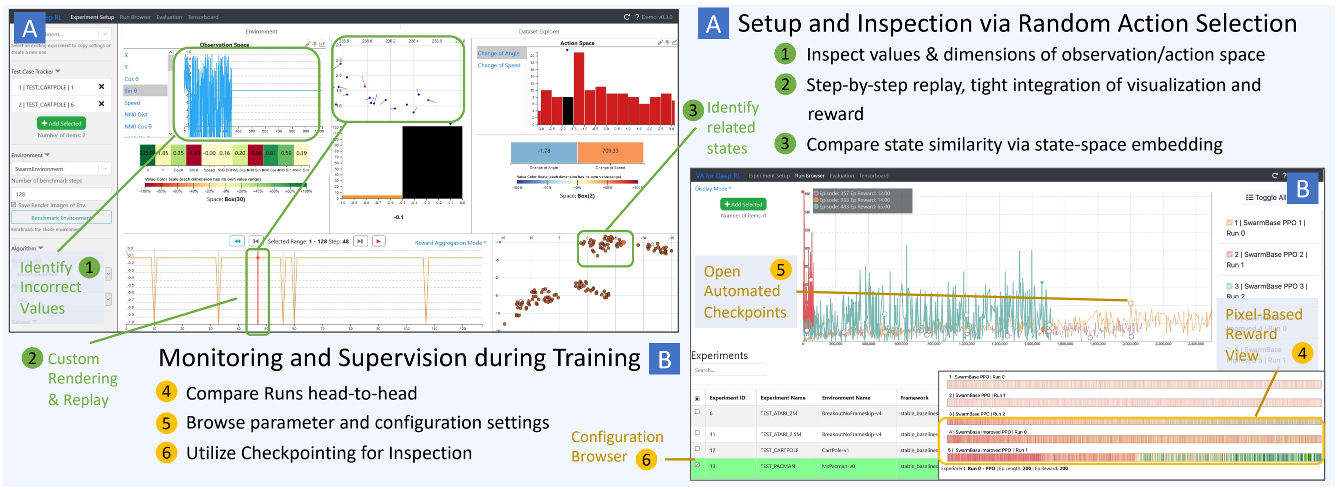
A Setup and Inspection via Random Action Selection
1 Inspect values & dimensions of observation/action space
2 Step-by-step replay, tight integration of visualization and reward
3 Compare state similarity via state-space embedding
3 Identify related states
1 Identify Incorrect Values
2 Custom Rendering & Replay
Monitoring and Supervision during Training B
4 Compare Runs head-to-head
5 Browse parameter and configuration settings
6 Utilize Checkpointing for Inspection
Configuration Browser 6
5 Open Automated Checkpoints
Pixel-Based Reward View 4
Experiments
Observation Space
Action Space

Figure 2: *The implementation of the RIVA workflow applied to swarm reinforcement and imitation learning. UI elements are linked to the respective steps of the workflow.* ***A*** *show the Setup View page with three exemplary interactions: (1) The user analyzes values passed as observations and spots violations of bounds and faulty values, (2) looks at the environment with a custom, interactive rendering view, and linked controls, (3) analyzes and selects related states via the state embedding.* ***B*** *shows the run browser, where the user can (4) use a pixel-based view displaying episode reward over training to quickly analyze many different runs, (5) Browse parameter configurations of the runs, and (6) and analyze the agents via the established inspection methods at regular checkpoints during training (not displayed).*

has to be passed to the agent at every step to specify the success of solving a potentially long-term goal.

An overly informative reward function can lead to behavior like reward hacking [Irp18, EH19], i.e., learning behavior that does not contribute to the actual intended goal. Throughout the workflow, reward metrics are tightly coupled with replay mechanics of the agent behavior in the environment to spot behavior like reward hacking. On the other hand, during random exploration, an agent must receive non-zero feedback at least at some timesteps to learn. To facilitate this issue, we propose testing with initial random action selection to investigate the explorability of initial strategies.

Model Training Choices during training setup have to be carefully tracked and analyzed:

6. Choose the *algorithm*: There are many different algorithms in the space of deep reinforcement/imitation learning, which can be adapted to specific problems.
7. Choosing *Neural Network Architectures*: In the context of deep learning, choosing an appropriate network type (e.g., Feed-Forward Network vs. LSTM) and hyperparameters like layer size continue to be essential considerations.
8. Choosing algorithm *Hyper Parameters*: Compare to supervised deep learning, deep reinforcement learning is very susceptible to the choice of hyperparameters, which can hugely affect training performance [ARS*20, HIB*17].

By recording these choices during the design and training phase need, users can investigate their effect on training performance. Therefore, instantiations need to enable automated **experiment tracking**. Results are linked with the chosen settings in a searchable, comparable way.

Supervise the Learning Process Dynamic supervision of training:

9. *Monitoring* the training progress: Experiments in RL can be very computationally demanding and take a long time. Monitoring training statistics can indicate progress and can avoid spending resources on unsuccessful training.

Monitoring should be facilitated at different scales, and encompass both performance metrics and the interactive inspection of agent behavior during training, e.g., via a checkpointing mechanism.

Evaluation and Understanding The final step of the experimentation workflow is the evaluation of a trained agent. Particularly, our workflow encompasses four dedicated **evaluation tools**:

10. *Comparison against a baseline*: A widespread measure to evaluate the performance of an agent is to compare its achieved reward against a baseline, e.g., a human expert or an existing algorithm.
11. Testing in a *validation environment*: Users must be able to test agent performance in separate validation environments.
12. Probe behavior in specific *test cases*: **Provenance Tracking**, at each stage of the process, enables to later test the behavior of an agent in a domain-specific situation of interest.
13. *Understand the agent behavior* via **XAI methods**. Crucially, the user must be able to investigate, e.g., points of failure in the agent's behavior.

Many of the presented mechanics, like monitoring, provenance tracking, and interactive rendering, are also highly important for the possible phase of **Utilization**. Embedding deployed agents in an interactive visual analytics-based framework helps users to apply and supervise RL-based agents in real-world scenarios.

3. RIVA: Reinforcement and Imitation Learning with VA

Based on the proposed workflow, we present RIVA: An Integrated Workflow for **R**einforcement and **I**mitation Learning with **V**isual **A**nalytics. RIVA operationalizes the proposed workflow via a stand-alone web-based application. RIVA is tightly integrated with existing community-driven frameworks for reinforcement learning (*StableBaselines3* [RHE*19]) and imitation learning (*Imitation* [WTGE20]), both widely used throughout the RL research community. Furthermore, more generic experiment tracking tools like *Tensorboard* are also integrated as separate tabs.
Design decisions for the tools are driven by the presented workflow. Additionally, our goal is to support the mentioned libraries with minimal modifications of the respective code bases. The application is kept use-case agnostic and highly modular to accommodate a wide range of use cases. We present one such use case and respective modular extensions.

During **Setup** and **Debugging**, we utilize a combined view of environment rendering, and inspection views of the observation and action spaces, as well as the reward distribution. A state embedding serves as an additional linked view to quickly identify related states that the agent encounters. The setup view of RIVA allows analyzing the environment step-by-step via fine-grained controls, which enables to carefully assess the effect of actions taken in the environment. During this initial phase, actions are performed by a random agent, with the possibility to, e.g., use a hand-crafted baseline agent. At each step, the dimensions of vector-valued observations can be carefully analyzed individually, e.g., to determine violations of the pre-determined maximum- and minimum value bounds that have to be specified before running an experiment. The inspection tools provide user-controllable line charts and histograms to analyze each individual dimension. Figure 2-**A** shows some possible interactions in the setup view. The application allows saving different configurations of the environment, which can be later analyzed to enable systematic tracking of the effect of design choices on the performance of agent training. As in later stages, users can add each individual step as a test case for a detailed comparison of agents in controlled situations.

To support **model training**, the application provides an interface to choose settings and hyperparameters, with parameter descriptions automatically generated from the documentation. The tool enables simple comparison to other hyperparameter configurations used for different runs. A table summarizes experiment configurations and parameter settings. The table is linked to an interactive line chart or pixel-based visualization which shows agent performance during training. The pixel-based visualization provides more fine-grained insight into the training results on an episode-by-episode basis. The user can choose the normalization of the rewards scale and episode length in the pixel-based view to enable head-to-head comparison between runs. A check-pointing mechanism enables direct insight into the agent's behavior at different points during training. For each checkpoint, a widget provides detailed rendering and inspection. This lets the user understand the resulting behavior at specific reward values. Figure 2-**B** shows the *Experiment Browser* view.

Finally, **Evaluation** is facilitated in a separate evaluation view. The view includes tools to interactively test and compare models across different environments. Models saved at checkpoints can be further analyzed to track the change of a policy during model training. Specific user-supplied validation environments can be selected for benchmarking to test properties like robustness and generalization. The evaluation view also natively supports input attribution methods (e.g., *IntegratedGradients* [STY17]), which can display attributions for both vector and image-based observations. As a final component, the evaluation view contains a list of previously saved instances, which can be pinned at any time during setup, training, and evaluation. This enables the user to directly compare the performance of different agents in controlled situations.

4. Applied Imitation Learning for Collective Behavior

We apply the proposed framework and developed an application in the use case of imitation and reinforcement learning for collective behavior: data-driven learning of the behavior of fish schools (collective movement of fish swarms). We cooperated with a domain expert throughout the entire process, from designing custom environments and agents, training, to final evaluation. Modeling the behavior of individual actors in swarm systems has been a long-standing problem in biology [Rey87, Sum06, CLN*13]. Learning individual policies that lead to coordinated collective behavior via both reinforcement learning, and imitation learning from recorded trajectories, is an exciting application that promises to overcome existing simplifications in hand-crafted models.

The use case can be well integrated into our workflow and application with minimal modifications. Noticeably, a custom interactive rendering of the environment was added (see Figure 2A.**2**). We utilize the modularity of the software to integrate additional components like custom visualizations. During the design phase, the inspection views were used to ensure consistency between environment, agent, and dataset, e.g. to spot premature episode termination (see Fig. 2A.1). Our workflow was highly effective in maintaining a high level of productivity and consistency through an iterative design process, in which we experimented with different observation space designs, reward functions, network types, and hyperparameter configurations. The set of evaluation tools is used both for internal evaluation and external presentation.

5. Conclusion

We presented RIVA, a comprehensive workflow for RL and IL using VA concepts, and an application to showcase the proposed workflow on an applied use case. We argue that a holistic approach to RL/IL training can enable better and more impactful development and research. Not fully addressing the challenges posed by reinforcement and imitation learning can have negative consequences of hindering progress, potentially damaging the reputation of these methods, and decreasing trust from potential users. Future work could refine the visual presentation of the workflow. A specific focus could be given to advanced provenance and change tracking to trace the effect of design changes made during the process.

Acknowledgement

This work was supported by the German Research Foundation as part of the priority programme "Volunteered Geographic Information: Interpretation, Visualisation and Social Computing" (VGIscience, priority programme 1894)

References

[ACD*15] AMERSHI S., CHICKERING M., DRUCKER S. M., LEE B., SIMARD P., SUH J.: Modeltracker: Redesigning performance analysis tools for machine learning. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (New York, NY, USA, 2015), CHI '15, Association for Computing Machinery, p. 337–346. URL: https://doi.org/10.1145/2702123.2702509, doi:10.1145/2702123.2702509. 1

[Ach18] ACHIAM J.: Spinning Up in Deep Reinforcement Learning. https://spinningup.openai.com/en/latest/spinningup/spinningup.html, 2018. Accessed: 2022-02-26. 2

[ALA*18] ANDRIENKO N., LAMMARSCH T., ANDRIENKO G., FUCHS G., KEIM D., MIKSCH S., RIND A.: Viewing visual analytics as model building. In *Computer graphics forum* (2018), vol. 37, Wiley Online Library, pp. 275–299. 1

[ARS*20] ANDRYCHOWICZ M., RAICHUK A., STAŃCZYK P., ORSINI M., GIRGIN S., MARINIER R., HUSSENOT L., GEIST M., PIETQUIN O., MICHALSKI M., GELLY S., BACHEM O.: What Matters In On-Policy Reinforcement Learning? A Large-Scale Empirical Study. *arXiv e-prints* (June 2020), arXiv:2006.05990. arXiv:2006.05990. 3

[BCP*16] BROCKMAN G., CHEUNG V., PETTERSSON L., SCHNEIDER J., SCHULMAN J., TANG J., ZAREMBA W.: Openai gym, 2016. arXiv:arXiv:1606.01540. 2

[BNVB13] BELLEMARE M. G., NADDAF Y., VENESS J., BOWLING M.: The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research 47* (jun 2013), 253–279. 1

[CLN*13] CALOVI D., LOPEZ U., NGO S., SIRE C., CHATÉ H., THERAULAZ G.: Swarming, schooling, milling: Phase diagram of a data-driven fish school model. *New Journal of Physics 16* (08 2013). doi:10.1088/1367-2630/16/1/015026. 4

[DLM*20] DULAC-ARNOLD G., LEVINE N., MANKOWITZ D. J., LI J., PADURARU C., GOWAL S., HESTER T.: An empirical investigation of the challenges of real-world reinforcement learning. *arXiv e-prints* (Mar. 2020), arXiv:2003.11881. arXiv:2003.11881. 1

[EH19] EVERITT T., HUTTER M.: Reward tampering problems and solutions in reinforcement learning: A causal influence diagram perspective. *CoRR abs/1908.04734* (2019). URL: http://arxiv.org/abs/1908.04734, arXiv:1908.04734. 3

[ERT*18] ENDERT A., RIBARSKY W., TURKAY C., WONG W., NABNEY I., DÍAZ BLANCO I., ROSSI F.: The State of the Art in Integrating Machine Learning into Visual Analytics. *arXiv e-prints* (Feb. 2018), arXiv:1802.07954. arXiv:1802.07954. 1

[HIB*17] HENDERSON P., ISLAM R., BACHMAN P., PINEAU J., PRECUP D., MEGER D.: Deep Reinforcement Learning that Matters. *32nd AAAI Conference on Artificial Intelligence, AAAI 2018* (sep 2017), 3207–3214. URL: http://arxiv.org/abs/1709.06560, arXiv:1709.06560. 1, 3

[HLv*20] HE W., LEE T., VAN BAAR J., WITTENBURG K., SHEN H.: Dynamicsexplorer: Visual analytics for robot control tasks involving dynamics and lstm-based control policies. In *2020 IEEE Pacific Visualization Symposium (PacificVis)* (2020), pp. 36–45. doi:10.1109/PacificVis48177.2020.7127. 1

[HRE*18] HILL A., RAFFIN A., ERNESTUS M., GLEAVE A., KANERVISTO A., TRAORE R., DHARIWAL P., HESSE C., KLIMOV O., NICHOL A., PLAPPERT M., RADFORD A., SCHULMAN J., SIDOR S., WU Y.: Stable baselines. https://github.com/hill-a/stable-baselines, 2018. 2

[Irp18] IRPAN A.: Deep reinforcement learning doesn't work yet. https://www.alexirpan.com/2018/02/14/rl-hard.html, 2018. 2, 3

[LSL*16] LIU M., SHI J., LI Z., LI C., ZHU J., LIU S.: Towards better analysis of deep convolutional neural networks. *CoRR abs/1604.07043* (2016). URL: http://arxiv.org/abs/1604.07043, arXiv:1604.07043. 1

[LWLZ17] LIU S., WANG X., LIU M., ZHU J.: Towards better analysis of machine learning models: A visual analytics perspective. *CoRR abs/1702.01226* (2017). URL: http://arxiv.org/abs/1702.01226, arXiv:1702.01226. 1

[M*22] MANDHANE A., ET AL.: MuZero with Self-competition for Rate Control in VP9 Video Compression. *arXiv e-prints* (Feb. 2022), arXiv:2202.06626. arXiv:2202.06626. 1

[MGL21] MARTÍN-GUERRERO J. D., LAMATA L.: Reinforcement learning and physics. *Applied Sciences 11*, 18 (2021). URL: https://www.mdpi.com/2076-3417/11/18/8589, doi:10.3390/app11188589. 1

[NL19] NGUYEN H., LA H.: Review of deep reinforcement learning for robot manipulation. In *2019 Third IEEE International Conference on Robotic Computing (IRC)* (2019), pp. 590–595. doi:10.1109/IRC.2019.00120. 1

[Rey87] REYNOLDS C. W.: Flocks, herds and schools: A distributed behavioral model. In *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1987), SIGGRAPH '87, Association for Computing Machinery, p. 25–34. URL: https://doi.org/10.1145/37401.37406, doi:10.1145/37401.37406. 4

[RHE*19] RAFFIN A., HILL A., ERNESTUS M., GLEAVE A., KANERVISTO A., DORMANN N.: Stable baselines3. https://github.com/DLR-RM/stable-baselines3, 2019. 2, 4

[SKKC19] SACHA D., KRAUS M., KEIM D. A., CHEN M.: VIS4ML: an ontology for visual analytics assisted machine learning. *IEEE Trans. Vis. Comput. Graph. 25*, 1 (2019), 385–395. URL: https://doi.org/10.1109/TVCG.2018.2864838, doi:10.1109/TVCG.2018.2864838. 1, 2

[SPBA19] SALDANHA E., PRAGGASTIS B., BILLOW T., ARENDT D.: ReLVis : Visual Analytics for Situational Awareness During Reinforcement Learning Experimentation. In *EuroVis (Short Papers)* (2019), Eurographics Association, pp. 43–47. doi:10.2312/evs.20191168. 1

[SSSE20] SPINNER T., SCHLEGEL U., SCHÄFER H., EL-ASSADY M.: explainer: A visual analytics framework for interactive and explainable machine learning. *IEEE Transactions on Visualization and Computer Graphics 26*, 1 (2020), 1064–1074. doi:10.1109/TVCG.2019.2934629. 1

[STY17] SUNDARARAJAN M., TALY A., YAN Q.: Axiomatic attribution for deep networks. *CoRR abs/1703.01365* (2017). URL: http://arxiv.org/abs/1703.01365, arXiv:1703.01365. 4

[Sum06] SUMPTER D.: The principles of collective animal behavior. *Philosophical transactions of the Royal Society of London. Series B, Biological sciences 361* (02 2006), 5–22. doi:10.1098/rstb.2005.1733. 4

[WGSY19] WANG J., GOU L., SHEN H. W., YANG H.: DQNViz: A Visual Analytics Approach to Understand Deep Q-Networks. *IEEE Transactions on Visualization and Computer Graphics 25*, 1 (2019), 288–298. doi:10.1109/TVCG.2018.2864504. 1

[WTGE20] WANG S., TOYER S., GLEAVE A., EMMONS S.: The *imitation* library for imitation learning and inverse reinforcement learning. https://github.com/HumanCompatibleAI/imitation, 2020. 4

[ZBP18] ZHANG A., BALLAS N., PINEAU J.: A dissection of overfitting and generalization in continuous reinforcement learning. *CoRR abs/1806.07937* (2018). URL: http://arxiv.org/abs/1806.07937, arXiv:1806.07937. 1