An Architecture for Reliable Learning Agents in Power Grids

Eric MSP Veith
Carl von Ossietzky University Oldenburg
Research Group Adversarial Resilience Learning
Oldenburg, Germany
Email: eric.msp.veith@uol.de

***Abstract*—Agent systems have become almost ubiquitous in smart grid research. Research can be roughly divided into carefully designed (multi-) agent systems that can perform known tasks with guarantees, and learning agents based on technologies, such as Deep Reinforcment Learning (DRL), that promise real resilience by learning to counter the unknown unknowns. However, the latter cannot give guarantees regarding their behavior, while the former are limited to the set of problems known at design time. This paper presents a hybrid architecture that enables a learning agent to give guarantees about its behavior, making it suitable for usage in Critical National Infrastructures (CNIs), such as the power grid.**

***Keywords*—agent systems; reinforcement learning; trustworthy AI; resilience; power grid**

I. Introduction

Over the last years, agent systems and especially Multi-Agent Systems (MASs) [1]–[4] have emerged as one of the important tools to facilitate management of complex energy systems. As swarm logic, they can handle numerous tasks, such as maintaining real power equilibria, voltage control, or automated energy trading [5]. The fact that MASs implement proactive and reactive distributed heuristics allows to analyze their behavior and give certain guarantees, a property that has helped in their deployment.

However, modern energy systems have also become valuable targets. Cyber-attacks have become more common [6], [7], and establishing local energy markets, although being an attractive concept of self-organization, can also be to manipulation, e. g., through artificially created congestion [8]. Attacks on power grids are no longer carefully planned and executed, but also learned by agents, such as market manipulation or voltage band violations [9]. Thus, carefully designing software systems that provide protection against a widening field of adversarial scenarios have become a challenge, especially considering that complex, inter-connected Cyber-Physical Systems (CPSs) are inherently exploitable due to their complexity itself [10].

Learning agents, particularly those based on DRL, have gained traction as a potential solution: If a system faces unknown unknowns, a learning agent can devise strategies against it. In the past, researchers have published using DRL-based agents for numerous tasks related to power grid operations—e. g., voltage control [11]—, but the approach to use DRL for general resilient operation is relatively new [12], [13]. DRL—the notion of an agent with sensors and actuators that learns by "trial and error"—is at the core of many noteworthy successes, such as MuZero [14], with modern algorithms such as Twin-Delayed DDPG (TD3) [15], Proximal Policy Gradient (PPO) [16], and Soft Actor Critic (SAC) [17] having proved to be able to tackle complex tasks.

All modern DRL use deep Artificial Neural Networks (ANNs) at least for the policy (or multiple, e. g., for the critic). Actual parameter optimization is commonly done with gradient descent algorithms. However, these ANNs' architectures still need to be provided by the user, in addition the hyperparameters of the algorithm. No DRL agent is, therefore, a "deploy and forget" approach; careful tuning is usually required for a specific use case. Evolving these networks, or using genetic or evolutionary algorithms as an alternative entirely, has gained interest among scientist during the last years [18], [19].

However, these model-free algorithms themselves cannot give guarantees with regard to their behavior, which is important for deployment with high autonomy in any CNI. *Safe DRL* algorithms target this research gap, but currently, they learn inefficiently or explore insufficiently [20]. Moreover, Safe DRL does not (yet) tackle changes in tasks or environment, the problem of *Online Learning* [21].

In a very similar vein, learning agents for CNIs not only need to give guarantees, but they must also offer introspection. I. e., their strategies must be inspectable by humans. In the context of CNIs, this allows audition or certification, or testing and validation even at runtime. Finally, it is an important factor for acceptance. This introspection is provided by techniques of eXplainable Reinforcement Learning (XRL) [22]. However, the most common techniques, such as saliency maps, give only indirect interpretation and are useful for experts in the DRL domain, but not for practitioners in CNIs. Recent approaches to convert a DRL agent's policy network into a rule-based representation, e. g., as decision tree [23], will satisfy the outlined requirements, but are not part of an DRL agent architecture yet.

Therefore, we identify the following research gap: Learning agents are necessary for modern, complex CNIs, such as the power grid. In order to cope with the complexity of the power grid, changing actor behavior and, thus, changing marginal distributions, online learning must be explicitly considered. In addition, such a learning agent must be inspectable and provide guarantees. To this end, we propose a hybrid architecture in the Adversarial Resilience Learning (ARL) agent. It combines a learning agent with a rule-based agent. The learning agent's policy is constantly converted into rule sets, represented in the Boolean domain, in order to enable the benefits of XRL.

 ISBN: 978-1-68558-054-4

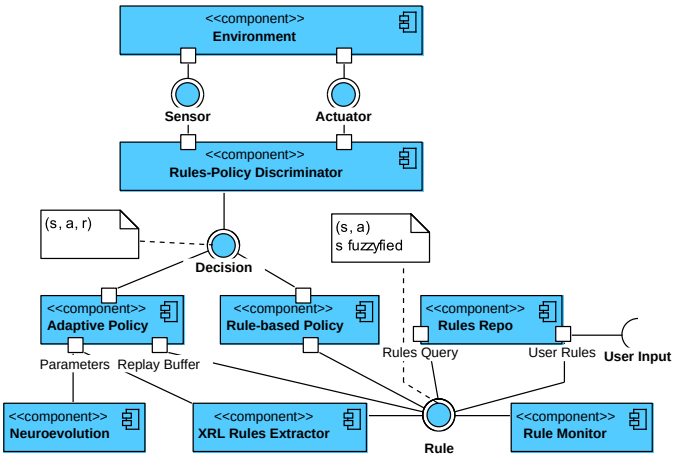

Figure 1. ARL Agent Architecture

However, the agent is also able to absorb obsolete rules when a better strategy is (reliably) devised, in order to maintain the agent's adaptability. Through this cycle of rule extraction (XRL) and rule consumption (online learning), the agent stays inspectable and validatable, providing the necessary security for deployment in CNIs, while still being able to learn and counter the unknown unknowns.

The remainder of this work-in-progress paper is structured as follows: Section II outlines the planned architecture and gives the rationale for each module, as well as their interfaces. In Section III, we outline our testing scenarios, which serve as overall validation of the architecture. Finally, we outline the next steps in Section IV.

II. Agent Architecture and Modules

The key design goal of the ARL agent architecture is to fuse a learning component based on DRL with the analyzability of a rules-based architecture. Therefore, an *Adaptive Policy* based on DRL, as well as a *Rule-based Policy*, are employed alongside each other. The DRL-based architecture will resort to off-policy algorithms, such as TD3 [15] and SAC [17]. This will simplify the extended usage of replay buffers, e. g., for rehearsal or offline learning.

This design iteration of the architecture does not consider MASs, therefore, the design follows more the premises of DRL-based agents and eschews the usual message inbox, journal, etc. that would normally be present for agents of a MAS. The resulting component design is depicted in Figure 1.

The agent receives data from the environment using *Senors*, which constitutes the agent's world state at time t: s_t. A sensor is a simple software interface that transfers sensor readings according to a simple mathematical space definition that allows for Boolean values, intervals of discrete numbers, n-dimensional real-valued spaces, and more complex types created as combinations thereof. We loosely follow the example of OpenAI Gym [24] and use the definition that we introduced in prior works [25].

Sensor readings are fed into the *Rules-Policy Discriminator*. This module is responsible for choosing the agent's actions, which stem either from the adaptive policy or the rule-based one. The discriminator gives precedence to behavior emitted from the rules-based policy, thus ensuring well-defined behavior whenever possible. If the rule-based policy cannot emit an action given the current sensor readings, the discriminator uses the action proposed by the adaptive policy. However, even if a matching rule can be emitted from the rule-based policy, the adaptive policy is still queried for an action. This way, $(s_t, a_t, \hat{r}_t)$ triplets can be fed to the replay buffer. The adaptive policy's action is preferred if it, checked against a world model, provides a higher reward. The triplet of state s_t—i. e., the current sensor readings—, the planned action a_t of the agent, as well as the reward $\hat{r}_t$ the agent expects, is an agent's *Decision*. The discriminator also propagates the actual reward from the last action, r_{t-1}, to the policies in order to update their replay buffer or rules assessment (adaptive and rule-based policy, respectively).

The rule-based policy uses a *Rules Repository* for storage. The repository uses Ternary Vector Lists (TVLs) [26] for efficient storage. TVLs represent systems of boolean equations as lists of disjoint vectors, where each vector represents the assignment of variables. In contrast to binary vectors, ternary vectors provide efficient storage by introducing a third symbol, such that a ternary value is defined as $tv \in \{0, 1, -\}$. The dash expands to both 0 and 1, so that the ternary vector $[1, -]^\top$ expands to two binary vectors, $[1, 0]^\top$ and $[1, 1]^\top$. Set operations on TVLs are well-defined and expand to the appropriate operations on Boolean values. The ARL agent will expand the notion of TVLs fuzzy logic in order to allow rules-based inference of actions based on sensor readings when those readings will seldom provide the exact values in $\mathbb{R}^n$ that are given in the (Boolean) rule sets.

A *Rules Monitor* supervises the repository: The agent still needs to be adaptive and learn, i. e., develop new strategies to situations unknown at design time. Therefore, the monitor will feed rules of known sensor readings back to the DRL policy's replay buffer. Should the adaptive policy have devised better rules, then this is merely the same problem off-policy DRL algorithms solve with their replay buffer, which contains triplets of $(s, a_1, r_1), (s, a_2, r_2) : r_1 < r_2$ and is a simple optimization problem. In the case of catastrophic forgetting [21] during online learning, the rules repository serves as a rehearsal device [21].

The combination of rules repository and rules monitor serves the training aspect of the agent: The adaptive policy will be trained using *Neuroevolution*, i. e., the ANNs are evolved during training, their architecture not provided beforehand. Usually, employing neuroevolutionary strategies reduces the sample efficiency considerably. However, the rules repository-monitor modules are able to serve as extended replay buffer, allowing for extended iteration over samples to use them during neuroevolution.

The rules repository is the central piece that allows introspection of the agent's policy, i. e., behavior, and thus aids its interpretability. It interfaces to the *XRL Rules Extractor*, which takes care of generating rules from the adaptive policy. This happens in two ways: First, given any $(s_t, a_t, \hat{r}_t)$ triplets

 ISBN: 978-1-68558-054-4

```
1: procedure ACT(s_t: List[SensorReading], r_{t-1})
2:     UPDATEREWARD(rulePolicy, r_{t-1})
3:     UPDATEREWARD(adaptivePolicy, r_{t-1})
4:     decision_t^(rule) ← DECIDE(rulePolicy, s_t)
5:     decision_t^(adaptive) ← DECIDE(adaptivePolicy, s_t)
6:     if ¬decision_t^(rule) then
7:         a_t ← decision_t^(adaptive).a
8:     end if
9:     if decision_t^(rule) then
10:        a_t ← decision_t^(rule).a
11:        if decision_t^(rule).r < decision_t^(adaptive).r ∧
   decision_t^(rule).a ≈ decision_t^(adaptive).a then
12:            STORE(replayBuffer, decision_t^(rule))
13:        end if
14:        if decision_t^(rule).r < decision_t^(adaptive).r ∧
   decision_t^(rule).a ≠ decision_t^(adaptive).a then
15:            a_t ← decision_t^(adaptive).a
16:        end if
17:    end if
18:    ONLINETRAINANDEVOLVE(adaptivePolicy)
19:    rule ← TORULE(adaptivePolicy, decision_t^(adaptive))
20:    STORE(rulesRepo, rule)
21:    return a_t
22: end procedure
```

Figure 2. Agent Act Routine

from the adaptive policy, a corresponding rule is created by treating the adaptive policy as a black box. Furthermore, the policy ANN is converted into a decision tree [23] and this decision tree is examined for yet-uncovered rules. Thus, the rules extractor constantly feeds rules to the repository, which are ranked according to $(s, a_1, r_1) > (s, a_2, r_2) : r_1 > r_2$.

This completes a two-way interface between the adaptive and the rule-based policy. The rule-based policy yields most of the agent behavior in *learned* situations, allowing to give guarantees with regards to the agent's actions and strategy. The adaptive policy learns to cope with yet unknown situations. At the same time, behavior learned by the adaptive policy is immediately converted into rules, allowing for introspection and extending the behavior governed by guarantees. The rule-based policy is then also able to solve behavioral conflicts. Additionally, rules can be obsoleted by better strategies found by the adaptive policy, which is why sensor readings are always fed to both policies. The rule monitor identifies these obsoleted rules and removes them from the rules repository.

Figure 2 cover the description in this section.

III. Discussion

Obviously, the proposition of the ARL agent architecture is a bold one. Therefore, careful experimentation with benchmark scenarios must be conducted in order to verify the hypothesis underlying the architecture.

Scenario 1 considers voltage regulation as a basic use case. The agent should be able to learn to keep the voltage close to $1.0\,\mathrm{p\,u}$ in a medium voltage grid, such as the CIGRÉ MV grid. The proposition should hold under time series (i. e., time series data for Photovoltaic (PV) and wind power feed-in, as well as time-series-based customer consumption), as well as under grid constraints (i. e., grid codes). In order to master scenario 1, the ARL agent should be able to cope with the given situations at least as well as, or better, than simple Volt/VAr controllers, as well as simple DRL agents. We expect the rules repository to contain rules similar to that of simple reactive power controllers.

In *Scenario 2*, the agent must cope successfully with changing marginal distributions, such as the introduction of Virtual Power Plants (VPPs) or changing customer behavior. This tests the online learning capabilities of the design. The task is still keeping the voltage close to $1.0\,\mathrm{p\,u}$. Again, the rules repository will serve as an indicator for the quality of strategies learned, accompanied by the usual DRL metrics, such as reward, objective value, and entropy.

In *Scenario 3*, the ARL agent must succeed against a simple attacker, such as the oscillating attacker by Ju *et al.* [27] or other documented forms of attack. If the ARL agent really constitutes a better concept than the pure DRL approach, then it will not just be able to counter the attack, it will also succeed against different attack strategies. In practice, this is not only another test for the agent's online learning capabilities, but also a way to extract real resilience strategies.

In *Scenario 4*, the ARL agents compete against each other ("attacker" versus "defender"). This is more than just the logical extension of scenario 3: As documented, this forces the agents to sample the extreme areas of the action distribution [12], given a plethora of extractable strategies and documentation of weaknesses of a grid design.

The test scenarios are intended to test the overall behavior of the system: Its ability to adapt through learning, stay interpretable, and give guarantees. Since the research gap addressed by the ARL agent architecture is the combination of learning agent and guaranteed behavior, these scenarios can test the agent by formulating invariants based on expected guarantees.

If successful, we expect the ARL agent to be viable for introduction in grid operator control centers. An initial use case will that of a support and recommender system that helps grid operators to keep situational awareness in complex situations. Later on, the agent can manage parts of the grid (e. g., LV branches with a high number of prosumers) in order to redtce the complexity of grid management. Furthermore, we design the agent to act as a defender against actual cyber attacks.

IV. Conclusion and Future Work

In this paper, we have proposed an agent architecture that fuses rule-based behavior with the learning capabilities of DRL. We did so in order to provide a learning agent that can still give guarantees about its behavior.

In the future, we will develop the respective modules and provide benchmarks and test results, with a special focus on the applicability in CNIs.

 ISBN: 978-1-68558-054-4

ACKNOWLEDGEMENTS

This work was funded by the German Federal Ministry for Education and Research (BMBF) under Grant No. 01IS22071.

REFERENCES

[1] E. M. Veith, *Universal Smart Grid Agent for Distributed Power Generation Management*. Logos Verlag Berlin GmbH, 2017.

[2] E. Frost, E. M. Veith, and L. Fischer, "Robust and deterministic scheduling of power grid actors," in *7th International Conference on Control, Decision and Information Technologies (CoDIT)*, IEEE, IEEE, 2020, pp. 1–6.

[3] M. Sonnenschein and C. Hinrichs, "A distributed combinatorial optimisation heuristic for the scheduling of energy resources represented by self-interested agents," *International Journal of Bio-Inspired Computation*, 2017, ISSN: 1758-0366. DOI: 10.1504/IJBIC.2017.10004322.

[4] O. P. Mahela *et al.*, "Comprehensive overview of multi-agent systems for controlling smart grids," *CSEE Journal of Power and Energy Systems*, vol. 8, no. 1, pp. 115–131, Jan. 2022, Conference Name: CSEE Journal of Power and Energy Systems, ISSN: 2096-0042. DOI: 10.17775/CSEEJPES.2020.03390.

[5] *Flexibility management and provision of balancing services with battery-electricautomated guided vehicles in the Hamburg container terminal Altenwerder*, Energy Informatics, SpringerOpen, 2020. DOI: https://doi.org/10.1186/s42162-020-00129-1.

[6] J. Styczynski and N. Beach-Westmoreland, "When the lights went out: Ukraine cybersecurity threat briefing," *Booz Allen Hamilton*, vol. 12, p. 20, 2016.

[7] A. Aflaki, M. Gitizadeh, R. Razavi-Far, V. Palade, and A. A. Ghasemi, "A hybrid framework for detecting and eliminating cyber-attacks in power grids," *Energies*, vol. 14, no. 18, p. 5823, Jan. 2021, Number: 18 Publisher: Multidisciplinary Digital Publishing Institute, ISSN: 1996-1073. DOI: 10.3390/en14185823. [Online]. Available: https://www.mdpi.com/1996-1073/14/18/5823 (visited on 12/11/2022).

[8] T. Wolgast, E. M. Veith, and A. Nieße, "Towards reinforcement learning for vulnerability analysis in power-economic systems," in *DACH+ Energy Informatics 2021: The 10th DACH+ Conference on Energy Informatics*, Freiburg, Germany, Sep. 2021.

[9] E. M. Veith, N. Wenninghoff, S. Balduin, T. Wolgast, and S. Lehnhoff, *Learning to attack powergrids with ders*, 2022. DOI: 10.48550/ARXIV.2204.11352. [Online]. Available: https://arxiv.org/abs/2204.11352.

[10] O. Hanseth and C. Ciborra, *Risk, complexity and ICT*. Cheltenham, UK: Edward Elgar Publishing, 2007.

[11] R. Diao *et al.*, "Autonomous voltage control for grid operation using deep reinforcement learning," in *2019 IEEE Power & Energy Society General Meeting (PESGM)*, Atlanta, GA, USA: IEEE, Aug. 2019, pp. 1–5, ISBN: 978-1-72811-981-6. DOI: 10.1109/PESGM40551.2019.8973924.

[12] L. Fischer, J. M. Memmen, E. M. Veith, and M. Tröschel, "Adversarial resilience learning—towards systemic vulnerability analysis for large and complex systems," in *ENERGY 2019, The Ninth International Conference on Smart Grids, Green Communications and IT Energy-aware Technologies*, Athens, Greece: IARIA XPS Press, 2019, pp. 24–32, ISBN: 978-1-61208-713-9.

[13] E. Veith, L. Fischer, M. Tröschel, and A. Nieße, "Analyzing cyber-physical systems from the perspective of artificial intelligence," in *Proceedings of the 2019 International Conference on Artificial Intelligence, Robotics and Control*, ACM, Dec. 2019, ISBN: 978-1-4503-7671-6.

[14] J. Schrittwieser *et al.*, "Mastering Atari, Go, Chess and Shogi by planning with a learned model," pp. 1–21, 2019. arXiv: 1911.08265. [Online]. Available: http://arxiv.org/abs/1911.08265.

[15] S. Fujimoto, H. van Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," *arXiv:1802.09477 [cs, stat]*, Oct. 22, 2018. arXiv: 1802.09477. [Online]. Available: http://arxiv.org/abs/1802.09477 (visited on 12/22/2021).

[16] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," Jul. 19, 2017. arXiv: 1707.06347. [Online]. Available: http://arxiv.org/abs/1707.06347.

[17] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," *arXiv:1801.01290 [cs, stat]*, Aug. 8, 2018. arXiv: 1801.01290. [Online]. Available: http://arxiv.org/abs/1801.01290 (visited on 12/22/2021).

[18] F. P. Such *et al.*, "Deep neuroevolution: Genetic algorithms are a competitive alternative for training deep neural networks for reinforcement learning," 2017. arXiv: 1712.06567. [Online]. Available: http://arxiv.org/abs/1712.06567.

[19] M. I. Radaideh *et al.*, *NEORL: NeuroEvolution optimization with reinforcement learning*, Dec. 1, 2021. DOI: 10.48550/arXiv.2112.07057. arXiv: 2112.07057[cs]. [Online]. Available: http://arxiv.org/abs/2112.07057 (visited on 12/11/2022).

[20] R. Cheng, G. Orosz, R. M. Murray, and J. W. Burdick, "End-to-end safe reinforcement learning through barrier functions for safety-critical continuous control tasks," *33rd AAAI Conference on Artificial Intelligence, AAAI 2019, 31st Innovative Applications of Artificial Intelligence Conference, IAAI 2019 and the 9th AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019*, pp. 3387–3395, 2019, ISSN: 2159-5399. DOI: 10.1609/aaai.v33i01.33013387. arXiv: 1903.08792.

[21] K. Khetarpal, M. Riemer, I. Rish, and D. Precup, *Towards continual reinforcement learning: A review and perspectives*, Dec. 24, 2020. DOI: 10.48550/arXiv.2012.13490. arXiv: 2012.13490[cs]. [Online]. Available: http://arxiv.org/abs/2012.13490 (visited on 10/14/2022).

[22] E. Puiutta and E. M. S. P. Veith, "Explainable reinforcement learning: A survey," in *Machine Learning and Knowledge Extraction. CD-MAKE 2020*, Dublin, Ireland: Springer, Cham, 2020, pp. 77–95. DOI: 10.1007/978-3-030-57321-8_5.

[23] C. Aytekin, *Neural networks are decision trees*, Oct. 25, 2022. DOI: 10.48550/arXiv.2210.05189. arXiv: 2210.05189[cs]. [Online]. Available: http://arxiv.org/abs/2210.05189 (visited on 11/09/2022).

[24] G. Brockman *et al.*, *OpenAI gym*, 2016. eprint: arXiv:1606.01540.

[25] E. M. Veith *et al.*, "Analyzing power grid, ICT, and market without domain knowledge using distributed artificial intelligence," in *CYBER 2020, The Fifth International Conference on Cyber-Technologies and Cyber-Systems*, IARIA, IARIA, 2020, pp. 86–93.

[26] C. Posthoff and B. Steinbach, "Extremely complex problems: Binary models for computer science," in Jan. 2019, pp. 339–360, ISBN: 978-3-030-02419-2. DOI: 10.1007/978-3-030-02420-8_8.

[27] P. Ju and X. Lin, "Adversarial attacks to distributed voltage control in power distribution networks with DERs," in *Proceedings of the Ninth International Conference on Future Energy Systems*, ACM, 2018, pp. 291–302, ISBN: 9781450357678. DOI: 10.1145/3208903.3208912.