WASP Software Engineering Assignment

Franco Ruggeri

November 14, 2025

1 Introduction

I'm an industrial PhD student at Ericsson Research and KTH Royal Institute of Technology. My research is focused on eXplainable Reinforcement Learning (XRL) applied to telecommunication networks [1].

Reinforcement Learning (RL) is a branch of Machine Learning (ML) that involves training an agent to act optimally in an environment in order to optimize a reward signal. Given its closed-loop framework, RL is a promising solution for automating the optimization of large 5G networks, which involves tuning a large number of interconnected parameters. Given the scale of such systems, this optimization cannot be manually manually performed by human operators, and today's commercial solutions rely on rule-based systems that are rigid and suboptimal. RL can instead adapt to dynamic network conditions and optimize performance in real-time.

However, network automation comes with risks, as wrong parameter adjustments can lead to degraded performance and services for end users. For this reason, network operators seek transparency and reliability in such solutions. State-of-the-art RL, similarly to other branches of ML, is notably opaque and lacks the required transparency to be adopted in production environments.

The goal of XRL is to design methods that enhance the interpretability of RL agents so that humans can understand and trust RL-based decisions.

2 Lecture Principles

In the second lecture on *Quality Assurance and Testing in SE*, we discussed principles and techniques for testing in software engineering. An idea that I found inspiring is the usage of Anchors [2] for ML invariant/metamorphic testing. In my opinion, using eXplainable Artificial Intelligence (XAI), or XRL in the specific case of RL models, for automated testing is a promising research direction. In my research experience, I had the opportunity to collaborate with engineering teams to deliver novel RL solutions to customers. During these projects, we used XRL methods to produce plots that were later manually analyzed to extract actionable insights for improving the models. In this context, using XRL methods for automating testing would remove the need for human analysis, speed up iterations, and bring best practices from software engineering. For instance, such automated XRL-based testing would enable continuous integration pipelines to avoid bugs from code changes (e.g., in the training code).

In the fourth lecture on *Science vs Engineering*, we discussed how machine learning is in practice a hybrid between science and engineering. I found the 9 technology readiness levels [3] well-aligned with my research experience. During the past few years, I had the opportunity to work at different levels. In the beginning of my PhD, I experimented with existing methods from the literature, applying them to a telecom use case and even collaborating with engineering teams to deliver production-ready solutions (higher level of technology readiness). Later, I worked on a more fundamental RL problem and devised a novel XRL method (lower level of technology readiness). I also noticed that many researchers at low-level of technology readiness neglect software engineering best practices. In my opinion, instead, high-quality code with strong modularity and testing should always be maintained, not only for high technology readiness levels closer to production and engineering. I believe good software engineering practices can really speed up research and science in the long run. While there is certainly an initial investment, reusable high-quality research software can facilitate replicating results and extending existing ideas. This problem is also related to the crucial need for reproducibility in ML.

3 Guest-Lecture Principles

In the guest lecture on *Requirements Engineering*, Julian Frattini gave an overview of requirements engineering and presented several techniques to conduct it. In relation to my industrial research on XRL, I find the practice of requirements engineering really important to apply within a company, even in research projects. Since industrial research is driven by real business needs rather than only intellectual interest, finding stakeholders and defining their requirements is a crucial activity to define and solving research problems that are of interest for the company.

I also think the idea of incremental refinement of requirements presented in the lecture is really useful and a necessary practice under the high uncertainty typical of research projects. In my research experience, I had the opportunity to communicate with domain experts in telecommunications, which represented stakeholders for my research on XRL. Defining clear, unambiguous explainability needs has been, and continues to be, a challenge. Thus, I believe regular meetings with direct feedback from stakeholders are key to ensure the success of an industrial research project.

4 Data Scientists versus Software Engineers

The first two chapters of [4] present an overview of ML-enabled software products. The author argues that building such products requires the collaboration of data scientists and software engineers. Data scientists are expert in ML models and training algorithms, while software engineers are required to build the overall software product following the engineering process. I definitely agree with this distinction. In my experience, data scientists come from diverse backgrounds that might have very little to do with software engineering. For instance, many successful data scientists have backgrounds in statistics or mathematics. I believe software engineers who understand machine learning are necessary to build ML-enabled software products, as many data scientists are only used to experimentation tools such as Jupyter notebooks and have no experience with production software and infrastructure. Furthermore, delivering high-quality software products require a proper engineering process that includes requirements engineering, design, testing, and maintenance. Data scientists with no software engineering experience are certainly not the right profiles to perform these tasks.

In my opinion, the role of data scientist will evolve and be required to know more about software engineering, but not the other way around, as there is still a huge part of software engineering that concerns non-ML components. The recently emerging role of AI/ML engineer is, in my view, a software engineer specialized in ML and, in a sense, can be seen as an evolution of a data scientist specialized in integrating ML in larger software systems. I believe the roles of data scientists, AI/ML engineers, and software engineers will keep co-existing, but there will be further specialization. The main reason is that the current spectrum of skills expected from AI/ML engineers is really wide. For instance, cloud engineers with some understanding of ML might be labeled with a new name and focus on the deployment of ML components at scale in cloud environments.

5 Paper Analysis

5.1 Novel Contract-based Runtime Explainability Framework for End-to-End Ensemble Machine Learning Serving [5]

In current ML serving frameworks, where an ML provider serves ML models to ML consumers, Service Level Agreements (SLAs) typically include only latency and throughput. With these limited SLAs, ML consumers use ML essentially as a black box. ML consumers need instead more detailed SLAs and reports with ML-specific metrics that enhance the service transpacency. Furthermore, ML providers do not receive feedback from ML consumers about served inferences. In presence of data drift, ML providers have therefore a hard time updating models to maintain a high-quality service.

Nguyen et al. [5] introduce the concept of Runtime eXplainability (RX). While traditional XAI focuses on model-level interpretability (why an ML model makes a certain inference), RX considers service-level explainability, with a holistic perspective of the ML serving framework. RX includes data quality metrics, inference confidence metrics, and inference accuracy metrics. The authors propose a novel contract-based RX framework to provide RX in an ML serving system. In short, the framework collects RX metrics to produce an explainability report, which is then used to generate a quality report for the ML consumer and a performance report for the ML provider. The performance report can additionally include feedback from the ML consumer. The RX framework can be seen as an ML observability platform for monitoring ML services. In fact, in my opinion, the term *observability* fits better than *explainability*, given that the framework focuses on collecting metrics and generating reports rather than explaining reasons behind inferences. Neverthelss, observability is a crucial aspect for enhancing transparency and trustworthiness of ML services.

An example of a real ML-enabled software product where this framework would fit is antenna tilt control in a 5G network, where the vertical electrical tilt of each antenna is controlled using RL. The ML component of this system would be an ML consumer interacting with an ML provider on the cloud. The ML provider serves RL agents that propose adjustments (actions) to update the tilt angle of antennas. Since wrong adjustments can degrade network performance and violate SLAs with customers, knowing the inference confidence and expected result is important. The proposed framework can attach quality reports to the proposed actions so that network operators can check and manually confirm or reject changes. The human manual checks would also serve as feedback to the ML provider to update the RL models. My XRL research could be integrated into this framework to extend reports with model-level explanations of proposed actions. For instance, XRL can clarify which aspect of the current state the RL model considered for the proposal [6, 7] and predict expected future outcomes [8]. Such detailed model-level explanations can complement the service-level metrics provided by the RX framework.

The concept of RX is also inspiring for my future research, as service-level XRL, or RX for RL, is an underexplored research area that can be of high interest for telecommunication systems. Typically, network operators are more interested in high-level reports using traditional network Key Performance Indicators (KPIs), rather than in the internal details of RL models, which is aligned with the ideas introduced in the paper. Also, there are unique challenges to extend RX to RL. First, data quality metrics should account for the mix of simulated and real data, as RL models are typically trained in simulators due to the actionability requirement. Given that simulators are imperfect representations of the real world, service-level XRL would need to consider the uncertainty coming from the reality gap between simulated and real data when presenting reports to ML consumers, quantified by adequate metrics. Second, since RL models output actions aiming to maximize a reward signal over a sequence of actions, typically predicting action values, the inference confidence has a more opaque meaning. Instead, XRL should focus on the confidence the agent has in achieving certain future outcomes. Contracts and SLAs between RL provider and RL consumer could then include future desired outcomes.

A further consideration is that the boundary between model-level and service-level explainability is more blurry when it comes to RL, as the RL framework is designed to be applied end-to-end directly into the real world, with interactions between agent and environment. For example, the confidence of future outcomes can be seen as a model-level explanation or service-level explanation, depending on how the outcome is defined. If the outcome is defined as the reward used by the RL algorithm the explanation would be at model-level, while service-level explanations would typically focus on KPIs used in SLAs.

5.2 Towards a roadmap on software enigneering for responsible AI [9]

Current responsible AI guidelines are abstract and do not provide concrete best practices for AI practitioners and developers. Also, AI researchers focus on model-level interpretability, while responsible AI concerns the wider software development lifecycle. Given these problems, Lu et al. [9] conducted a systematic literature review to provide a research roadmap for operationalizing responsible AI. The paper presents the current state of responsible AI and a set of research challenges to guide future research in software engineering for responsible AI.

Responsible AI is analyzed from different perspectives: governance, process, and system. The governance perspective includes structures and processes designed to ensure compliance with ethical regulations. The process perspective concerns practices that developers should apply during the software development lifecycle (requirements, design, implementation, testing, and operations) for compliance with regulations. The system perspective considers architectural and design patterns for responsible AI by design in AI-enabled software systems that include AI and non-AI components.

The topics discussed in the paper are really relevant for my research on XRL for telecommunication networks. In fact, XAI is a key enabler for responsible AI and is mentioned by the authors in both process-level and system-level responsible AI. Given my research focus on algorithmic interpretability of RL models, I believe XRL research provides valuable methods to achieve responsible AI by design for RL components.

From the process perspective, my XRL research could be used to improve two of the design techniques discussed in the paper for reducing ethical risk. The first technique is *reducing frequency of occurrence*, which consists of reducing how often an AI system makes automatic decisions. If an RL-generated action proposal is accompanied with explanations generated by XRL methods, humans can more easily decide whether to approve or reject the action proposal [8]. The second technique, *consequence response*, involves defining procedures to respond to ethical problems. A known use of XAI and XRL is to perform root-cause analysis [10], which can help to design effective consequence response procedures.

From the system perspective, my XRL research could be integrated into the *AI mode switcher* architectural pattern described in the paper. This pattern consists of designing the AI-enabled software system so that AI components can be switched on and off, either by the user or by automatic procedures. For example, a built-in guard can activate the AI component only when predefined conditions are met (e.g., inputs in a certain domain). Similar to the process-level, where XRL can help humans validate and override actions, explanations can provide useful information based on which humans or automated rules can decide whether to use the AI component or not. For example, in my XRL research, I propose to predict expected future outcomes of RL actions [8]. Such predictions can be used to decide whether to activate or deactivate the RL agent based on predefined rules or human validation.

Overall, responsible AI is a crucial topic in my research on XRL for telecommunication networks, as network operators require trustworthy and transparent solutions to adopt RL-based network automation. Thus, I find the research roadmap presented in the paper a useful guide for my future research on XRL. My future research is already well aligned with the roadmap proposed in the paper due to the relevance of explainability in responsible AI. A slight adaptation could be to extend the scope of my research to system-level aspects such as data quality rather than only model-level interpretability, as also discussed for [5].

6 Research Ethics & Synthesis Reflection

I used the following method to select the papers [5, 9] described in Section 5:

1. Checked the list of accepted long papers from CAIN 2022-2025 using the official CAIN website.
2. Filtered the papers containing one of the following keywords in their title: reinforcement learning, explainability, explainable AI, trustworthy AI, trustworthiness, responsible AI. The result was a set of 8 papers.
3. Read title and abstract to select 2 papers based on the perceived relevance to my research.

The above process was the result of a few adjustments:

- Before searching in the official CAIN website, I tried to select papers via IEEE Xplore. After selecting two papers on RL, I realized they were not long papers, and that IEEE Xplore includes industry talks and posters in the list.
- I initially used only reinforcement learning as a keyword for my search, but had to extend the set of keywords due to lack of long papers on RL.

I did not use any Large Language Model (LLM) detector to ensure originality of the selected papers. I used my critical thinking and intuition while reading the papers. I also trust that the reviewers have checked the originality of the content. For writing this assignment, I did not use any LLM-based aid.

References

[1] Franco Ruggeri. *Explainable Reinforcement Learning for Mobile Network Optimization.* Stockholm, Sweden: KTH Royal Institute of Technology, 2025. 113 pp. ISBN: 978-91-8106-180-2.

[2] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "Anchors: High-Precision Model-Agnostic Explanations". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 32.1 (Apr. 25, 2018). ISSN: 2374-3468, 2159-5399. DOI: `10.1609/aaai.v32i1.11491`. URL: `https://ojs.aaai.org/index.php/AAAI/article/view/11491` (visited on 11/10/2025).

[3] Alexander Lavin et al. "Technology Readiness Levels for Machine Learning Systems". In: *Nature Communications* 13.1 (Oct. 20, 2022), p. 6039. ISSN: 2041-1723. DOI: `10.1038/s41467-022-33128-9`. URL: `https://www.nature.com/articles/s41467-022-33128-9` (visited on 11/10/2025).

[4] Christian Kästner. *Machine Learning in Production: From Models to Products.* First edition. Cambridge, Massachusetts: The MIT Press, 2025. 1 p. ISBN: 978-0-262-04972-6.

[5] Minh-Tri Nguyen, Hong-Linh Truong, and Tram Truong-Huu. "Novel Contract-based Runtime Explainability Framework for End-to-End Ensemble Machine Learning Serving". In: *Proceedings of the IEEE/ACM 3rd International Conference on AI Engineering - Software Engineering for AI.* CAIN 2024: IEEE/ACM 3rd International Conference on AI Engineering - Software Engineering for AI. Lisbon Portugal: ACM, Apr. 14, 2024, pp. 234–244. ISBN: 979-8-4007-0591-5. DOI: `10.1145/3644815.3644964`. URL: `https://dl.acm.org/doi/10.1145/3644815.3644964` (visited on 11/12/2025).

[6] Bradley Hayes and Julie A. Shah. "Improving Robot Controller Transparency Through Autonomous Policy Explanation". In: *Proceedings of the 2017 ACM/IEEE International Conference on Human-Robot Interaction.* HRI '17: ACM/IEEE International Conference on Human-Robot Interaction. Vienna Austria: ACM, Mar. 6, 2017, pp. 303–312. ISBN: 978-1-4503-4336-7. DOI: `10.1145/2909824.3020233`. URL: `https://dl.acm.org/doi/10.1145/2909824.3020233` (visited on 03/06/2023).

[7] Ahmad Terra, Rafia Inam, and Elena Fersman. "BEERL: Both Ends Explanations for Reinforcement Learning". In: *Applied Sciences* 12.21 (Oct. 28, 2022), p. 10947. ISSN: 2076-3417. DOI: `10.3390/app122110947`. URL: `https://www.mdpi.com/2076-3417/12/21/10947` (visited on 11/21/2022).

[8] Franco Ruggeri et al. *Explainable Reinforcement Learning via Temporal Policy Decomposition*. Jan. 7, 2025. DOI: `10.48550/arXiv.2501.03902`. arXiv: `2501.03902 [cs]`. URL: `http://arxiv.org/abs/2501.03902` (visited on 01/08/2025). Pre-published.

[9] Qinghua Lu et al. "Towards a Roadmap on Software Engineering for Responsible AI". In: *Proceedings of the 1st International Conference on AI Engineering: Software Engineering for AI*. CAIN '22: 1st Conference on AI Engineering - Software Engineering for AI. Pittsburgh Pennsylvania: ACM, May 16, 2022, pp. 101–112. ISBN: 978-1-4503-9275-4. DOI: `10.1145/3522664.3528607`. URL: `https://dl.acm.org/doi/10.1145/3522664.3528607` (visited on 11/12/2025).

[10] Ahmad Terra et al. "Explainability Methods for Identifying Root-Cause of SLA Violation Prediction in 5G Network". In: *GLOBECOM 2020 - 2020 IEEE Global Communications Conference*. GLOBECOM 2020 - 2020 IEEE Global Communications Conference. Taipei, Taiwan: IEEE, Dec. 2020, pp. 1–7. ISBN: 978-1-7281-8298-8. DOI: `10.1109/GLOBECOM42002.2020.9322496`. URL: `https://ieeexplore.ieee.org/document/9322496/` (visited on 02/28/2022).