Essay 4 – Testing AI-based Systems

This essay is based on the *Systematic literature review of validation methods for AI systems* article written by Myllyaho et.al (2021). The researchers studied articles that combine artificial intelligence, validation, and an emphasis on end-users and industries. The different kinds of AI systems presented in the literature were analyzed, and their validation methods were examined. Most of the research papers in the literature survey were categorized as ML such as classification, not AI in general.

Validation Methods for AI/ML-powered Systems

The main findings were a division of validation activities and approaches into initial and continuous validation. Initial validation was more frequently used in the research papers than the continuous approach. (Myllyaho et.al. 2021)

Initial Validation Methods

Initial validation entails validation at deployment to ensure desired system functionality. Different research papers examined by Myllyaho et.al. (2021) used these methods in different combinations, including only using just one method for validation.

Simulation

Simulations can be virtual or real-world environments. Myllyaho et.al. noted that simulations were used in three variations: fully virtual, hardware-in-the-loop, and system-in-the-loop. In **fully virtual** simulations the final deployment environment is replicated with a virtual simulator, with possibly some of the components of the system under test or the system software. Some common techniques are simulator-created inputs and assessing outputs, passing down test cases as ontologies or constraints, test design by hand, and fault injection. Simulations in general provide safety in testing critical systems, but it is noteworthy that validating the simulator itself is challenging and costly. (Myllyaho et.al. 2021)

Hardware-in-the-loop (HIL) simulation, which is similar to the fully virtual approach, but with the addition of non-virtual components. Cyber-physical systems like cars and robots were prominent in this category. One approach is to provide inputs to the system via the sensors, instead of directly to the system. HIL has similar benefits and drawbacks as fully virtual simulations, but it allows for fault injection to both hardware and software and allows for the monitoring of hybrid faults originating from the interaction of these. (Myllyaho et.al. 2021)

In **system-in-the-loop** (SIL) simulation, a full system is placed into an artificial environment. The environment might mimic just some features of the real environment. Laboratory settings can be used to check that a robot does what it is intended to do before it is set into a real environment. The environment can be fully virtual as well, but in comparison with the fully

virtual simulation, the whole system is used, not just its software. Using the whole system gives a more precise picture of system behavior, but as with all simulations, it can be costly and requires careful design of the simulation. Careless simulation design can result in a false impression of the functionality. (Myllyaho et.al. 2021)

Trial

Entire systems under test can be deployed and monitored in real or almost real environments, planned in a manner so that at least some key scenarios are covered. **Trial in a real environment** aims to validate systems in their final environment, and validation settings can be arranged so that key scenarios are examined. This type of trial is costly and rigorous but might reveal issues in areas such as component collaboration and usability. (Myllyaho et.al. 2021)

Trial in a mock environment involves a full system under test, that is validated in a replication of its environment. Mocked environments might be conducted in a laboratory setting, minimizing the risks of damage in a real setting. Distinguishing between a SIL and trials in a mock environment is not straightforward, but Myllyaho et.al. (2021) have categorized fully artificial environments as SIL. The environments are more extensive in the trial mock approach. (Myllyaho et.al. 2021)

Model-centered

In this data-driven approach, trust in the system functionality stems from the underlying model validation. Common techniques are often seen in traditional ML model validation, such as cross-validation. The robustness of a model could be tested with "torture tests", where the model is intentionally altered, or inputs are slightly modified to assess effects on model performance. Model-centered validation can be used as initial validation before proceeding to other validation techniques. (Myllyaho et.al. 2021)

Expert Opinion

Expert opinions were rarely reported in the survey articles. In some cases, experts in the business domain of the system under test were brought in to assess example scenarios. Expert opinions can guide development. In addition to bringing in experts, the system under test can be compared to a benchmark application to gain insight into which scenarios and failures are crucial. It is suggested that expert opinions are used in combination with other validation methods. (Myllyaho et.al. 2021)

Continuous Validation Methods

After deployment, systems can be monitored and validated to ensure that the desired functionality is still present, even in situations that were not initially expected and in cases where systems learn and adapt to changes. Continuous validation was not discussed in most research papers, the focus of the majority was initial validation. (Myllyaho et.al. 2021)

Failure Monitoring

Ensuring that system malfunctions will not go unnoticed is part of continuous validation during the system lifecycle. The system, sub-systems, and components are designed to alert users in cases of malfunction and unexpected behavior. Human intervention is always required, which might be dangerous in situations that require rapid action. (Myllyaho et.al. 2021)

Safety Channel

Backup components are safety channels that take over if a component is compromised. The system should continue to function during a malfunction, possibly by offering a way for the system to fail safely by using a simpler replacement component. Often safety channels reduce the system functionality but provide some time to react to the malfunction. (Myllyaho et.al. 2021)

Redundancy

In contrast to safety channels, in the redundancy approach, critical components are duplicated within a system to maintain full system functionality. Redundancy prepares systems better for malfunction and failure but is more costly. (Myllyaho et.al. 2021)

Voting

Multiple intelligent components can be integrated into a system to execute the same task, and to vote which approach should be executed. If a single component fails damage is reduced. As for redundancy, the requirement for more components adds costs. (Myllyaho et.al. 2021)

Output and Input Restrictions

By restricting the **outputs** of a system, a system can be continuously validated to not take actions that are known to be dangerous or false, which also increases the system's predictability. Limited outputs can be applied to autonomous cars, for example restricting lane switching when another car is nearby. **Input** restriction limits the types of situations a system is allowed to handle autonomously, to ensure that the system does not handle situations it was not trained to handle. (Myllyaho et.al. 2021)

Current and Upcoming Challenges and Possibilities

In this section, the literature survey is compared with lecture materials, and additional sources to broaden my understanding of and reflect upon testing AI-based systems. This section briefly handles some of the possible topics that can be discussed, both from a challenges and possibilities angle, but also in relation to current software testing practices.

Oracle problem

In the survey, Myllyaho et.al. (2021) noted that most of the papers did not mention traditional software testing methods. It is proposed that one reason for this is the Oracle problem, where testing non-deterministic models with deterministic tests is difficult (Myllyaho et.al. 2021). As there is a need for stochasticity in AI models to achieve better results, including techniques such as simulated annealing, the same output for the same input can not always be achieved. This non-deterministic feature might lead to problems in unit testing and automated regression tests.

The *Certified Tester AI Testing (CT-AI) syllabus* by the ISTQB board also notes this issue of test oracles in AI-based systems. In addition to the previously stated test oracle challenges, it is noted that an interpretation of an AI-based system's correct behavior is often subjective. One way to tackle this, according to the syllabus (ISTQB n.b.) is to allow for some deviation in the expected results. The lack of traditional testable requirements from where test cases can be derived is also noted in the syllabus, as often AI-based systems have high-level business-oriented specifications.

One possibility is to use AI in software testing, to decide whether an output is acceptable or not. In addition, allowing for deviation requires that the system can handle these deviations, as often expected outputs are compared in unit and integration testing in a flow within and between components. Therefore, it is expected that AI/ML does not just affect how we test, but also how we build software. As stated in the syllabus, AI-based system development is an exploratory process, where the development process starts with exploring data, rather than gathering requirements.

Flexibility and Adaptability

AI-based systems should be flexible and handle situations they were not originally trained for, and adaptable in the sense that there should be ease in how well they fit new situations. Thus, specifications are harder to formulate, as the concept of correct behavior is wide.

Evolving AI-based systems will need more continuous validation techniques, such as the ones presented in the survey by Myllyaho et.al. (2021). AI-based systems expand validation to consist of initial and continuous validation.

Myllyaho et.al. (2021) note that traditional testing might find its place in an age of AI/ML as well, as not all systems change their behavior autonomously. One interesting aspect they bring up is the use of corner cases to handle adaptability, as systems can have corner cases where functionality should remain intact and thus be testable by regression testing (Myllyaho et.al. 2021)

Model Validation, Statistics, and Data

Standard ML model validation was also listed as a validation activity, which requires skills in statistics as well. Other key considerations are well-represented training and validation data sets, the correct model validation method, and sample sizes. Overall, it might be that the demand for statistical skills in testing and quality assurance increases as we move into a more data-driven world where AI/ML solutions are more common. As seen in the survey by Myllyaho et.al., (2021) the papers presented a lack of consideration of sample sizes when validating the AI/ML systems as well, showing that statistical knowledge is used outside of the ML model itself.

In addition to the statistical skills testers might need, the understanding of ML and AI concepts will probably increase. ML models can be used in different degrees and variations, posing different transparency, interpretability, and explainability issues, perhaps on both component and system levels. Easier models, such as linear regression results are easier to explain than neural network results. As AI-based systems gain popularity in different fields, these and other AI quality characteristics will be important.

As AI-based systems are data-driven, the importance of data processing and context awareness increases. Standard accuracy metrics do not reveal bias in data, and there are fairness issues to be considered. Often data-driven decisions are misleadingly seen as logical, but fairness-testing, for example using subgroup metrics, can help counter our biases. However, biases are hard to detect and exist to some degree in all societies.

AI as a Component in Systems

Often, AI is just seen in components of the system under test. In a *How AI changes Software Testing* blog post, Pyhäjärvi (2024) discusses how the emergence of AI has been seen in her testing career. She proposes that a system should be seen as components and not describe whole systems as AI-based systems when they have AI-based components (Pyhäjärvi 2024). As many components are still deterministic, the need for traditional software testing is probably not diminishing soon.

In the ISTQB syllabus, the listed test levels for AI-based systems are input data testing, ML model testing, component testing, component integration testing, system testing, and acceptance testing. Test data acquisition is listed as a challenge, as often testing of AI systems requires data. (ISTQB n.d.)

The component, system, and acceptance testing sections in the ISTQB syllabus (n.d.) explain that conventional testing levels are used in the testing of AI-based systems as well. The test approach is selected based on risk analysis, including risks such as poor data quality, choice of ML frameworks, self-learning services not meeting user expectations, and model overfitting.

Transparency

As stated in the ISTQB syllabus, often AI-based systems are used to implement tasks that are too difficult for humans to perform (ISTQB n.d.), which can also add to the test oracle problem. ML models outperform humans in, for example, pattern recognition, and these kinds of systems can be hard for a human to comprehend. Additionally, the syllabus notes that the complexity of AI-generated systems increases when AI-based components interact with each other. (ISTQB n.d.)

The outcomes of the AI-based systems should be explainable in many domains, such as loan-granting decisions that affect people. It is important to consider when and where AI-based components are used, and for example how explainable underlying ML models are.

Autonomous AI Agents

As the survey by Myllyaho et.al. was conducted in 2021, it does not cover the most recent AI agent literature, but some papers did handle autonomous vehicles such as self-driving cars. In comparison with the 2010s, where most papers in the survey are written, we have already entered a more multiagent world, as robots drive on our sidewalks and some agents have started handling some minor tasks such as booking appointments.

When even more AI agents hit the market, testing how different agents interact with each other will be even harder. We have limited insight into how AI agents make decisions, as the models that drive them are complex and the nature of reinforcement learning involves constant change.

Testing or Safeguarding

As seen in the continuous validation methods, testing and safeguarding intersect. Continuous validation acts as a safety net to catch and handle malfunctions.

Supervision of Autonomous Systems Paradox

It is also noteworthy to reflect upon the rigorous testing and validation aspects presented in the literature. As AI models are intended to be autonomous and adapted on their own, many domains still require continuous validation. Even though AI-based systems should be flexible and adaptable, many applications still require a human in the loop, which is counterproductive, if autonomy is the main goal of an AI-based system.

Conclusions

As seen in the considerations above, there are many aspects of validating AI-based systems, but no easy answers. It is hard to see when we will enter a new AI winter, where the development of AI/ML slows down. Currently, the rising complexity and decreasing transparency of the models might pose more changes to testing. In my opinion, this question of possible progress speed is one of the bigger challenges in software development at the moment. As it seems that AI-based systems get increasingly more powerful, the complexity of the systems grows exponentially.

Pinpointing when self-evolving systems change can also be harder as they grow. We will probably have to work harder to develop simulations and mock environments. As stated by Myllyaho et.al. (2021) most simulations and mock environments were not properly validated, at least in the academic setting in the papers. It remains to be seen how the software field will be regulated in the future, as ensuring safe systems in an autonomous agent age is increasingly more difficult. As many validation methods such as simulations presented in the survey are costly and difficult to design, there is a conflict with the desire to cut costs in organizations.

The cost-security trade-off might limit the emergence of autonomous agents such as self-driving cars for a while longer, at least in strictly regulated areas such as the EU. There are still many ethical things to consider, such as how autonomous vehicles behave in accidents with fatal outcomes. Many things that have previously been left to mere chance must now be decided by the system, which brings new challenges to legislation and the judicial system. Many questions about how we place accountability are still open, as even if autonomous agents can act in an environment, they cannot be legally liable themselves.

There is also the issue of a lack of AI literacy in the general public, such as end-users and people who are affected by the decisions made by AI-based systems. Quality work for end-users will have to go beyond validation, as the systems have more and more components that are harder to explain in user manuals. Together with the software development field, the general public will also have to shift toward a more AI-based world. Real-world change is slower, which inevitably will cause friction.

Even when many real-world questions remain to be answered, there is optimism. AI-based systems might cause new problems, but might also lead to advancements in areas such as diagnostics and medical advancement.

References

Lecture materials

International Software Testing Qualifications Board. (n.d.). *Certified Tester AI Testing (CT-AI) syllabus* (Version 1.0). https://www.istqb.org/wp-content/uploads/2024/11/ISTQB_CT-AI_Syllabus_v1.0_mghocmT.pdf

Myllyaho, L., Raatikainen, M., Männistö, T., Mikkonen, T., & Nurminen, J. K. (2021). Systematic literature review of validation methods for AI systems. *Journal of Systems and Software, Volume 181*

Pyhäjärvi, M. (2024). *How AI changes software testing?* A Seasoned Tester's Crystal Ball. https://visible-quality.blogspot.com/2024/03/how-ai-changes-software-testing.html