# Shifting Quality Assurance of Machine Learning Algorithms to Live Systems

Florian Auer[1] Michael Felderer[1]

The behavioral adaptability of machine learning algorithms makes the assurance of their quality challenging (e.g. previous inputs affect future processing). The complex problems tackled by these algorithms complicate the assurance techniques. Testing, for example, suffers from the situation that in most cases only costly or even no test oracles are available [Ba15]. Solutions presented in literature require additional effort in testing. For instance, the use of pseudo-oracles [MKA06] that require an additional, independent implementation or metamorphic testing [MSK09] that involves the manual finding of meaningful metamorphic properties and execution of additional test cases.

A fundamental weakness of existing solutions is the assumption that test environments sufficiently mimic the later application in order to allow quality assurance. However, given the data dependent behavior of these algorithms, only limited reasoning about their later performance is possible. Thus, meaningful quality assurance is not possible with test environments exclusive. A shift from the traditional testing environment to the live system is needed. Hence, costly test environments for simulation are replaced with available live systems that constantly execute the algorithm – in some cases even supervised by humans.

In order to observe an algorithm in live systems, its implementation has to be prepared for monitoring and the collection of the measured data has to be organized. Similar requirements are found for the evaluation of experimental features. *Live experimentation* is the controlled deployment of experimental features (like a new learning algorithm) in live systems by instrumenting the program, collecting runtime data and analyzing the affect [Fa14].

This kind of experimentation serves as a vehicle to enable quality assurance in live systems. The observation of the algorithm at runtime allows to evaluate quality attributes like functionality (e.g. unsatisfying results), reliability (e.g. frequency of failures), usability (e.g. user feedback) or efficiency (e.g. execution time). Furthermore, information about the input and output data can be collected. Given the behavioral dependency of these algorithms on data, its analysis can reveal valuable information about the data (e.g. value space, frequency of values) and about the related behavior of the algorithm (e.g. clusters of inaccurate output, learning behavior). This results into a deeper understanding of the algorithms and serves as an initial step towards improved explainability of machine learning algorithms.

Thus, it seems that live experimentation is suitable for quality assurance. Nevertheless, it requires further research on how to find the appropriate metrics, instrument the implementation, deploy the implementation (e.g. it is not desirable to deliver experimental/insufficient tested algorithms to all customers) and analyze the results. In addition, the immature state of

[1] University of Innsbruck, Quality Engineering, Innsbruck, Austria, firstname.lastname@uibk.ac.at

practice for live experimentation requires further research on a taxonomy of experimentation types along with patterns that represent best practices on the implementation of experiments. This would support initial decision making and give practitioners a clear guidance [Fa14].

Many of the traditional testing techniques are still applicable in live systems. For example, regression testing (e.g. between versions), invariant checking (e.g. in combination with automatic invariant detection [Ba15]), manual testing (e.g. feedback from users) or metamorphic testing (e.g. execution of input variants in background [MSK09]). However, the implications of their application in a live system are not well understood (e.g. performance impact). Thus, an evaluation of testing techniques for the use in live systems is necessary.

With the transition to the live system, testing is conducted implicitly by the users of the system. As a consequence, the test data is actual data from the users. This requires a different data management as in traditional testing environments. For instance, data may be confidential or business-critical. Another aspect that requires attention is user satisfaction in the presence of continuous change and less mature algorithms. Furthermore, ethical aspects have to be considered (e.g. wrong credit score assessment). As similar challenges are faced in crowdtesting, existing approaches for crowdtesting may be adaptable.

Finally, this approach to quality assurance of machine learning algorithms is not always applicable. For instance, safety-critical software has to provide reliable services. Deploying insufficient tested algorithms to such systems is dangerous and may lead to serious damage. In contrast, other applications (like suggestion services) can provide wrong results without serious consequences. Thus, research about techniques to support application cases that require correct results is necessary to broaden the applicability of this approach. Another challenge that requires further research is that not always a definition for correct output exists (e.g. output is approximation). As a result, the collected feedback may be inconsistent.

In order to provide an environment that allows further conclusions about this approach, the next step would be a software quality assurance analysis platform that allows to plan, prepare, conduct and analyze systematic deployments of instrumented algorithms.

## References

[Ba15] Barr, Earl T; Harman, Mark; McMinn, Phil; Shahbaz, Muzammil; Yoo, Shin: The oracle problem in software testing: A survey. IEEE transactions on software engineering, 41(5):507–525, 2015.

[Fa14] Fagerholm, Fabian; Guinea, Alejandro Sanchez; Mäenpää, Hanna; Münch, Jürgen: Building blocks for continuous experimentation. In: Proceedings of the 1st International Workshop on Rapid Continuous Software Engineering. ACM, pp. 26–35, 2014.

[MKA06] Murphy, Christian; Kaiser, Gail; Arias, Marta: A Framework for Quality Assurance of Machine Learning Applications. -, 2006.

[MSK09] Murphy, Christian; Shen, Kuang; Kaiser, Gail: Automatic system testing of programs without test oracles. In: Proceedings of the eighteenth international symposium on Software testing and analysis. ACM, pp. 189–200, 2009.