Product Assurance for On-Board Learning-Enabled Software

I. Dragomir [(1)], S. Tonetta [(2)], C. Englert [(3)], R. Lange [(3)]

[(1)] GMV Aerospace and Defence, Isaac Newton 11, PTM, Tres Cantos, 28760, Spain, Email: idragomir@gmv.com

[(2)] Fondazione Bruno Kessler, via S. Croce 77, 38122, Trento, Italy, Email: tonettas@fbk.eu

[(3)] European Space Agency (ESA), ESTEC, Keplerlaan 1, PO Box 299, Noordwijk, The Nertherlands, Email: ralf.lange@esa.int

Autonomy is deemed one of the most essential capabilities space systems shall provide for reliable operation in challenging dynamic environments, sometimes with limited bandwidth and long communication delays that restrict the possibility of direct operation from ground. Additionally, future missions have stringent autonomy requirements to maximize return both in terms of operations and science. In order to achieve autonomy, Artificial Intelligence, and in particular Machine Learning (ML), techniques may play a crucial role. Great advancements have been achieved by enabling autonomy through ML in everyday (terrestrial) applications, with self-driving cars being one of the most representative example.

ML is also gaining momentum in space on-board software, with such solutions recently deployed in satellites mainly for Earth observation. Several activities are investigating the use of ML for e.g., optimal control, fault detection, isolation and recovery (FDIR), landing and obstacle detection, but also its safe deployment through (formal) verification and validation given the critical nature of these systems.

The use of ML in on-board software requires a change of paradigm in the development, validation, verification and quality assurance processes. ML requires very large datasets to produce through training a model capable to infer properties of the provided inputs, e.g., classification - prediction of a discrete output, regression - prediction of a continuous output. The challenges that need to be addressed for each of these processes are manifold. On one hand, the model development approach needs to fit within the global approach of the software development. Hence specific activities related to data management and model inference need to be formalized with clear outcomes and timely schedule that would allow the correct and complete development. On the other hand, the model development needs to achieve a certain quality level that would enable its deployment. This quality shall also be expressed for the specific activities: data representativeness (accuracy, balance, etc.), completeness and independence can be considered for the data management, and reproducibility, replicability, verifiability, usability etc. for the model inference. Such properties shall be assessed through dedicated verification and validation activities that could employ possibly the use of formal models. For example, functional properties such as stability, robustness, resilience, monotonicity, etc., can be formally specified and checked with the different techniques proposed in the literature, although their maturity depends on the application and context.

Before deployment, the model needs to be implemented, validated and verified with respect to its requirements and also included for the overall system requirements validation. The implementation involves transforming the trained model into an executable model that runs in specific hardware, including additional software and optimizations that enable its execution. The validation and verification should demonstrate that the properties of the model hold also for its implementation, and that assumptions made at development phase also hold or that the implementation is able to mitigate them. Examples of such cases are the detection and handling of out-of-distribution data, handling of uncertainty - aleatoric and epistemic - for robustness and resilience, accurate calibration of the output probability for trustworthy predictions. The software quality model shall reflect the specific aspects of ML implementations.

In addition, the use of ML needs to be contemplated at system level. Even though evidence about the learning enabled subsystem can be obtained through the process, the robust operation in non-nominal conditions shall be supported through isolation and redundancy.

The Software Product Assurance of Autonomous On-Board Software (PASSIONS) project, funded by the European Space Agency, is addressing the above-mentioned topics. The aim of the activity is to provide a learning assurance methodology based on different verification, validation and safety analysis techniques for mixed traditional-ML on-board software in line with the ECSS standards. More specifically, the activity proposes a new development lifecycle that adapts the ECSS approach to the ML paradigm and milestones. A great importance is given to the validation and verification activities and the use of formal verification and validation to check the functional correctness, safety and dependability of ML, including adaptations of the architecture (e.g., redundancy) to ensure the system safety. A data, model and adapted software quality model are proposed, including (sub)-characteristics and associated metrics that can be automatically computed and validated. Finally, ECSS guidelines will be established on the obtained result to compose the learning assurance methodology. In this presentation we will discuss the results obtained to date on each of these topics.