Invited Talk: Benchmarking and Feasibility Aspects of Machine Learning in Space Systems

Amir Raoofy*, Gabriel Dax*, Max Ghiglione§, Vittorio Serra§, Martin Werner*, Carsten Trinitis*

TU Munich* Airbus Defence and Space GmbH, Munich§

{amir.raoofy, gabriel.dax, martin.werner, carsten.trinitis}@tum.de {max.ghiglione, vittorio.serra}@airbus.com

Abstract

Compute in space, e.g., in miniaturized satellites, requires dealing with special physical and boundary constraints, and requirements, including the limited energy budget. These constraints impose strict operational conditions on the on-board data processing system and its capability in dealing with sophisticated workloads suchlike Machine Learning. In the meantime, the breakthroughs in Machine Learning based on Deep Neural Networks in the last decade promise innovative solutions to drive the space industry forward and expand the functional capabilities of on-board data processing. However, due to the aforementioned special requirements, performance- and power-efficient, and novel solutions and architectures for deploying Machine Learning via, e.g., FPGA-enabled SoC, particularly Commercial-Off-The-Shelf (COTS) solutions, are gaining significant interest in the space industry.

In this context, it is essential to conduct extensive benchmarking and feasibility analyses in different aspects for deploying Machine Learning to space: specifically, such feasibility analyses would require investigation of frameworks for programming and deployment as well as the deployment of various models and datasets. Additionally, feasibility analyses of real-world use cases and applications are needed. To this end, a research and development activity is funded by the European Space Agency (ESA) General Support Technology Programme and is led by Airbus Defence and Space GmbH with the goal of developing a Machine Learning Application Benchmark (MLAB) that covers the benchmarking and feasibility aspects mentioned above.

In this invited talk, we provide an overview of the MLAB project and discuss development and progress in various directions, including framework analyses, model, and dataset investigation. We elaborate on a benchmarking methodology developed in the context of this project to enable the analysis of various hardware platforms and options. We specifically focus on a particular use case of aircraft detection as a real-world example and provide quantitative analyses for various performance and accuracy indicators including, accuracy, throughput, latency and power consumption.

Keywords

Onboard Computers, Machine Learning, Benchmarking, FPGA-enabled COTS

1 Introduction

With continuing advances of modern Machine Learning (ML) technologies such as Deep Neural Networks (DNNs) and especially Convolutional Neural Networks (CNNs), space applications can take advantage of the capabilities of models and algorithms. This helps to extend the functionalities of on-board computers in satellites for various applications such as, e.g., Earth observation [2] or optical guidance, navigation and control (GN&C) [1]. Therefore, the European Space Agency (ESA), as a player in the Space industry, is interested in deploying ML workloads to on-board computers.

Deploying ML inference to on-board computers of satellites has multiple advantages, including saving the on-board storage, and saving the limited bandwidth between satellites and ground stations. Additionally, it enhances the capabilities of on-board computers for data-driven real-time decision making. However, the energy consumption and dissipation demands of satellite missions are extreme: 1) the energy for ML inference needs to be supplied by the limited solar panels, and 2) the generated heat needs to be radiated away from the satellite.

In this context, there is a clear need for the development of ML inference benchmarks for various Space applications and models, that enable the feasibility analysis of various hardware solutions and configurations based on real-time performance and power efficiency requirements.

In recent years, Field-Programmable Gate Arrays (FPGAs) are getting rising attention for the acceleration of various ML workloads, in particular, ML inference [3]: Due to the increasing availability of both radiation-tolerant devices and Commercial-Off-The-Shelf (COTS) solutions. Moreover, the unique features of these devices, e.g., low power consumption and hardware (re)programmability, have drawn the attention of the Space industry to investigate these accelerators further. However, despite the rising attention and the increasing use of FPGA-based inference accelerators, there is little work on benchmarking ML inference on space-capable hardware with a focus on space-specific resource limitations.

In summary, there is a need to develop benchmarking methodologies and systems for the investigation of ML in Space on space-capable hardware that are interesting to the industry. This enables the space industry to systematically conduct feasibility analyses and evaluate the trade-offs of various hardware platforms and configurations, and contribute to the design of on-board computers for future missions.

2 Framework Options

We start by elaborating on the feasibility aspect and challenges associated with programming and deploying ML solutions using state-of-the-art developments and deployment frameworks.

Typical use cases of Neural Networks for remote sensing missions include classification, object detection, and segmentation. Classical models like Resnet, Yolo, and Unet are examples of DNN families that are commonly employed for these tasks. For example, ML-enalbed solutions targeting the Airbus wind turbine patch classification, Airbus aircraft detection, and Airbus ship segmentation datasets are using the above-mentioned models, respectively.

For the development of the benchmark, we use these models as targets for development and deployment, and to assess the capabilities and limitations of various tools in FPGA deployment.

While in the last couple of years, ML model development (i.e., training) and deployment tools targeting FPGA had a lot of limitations; it can be observed that the development environment is slowly maturing due to the emergence of active development, interest, and support in the community. Specifically, the support for the prominent development frameworks, e.g., TensorFlow, Pytorch, and Caffe, as development frontends, is becoming commonplace in many workflows. Therefore, FPGA-enabled SoC deployment for space can benefit from standard implementations of particular models (Resnet, Yolo, and Unet) in the above-mentioned frameworks. Alternatively, it is also possible to customize the algorithms with layers that are more easily supported by inference tools, which provides sufficient flexibility for the deployment and investigation of future ML-oriented approaches. Regardless of the deployment workflow and the efficiency of the developed solutions, the standard models that we target are implemented in at least one of the above-mentioned frameworks, which simplifies deployment to COTS systems significantly.

While using the same frontend for model development sounds very promising, in comparison to CPU or GPU deployment, the development workflow for FPGA has additional overheads that imply a supplemental development effort: 1) Deploying a model on FPGA requires conducting an investigation regarding the supported layers and software components for the selected model within the targeted deployment framework. The state-of-the-art approach for dealing with this limitation relies on either manual and rigorous investigation of NN layers or analysis of errors in downstream deployment workflow. Therefore, specific versions of the mentioned frameworks need to be exploited as the downstream deployment tools might impose strict version requirements. 2) The deployment workflows require conducting the development and training of the model in a custom frontend framework. In such cases, the development and deployment are fully coupled processes, and an extra effort for porting the model into the custom framework is expected in exchange for potential efficiency benefits.

In summary, in many developments and deployment frameworks, layer support, model format support, and versioning consistency issues might require redesign, simplification, modification, even re-implementation, and eventually re-training of the models in the development framework.

3 Benchmarking Methodology

Next, we focus on the benchmarking aspects of ML in Space. We identify the investigation of development frameworks and various SoC platforms as the core part of feasibility analysis. In particular, we are targeting benchmarking for a special set of hardware and software frameworks. In addition, Space tasks and scenarios are very special cases with the formerly-mentioned special requirements. Therefore, existing benchmarking methodologies and infrastructure (e.g., MLPerf Inference) are not fully aligned for our use case. While reproducibility, plays an important role in our benchmarking methodology, specification of the standard inference benchmark driver interface is less of a concern, as most of the target platforms offer a similar architecture. Therefore, we can rely on benchmark drivers and scripts that are developed and designed separately for each use case.

Benchmark Design: We identify a particular space use case by defining a *scenario* which determines whether the use case deals with a batching or streaming scenario, as well as a *task* that is used. Each benchmark instance specifies a particular scenario and a task to which the results (i.e., submission) should adhere. In addition, each benchmark instance specifies a particular set of configurations and performance requirements which are then used for verification.

Scenarios: In order to cover the realistic inference scenarios for space applications, we define two scenarios. These scenarios aim at performance evaluation for various hardware while considering feeding the input data to the accelerator with either *batches* or *streams*.

Tasks: We are considering various tasks associated with machine learning in space. The main focus is on vision tasks, e.g., classification and object detection, however, we are also considering anomaly detection tasks as well. We are also binding a dataset and a model to a particular task to prevent the additional complexity of in configuration of datasets and models.

Submission and Verification We gather and organize a database of benchmark records that are successfully prepared, using a submission system. We propose a submission system as the user interface for the benchmark developers to provide the results of their benchmarks. Therefore this submission system includes a simple database for record and metadata keeping as well as checker scripts that aim at assessing the satisfaction of performance requirements of a benchmark instance.

4 Aircraft Detection Use Case

In the context of machine learning in space, object detection tasks have great importance, as they can potentially serve as one of the critical building blocks in, e.g., navigating and control. Therefore, as a basic example of deep learning model for Space, we specifically focus and work on object detection tasks. Since Yolo model [**?**] is one of the most popular, accurate, and efficient object detection models, we define our object detection benchmark, based on this model. To base our discussions on a realistic deployment scenario and provide a real-world motivated benchmark, we use the Airbus Aircraft detection dataset.

In this talk, we further provide the details of a benchmark instance associated with aircraft detection and a particular submission to this benchmark. We further analyze the performance tradeoffs associated with various deployment configurations and performance metrics.

References

[1] Cornelius Dennehy. 2019. A NASA GN&C Viewpoint on On-Board Processing Challenges to Support Optical Navigation and Other GN&C Critical Functions. https://indico.esa.int/event/225/contributions/4249/

[2] Massimiliano Pastena. 2019. ESA Earth Observation on board data processing future needs and technologies. In *European Workshop on On-Board Data Processing (OBDP2019)*. https://indico.esa.int/event/225/contributions/3687/attachments/3357/4395/OBDP2019-S01-03-ESA_Pastena_ESA_Earth_Observation_On_board_data_processing_future_needs_and_technologies.pdf

[3] Yaman Umuroglu, Nicholas J. Fraser, Giulio Gambardella, Michaela Blott, Philip Leong, Magnus Jahre, and Kees Vissers. 2017. FINN. *Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays* (Feb 2017). https://doi.org/10.1145/3020078.3021744