AI-augmented Anomaly Detection Pipeline for Real-Time Onboard Processing of Spacecraft Telemetry

Andreas Koch*[1,2], Michael Petry[1,2], Martin Werner[2]

[1] *Telecom and Navigation Processing Germany, Airbus Defence and Space GmbH.*
[2] *Professorship Big Geospatial Data Management, School of Engineering and Design, Technical University of Munich.*

As the number of sensors and processing capabilities of satellites keep improving, the opportunity arises for a real-time onboard monitoring system to be implemented with the goal of detecting anomalies. In this paper, we identify sets of sensors suitable for monitoring, outline their real-time processing requirements and present an AI-augmented processing pipeline that matches or exceeds these requirements. The pipeline incorporates examples for every component of streaming anomaly detection algorithms including the capability to detect concept drift and autonomously adapt to it.

1 Introduction

As demand for data-driven earth observation services and satellite constellations increases, the number of satellites, their complexity and the amount of onboard components grow. This creates an incentive to reduce operations required for maintenance, thus paving the way for higher automation. One way of achieving a higher degree of automation is improving the awareness of a satellite's surroundings and detecting any effects that may be reflected in its sensors and could be the cause for faults. More specifically, the behavior of many different sensors could be analyzed for any out-of-the-ordinary events and appropriate actions could be performed semi-autonomously, depending on the confidence in identifying the origin of the effect. The potential of detecting faults onboard of satellites has been demonstrated in previous studies [1] [2]. Within the ESA study "ADAP" [2], three faults were presented that occurred due to unforeseen events and circumstances in the space environment. Firstly, unexpectedly low temperatures in the upper atmospheric layers caused an Earth sensor to fail and not pick up on the transition between Earth and deep space. This resulted in a faulty Attitude and Orbit Control Subsystem (AOCS) pitch and roll estimation. Secondly, in one instance outgassing caused a discharge event in the payload subsystem. It was observed that temperatures rose irregularly over a time interval of multiple hours, while still remaining in the temperature range of operation. Thirdly, micro-meteoroid impacts caused a permanent drop in current supplied by solar panel sections. As the satellite is being eclipsed constantly due to many different reasons, it is difficult to correlate this occurrence with the micro-meteoroid impact. The problem addressed in this paper has multiple aspects. First of all, an onboard monitoring in a satellite has to be capable of continuously running for the mission's duration. This includes the capability of adapting to changing circumstances, i.e. different sensor statistics. In satellite operations, the telemetry is regularly reviewed and fault detection, isolation and recovery (FDIR) levels are adapted for any sensor where it is required. Increased automation would mean reviewing telemetry more infrequently or even adapting nominal ranges algorithmically. The analogon for machine learning (ML) based algorithms would be to fine-tune the ML model on a new set of reference data. Therefore, we include the capability of onboard fine-tuning for the ML models within the anomaly detection pipeline presented in this paper, and we employ a strategy of maintaining a set of reference data, which is to be evaluated by a concept drift algorithm at every time step. For more information and an evaluation of these components and strategies, we refer to previous work [3]. In principle, any ML model suitable for streaming anomaly detection could be used in the pipeline. This includes the PCB-iForest [4], an online version of the ARIMA model [5], the FuseAD model [6], the SAND model [7] and the DeepStream model [8] amongst others. Note that the choice of ML models is constrained by supported operations of the AI accelerator, which is employed in the pipeline. Our contribution with this paper is to

- design and inspect the feasibility of a future ML-

*This activity has been funded by DLR within the project Machine Learning for Telecom Satellites (MaLeTeSa). Corresponding author. E-Mail: andreas.c.koch@airbus.com

Subsystem	Parameter	#	Rate [Hz]
AOCS	Temperatures	9	0.01
	Sensor measurements	24	8
	Actuator measurements	13	8
	Internal variables	31	8
Solar Panels	Regulator currents	6	8
	Total regulator power	1	8
	Eclipse flags	1	1
BUS	Battery	6	8
	Power distribution	60	8
	Temperatures	114	0.01
	Status flags	56	1
Other	Temperatures	200	0.01

Table 1: TM parameter groups worth monitoring for anomalies. With an assumed OBC clock speed of 8Hz, the onboard data rates of sensor measurements and electrical signals are significantly higher than in TM packets received on-ground. Temperature measurements should be sampled with a much lower data rate so as not to include temperature differences caused by heaters.

based onboard monitoring system,

- present the amount of telemetry data worth monitoring available onboard a satellite, as well as their processing requirements, how we foresee these requirements to be met and
- provide corresponding runtime measurements on space-grade hardware to support these claims.

The sensor data worth monitoring available onboard a satellite is presented in section 2.1, before the complete anomaly detection pipeline is outlined in section 2.2. The results are laid out in section 3 and are discussed in section 4.

2 Methods

This section will introduce the telemetry (TM) data that is to be processed, with the goal of determining an upper bound of processing requirements for the whole pipeline. Therefore, the number of TM parameters and their onboard data rates are important factors. Secondly, the components of the anomaly detection pipeline are going to be introduced and defined.

2.1 Onboard Data

On the basis of previous work, which identified existing anomaly cases that might occur on satellites in space, a selection of TM parameter groups are presented in Table 1, covering the AOCS and BUS subsystems, along with the solar panels. First of all, temperature measurements are collected across the whole satellite, aiming for a high coverage and the ability to detect any discharge events. As the effect of heaters would be prevalent on temperature readings with a higher sampling rate, a low sampling rate of 0.01Hz is chosen. Insipired by the faulty AOCS pitch and roll estimation, a comparable collection of TM parameters are to be used at an assumed Onboard Computer (OBC) clock speed of 8Hz. This collection covers sensor measurements of the magnetometers, inertial measurement units (IMUs) and star trackers. Actuator measurements of the reaction wheels and magnetorquers are also considered, including reaction wheel speeds and measured currents in magnetorquers. Lastly, the category of internal variables consists of the estimated body frame angles, the overall AOCS position and velocity, the best estimate of the rotational velocity based on IMU and star tracker measurements, the resulting attitude error, both Sun and Earth vector angle errors and other status flags of the sensors. Apart from the eclipse flags, the regulator currents from different sections of the solar panels are considered, as well as the resulting total regulator power, aiming to enable a behavioral analysis with regards to micro-meteoroid impacts. As the satellite bus connects many different components, it already contains a multitude of signals worth including into the overall monitoring. Both the regular as well as the redundant sensors for measuring the battery input and output currents are considered, along with the corresponding charge and discharge power. Furthermore, all output currents being supplied to various components by the power distribution units (PDUs) are monitored, as well as the currents observed in latched current limiters (LCLs). Naturally, including signals which are dependent on commands given, e.g. during maneuvers, raises the question of whether the resulting observed behavior will be identified as anomalous by the overall monitoring system. How to deal with dependent sensor measurements is an open question to be answered during mission planning phases for any specific mission looking to employ behavioral monitoring.

2.2 Pipeline Components

Starting with the sensors and status flags, any measurements to be monitored need to be written to the monitoring module's random access memory (RAM). As sensors and status flags are grouped together to form a TM parameter set targeting the detection of a distinct anomaly and sharing the same data rate, the measurements of the sensors in the same set should roughly be taken at the same time. Thus the process of writing to the monitoring module's RAM can

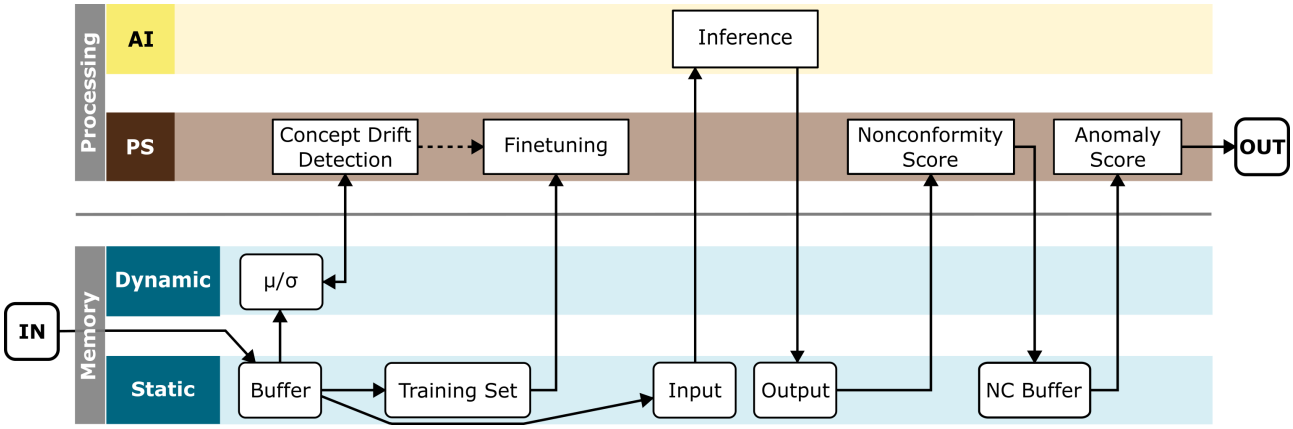

Figure 1: Streaming anomaly detection pipeline depicting the distribution of the pipeline components on static or dynamic memory, the processor system (PS) and the AI accelerator. Finetuning is performed once concept drift has been detected.

also occur at the same time for all TM parameters in a set. Thereby, the new data will be written to two buffers in RAM, a short buffer for inference with a length of the respective ML model input size and a longer buffer for maintaining a representation of the stream statistics for the purpose of concept drift detection. Both buffer sizes are to be seen as parameters of the overall pipeline. Considering the strategies for updating the reference representation presented in [9], the first option to be adopted in the pipeline is the **Sliding Window** strategy, which keeps the m most recent stream vectors in its reference training set $R_{\text{train},t} = \{x_{t-m+1}, ..., x_t\}$ for time step t and a stream vector with the size of its TM parameter set $x_t \in \mathbb{R}^N$,

$$R_{\text{train},t} = \begin{cases} R_{\text{train},t-1} - \{x_{t-m}\} + \{x_t\} & \text{if } t > m, \\ R_{\text{train},t-1} + \{x_t\} & \text{otherwise.} \end{cases} \quad (1)$$

The second option to be incorporated is the **Anomaly-aware Reservoir**,

$$R_{\text{train},t} = \begin{cases} R_{\text{train},t-1} + \{x_t\} & \text{if } t \leq m \\ R_{\text{train},t-1} - \{x_i\} + \{x_t\} & \exists\ i = c(ps, p_t) \\ R_{\text{train},t-1} & \text{otherwise,} \end{cases} \quad (2)$$

$$c(ps, p_t) = \underset{p_j}{argmin}\{p \in ps | p < p_t\},$$

which uses the final anomaly score for every time step to derive a priority p corresponding to the "normalness". The element with the lowest priority is to be removed if the most recent stream vector at time t has a higher priority. With an updating strategy in place, the **"μ/σ-Change"** method is adopted for detecting concept drift in the reference sets [3]. It keeps a running mean and standard deviation and detects concept drift if either of these measures changes to certain extent compared to a previous time step i. Continuing with inference, once new data has been written to the inference buffer, a process is triggered in the processor system (PS) to manage the ML model inference. The actual inference can then be calculated in the PS, although it is worth delegating this calculation to specialized hardware. In general, specialized hardware for AI inference achieve a lower inference time the larger the ML model [10] and the delegation frees up resources on the PS for other tasks. As the target platform in this paper is the AMD Versal AI Core series, we utilize the AMD-Xilinx Vitis AI Deep Learning Processing Unit (DPU) for hardware acceleration. Since the interface to the DPU is located in RAM and consists of an input and output memory region, the aggregated sensor values need to be copied to the input memory region for any specific TM parameter set. As multiple processes, each dedicated to processing new values of a specific TM parameter set, might need access to a shared resource, a lock with mutual exclusion (mutex) is chosen for the implementation. The last two components of the pipeline are a nonconformity score and an anomaly score. We adopt the cosine-similarity based nonconformity score [3] with $\hat{x}_t$ as the ML model's prediction,

$$a_t = 1 - \frac{x_t * \hat{x}_t}{\|x_t\|_2 \|\hat{x}_t\|_2}, \quad (3)$$

as well as the anomaly likelihood [11],

$$f_t = 1 - Q\left(\frac{\tilde{\mu}_t - \mu_t}{\sigma_t}\right), \quad (4)$$

as the anomaly score. Thereby, μ_t and σ_t are the mean and standard deviation of the past k nonconformity scores, $\tilde{\mu}_t$ is the short-term mean of the past k' nonconformity scores with $k' << k$ and finally $Q(x)$ is the Gaussian tail distribution function. The purpose of a secondary anomaly score is to increase its sensitivity in terms of short-term changes of the nonconformity score.

3 Results

Measurements of the execution time for every component in the pipeline are based on an example TM parameter set size of 142 distinct parameters. This set size reflects the maximum set size at 8Hz, while the maximum set size for 0.01Hz (which would be 323) is approximated by a multiple of 3 of the example set size. Table 2 exhibits the execution times of all components and Figure 2 (lower) gives a more detailed overview of the execution times of different sizes of the N-BEATS model [12], which is considered as a reference. It is a general time series forecasting model that can be used as an unsupervised anomaly detection method. The overall pipeline can be split into two processing chains, A and B, containing the anomaly score calculation and concept drift detection respectively. It is apparent that the ML model inference is the bottleneck for the whole pipeline, with an execution time of 6.32ms for the 7.6M parameters model. Furthermore, the DPU resource needs to be shared among different processes, each managing the processing of new values for a specific TM parameter set. Therefore, the number of TM parameter sets the pipeline is able to monitor scales with the number of ML model inferences per second. Figure 2 (upper) depicts the maximum number of sets that can be monitored at the data rates 8Hz, 1Hz and 0.01Hz. As the rest of processing chain A is executed in the PS, it can be distributed on both ARM A72 CPUs for a distributed workload of 0.55ms in the best case. As this is below the minimum inference time of 0.6ms, the processing steps executed in the PS are not the bottleneck of the pipeline for any of the evaluated ML models.

Component	Processing chain	Execution time [ms]
Write to initial buffer	A&B	< 0.01
Write to training set	B	0.01
Update μ/σ	B	0.48
Copy to AI input	A	0.02
AI accelerator inference	A	0.60 - 6.32
Copy from AI output	A	0.02
Calculate nonconformity score	A	0.34
Write to NC buffer	A	< 0.01
Calculate anomaly score	A	0.72
Write pipeline output	A	< 0.01

Table 2: Measured execution times of all pipeline components for an example TM parameter set of 142 distinct parameters on the target AMD-Xilinx Versal AI Core series. Note that all steps except the AI accelerator inference utilize the processor system (PS).

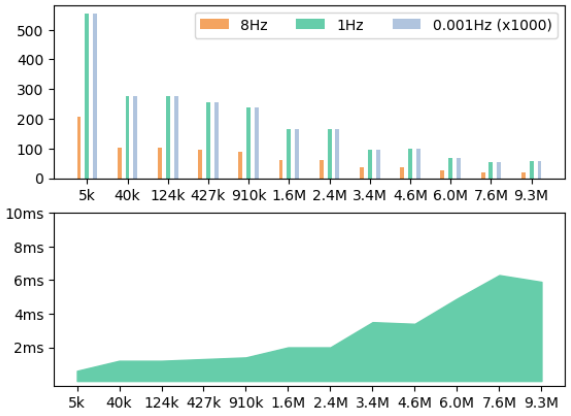

Figure 2: Inference times and corresponding maximum number of TM parameter sets for the reference N-BEATS model of various sizes. Measurements were performed for a window of 100 time steps and 142 channels. With a throughput of 20 example TM parameter sets at 8Hz for the largest model, the processing requirements are more than met.

4 Discussion

In this paper, we outlined a pipeline for performing streaming anomaly detection onboard of spacecraft. Based on previously observed anomalies and missions, relevant categories of TM parameters are identified, together with the number of parameters and their data rates. The distribution of the pipeline components on the target hardware and corresponding execution times are presented. With a maximum of 20 example TM parameter sets at 8Hz, which the pipeline is able to monitor in parallel, the overall data rate requirements are more than met even for the largest evaluated ML model. This is mainly due to fast inference times of the DPU, making the overall pipeline scalable for monitoring many TM parameter sets, each targeting the detection of a different anomaly. As the number of sensors and processing capabilities on spacecraft are improving, there is an opportunity for employing behavioral onboard monitoring systems. Since this paper focused on execution times for different components of the anomaly detection pipeline, future work can extend the scope to power requirements and measurements of the pipeline, investigate the quality of resulting detected anomalies and explore the time and resources required for onboard finetuning of ML models. Overall, the implementation of an AI-augmented anomaly detection pipeline might increase the reliability and automation of spacecraft, while providing the opportunity to continuously improve the anomaly detection capability.

References

1. Delande, M. P. IAF SPACE OPERATIONS SYMPOSIUM (B6) Innovative Space Operations Concepts and Advanced Systems (2). en.
2. Koch, A. *et al. On-Board Anomaly Detection on a Flight-Ready System* in *2023 European Data Handling Data Processing Conference (EDHPC)* (2023), 1–4.
3. Koch, A., Petry, M. & Werner, M. Extended Framework and Evaluation for Multivariate Streaming Anomaly Detection with Machine Learning, 1–8 (May 2024).
4. Heigl, M. *et al.* On the Improvement of the Isolation Forest Algorithm for Outlier Detection with Streaming Data. *Electronics* **10.** ISSN: 2079-9292. https://www.mdpi.com/2079-9292/10/13/1534 (2021).
5. Liu, C., Hoi, S. C., Zhao, P. & Sun, J. *Online arima algorithms for time series prediction* in *Proceedings of the AAAI conference on artificial intelligence* **30** (2016).
6. Munir, M., Siddiqui, S. A., Chattha, M. A., Dengel, A. & Ahmed, S. FuseAD: Unsupervised Anomaly Detection in Streaming Sensors Data by Fusing Statistical and Deep Learning Models. en. *Sensors* **19,** 2451. ISSN: 1424-8220 (May 2019).
7. Boniol, P., Paparrizos, J., Palpanas, T. & Franklin, M. J. SAND: streaming subsequence anomaly detection. en. *Proceedings of the VLDB Endowment* **14,** 1717–1729. ISSN: 2150-8097 (June 2021).
8. Harush, S., Meidan, Y. & Shabtai, A. DeepStream: Autoencoder-based stream temporal clustering and anomaly detection. en. *Computers Security* **106,** 102276. ISSN: 01674048 (July 2021).
9. Calikus, E., Nowaczyk, S., Sant'Anna, A. & Dikmen, O. No free lunch but a cheaper supper: A general framework for streaming anomaly detection. *Expert Systems with Applications* **155,** 113453. ISSN: 0957-4174. https://www.sciencedirect.com/science/article/pii/S0957417420302773 (2020).
10. Petry, M. *et al. Accelerated Deep-Learning Inference on the Versal adaptive SoC in the Space Domain* in *2023 European Data Handling Data Processing Conference (EDHPC)* (2023), 1–8.
11. Ahmad, S. & Purdy, S. Real-Time Anomaly Detection for Streaming Analytics. http://arxiv.org/abs/1607.02480 (July 2016).
12. Oreshkin, B. N., Carpov, D., Chapados, N. & Bengio, Y. *N-BEATS: Neural basis expansion analysis for interpretable time series forecasting* en. in (Mar. 2022). https://openreview.net/forum?id=r1ecqn4YwB.