35th Space Symposium, Technical Track, Colorado Springs, Colorado, United States of America
Presented on April 8, 2019

A Machine Learning Framework for Space Missions

Zhenping Li
Arctic Slope Technical Services (ASTS), Lanham, MD. 20706

ABSTRACT

A machine learning (ML) framework for space missions has been developed to creates situational awareness and enables anomaly detection, dynamic data filtering and sensor data quality assessments. The ML framework covers four areas; the ML architecture model, ML algorithms and formulism for satellite datasets, a scalable and extensible ML platform, and the ML application portfolio. The ML architecture model defines how a ML system interacts with space and ground assets, ML functionalities, the interfaces among different ML processes, and the ML framework operational concept. ML algorithms and formulism includes data representation, data training, anomaly detection and characterization, and the creation of actionable information for mission/Enterprise situational awareness. The ML platform is a software implementation of a ML architecture model that addresses the challenges of ML solutions in operational environments. The ML application portfolio focuses the applications of the ML framework to the health and safety of satellites and onboard instruments with differing orbital characteristics. The Advanced Intelligent Monitoring System (AIMS) implements the ML platform to provide a common infrastructure and services, while ML algorithms are treated as plug-and-play components. The applications of the ML framework the Geostationary Environment Operational Satellite(GOES) instrument data, and the health and safety data for the Suomi National Polar-orbiting Partnership (NPP) are presented, and it shows that the ML framework bring fundamental advances in maintaining the health and safety of space missions. The ML approach enables early anomaly detection, rapid turnaround in anomaly troubleshooting, and significant improvement in system resiliency.

1. INSTRODUCTION

A satellite and its onboard instruments are dynamical systems with states that are time dependent and generally non-deterministic. The amount of data representing the states of the system (or its subsystems or components) could be large in number of datasets and volume (on the order of gigabytes or more per day for example) to make it impossible to perform manual analysis to determine the system operational state. The challenge is how to obtain actionable information from a highly complex system generating large volumes of data in near real-time. The focus of this paper is to present a ML framework that creates situational awareness enabling automated engineering analysis, resulting in fundamental advances in how the health and safety of a highly complex system with a large number of sensors are maintained.

Situational awareness for a dynamical system is defined as the ability to perceive, analyze and predict its own behavior. Machine Learning provides a natural platform to create situational awareness by establishing data models through data training, to predict near-term behaviors. The ability of a system to predict its own expected behavior enables anomaly detection, dynamic data filtering, and sensor quality assessment. The theory of situational awareness[1] also provides insights into how a ML system interacts with its managed systems, such as a space mission with both space and ground assets, which leads to the architectural model for a ML system. The ML architecture model defines the ML functionalities , interfaces among ML processes, and the operational concepts.

There are considerable challenges in developing and executing ML algorithms in an operational environment due to the large number of datasets and large data volume required for processing (in near real-time). Furthermore, the data used in data training could be defective or anomalous that distort the training outcome. The data training process for a ML system in operational environments must be efficient while maintaining the accuracy of the data training outcome to prevent the distortion of training outcomes from defective data points. This requires flexibility in selecting data models and training algorithms to improve data training efficiency and a systematic approach to handling defective data points. Thus, an effective ML system for space missions must be scalable to handle large data volumes, flexible in selecting different ML algorithms for analyzing specific data patterns, and extensible to

address mission specific requirements. The ML platform is a software implementation of a ML architecture model that separates mission-specific and data pattern-specific logic from the logic common to all missions (within the aerospace domain). The algorithms specific to certain data patterns and the processes required to meet mission specific requirements are treated as plug-and-play components deployed within the ML platform. The ML application portfolios refer to the applications of the ML framework for specific missions, and each mission has its own data characteristics that may require different ML algorithms. The different orbital characteristics in space missions, such as Low Earth Orbit (LEO), Medium Earth Orbit (MEO) or Geosynchronous Earth Orbit (GEO), produce different data patterns. The application of the ML framework has been applied to the GOES instrument data[2,7] and Polar satellite health and safety data with considerable success.

This paper is organized as follows: Section 2 presents the ML architecture model for space missions; Section 3 shows the general formulism in data training, anomaly detection and characterization, and post training analysis; Section 4 focuses on the ML platform implementation; Section 5 presents the applications of the ML framework developed for GOES instrument data and the spacecraft housekeeping telemetry data for the NPP satellite; Section 6 provides the summary and future outlook of ML applications for space missions.

2. THE ML ARCHITECTURE MODEL FOR SPACE MISSIONS

Figure 1 shows the ML architecture model. The managed element represents a dynamical system characterized by its state variables $\{S_j(t_i)\}$, which are time dependent. The datasets $\{d_j(t_i)\}$ are the measurements of the state variables $\{S_j(t_i)\}$, which are noisy and follow the Gaussian probability distribution. The ML architecture model is based on the theory of situational awareness for dynamical systems[1] that defines how a ML system interacts with its managed element(s), and it is an information loop between a ML system and its managed element. The datasets from a dynamical system are monitored, analyzed, and appropriate actions are taken to change/enhance system behavior ensuring that system performance meets mission objectives. Situational awareness of the managed element is achieved through data training on existing data and establishes data models to predict near-term system behaviors, and enables dynamical monitoring for anomaly detection, dynamic data filtering, and data quality assessment.

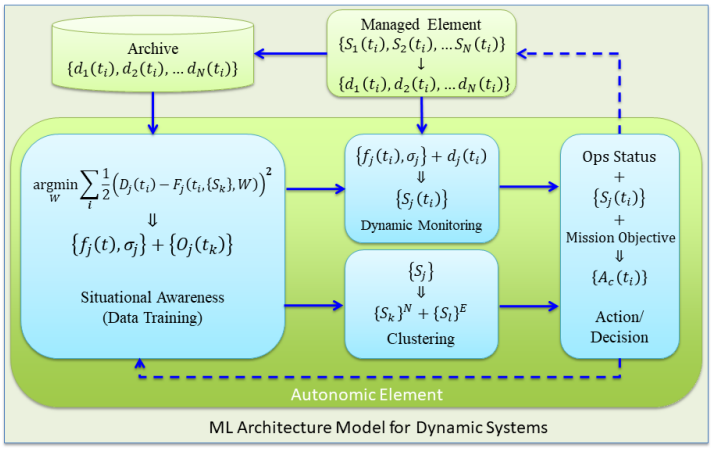

Figure 1 The Architecture Model of the ML System

The data models are trained with "normal" data, where "normal" data refers to data that was assessed for quality, containing few or no errors/anomalies. Since datasets with Gaussian probability distribution form a tight data bound, these data sets are highly sensitive to deviations from their expected behavior above the calculated noise level. The dynamic monitoring function compares the values of datasets with predictions of their data models. An anomaly in a dataset is defined as the unexpected change from its normal data pattern, which can be detected through either dynamic data monitoring in real or near real-time and or a post training analysis process.

Data training outputs allow dynamic data filtering to determine the actual values of state variables $\{S_j(t_i)\}$ by combining values of incoming datasets with predictions of ML data models. The Kalman filter is a dynamic filter widely used in space missions for determining spacecraft orbit characteristics, where the spacecraft orbit is predicted by classical orbital mechanics. The application of a Kalman filter with ML algorithms to predict the expected behavior of state variables has not been fully investigated. Additionally, a potential application of the dynamic filter is in

instrument data calibrations, where the calibration coefficients are derived from the observations in spacelooks and internal targets that are generally noisy. The ML algorithms can be used to predict its near future behavior.

The post training analysis process is performed after data training in each session to obtain actionable information from a large amount of data training outputs which include data quality and system operation status. Furthermore, long-term trending can also be performed in the post training analysis process to investigate system degradations and make predictions on when a system is expected to fail. Clustering techniques are generally implemented in post-training analysis, and these techniques are performed with data quality metrics as its attributes.

The operations concept for a ML system is shown in Figure 2. A training session generally covers a period long enough so that input training sets contain sufficient information for predictions of near-term behaviors. Training sessions are repeated periodically, which enables data models to adapt to long term or seasonal pattern changes. Since daily changes in data patterns are small, the retraining of datasets is a minor adjustment of the training output from previous training sessions. Data training for current sessions uses the output from previous training sessions as the input. This is particularly important for data training of nonlinear data models, such as neural networks; data retraining on existing data models makes the training algorithm much more efficient and enables data training for operational environments (near real-time processing). The overlaps between two consecutive training sessions ensures the stability of training outcomes. There are three processes running consecutively in each training session: data training, post training analysis, and report generation. The report generation process provides reports on data training output and post training results, and the content and format of data training results are generally mission specific. The real or near real time monitoring of data sets is performed continuously, and the data models used in the monitoring process are updated periodically by data training output.

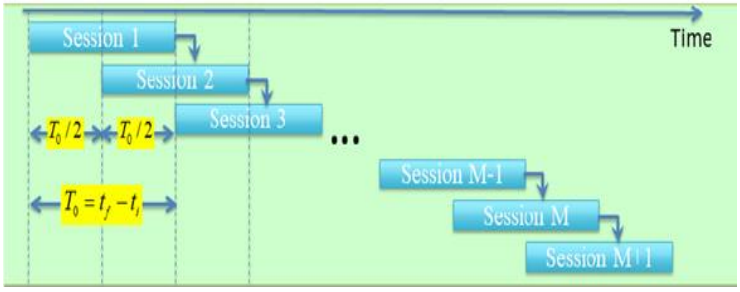

Figure 2 Data Training Operational Concepts

3. ML ALGORITHMS AND FORMULISM

A dataset $\{d_j(t)\}$ corresponding to the state variable $\{S_j(t_i)\}$ is characterized by its time dependent trend $\{s_j, \sigma_j\}$, which consists of a time dependent function s_j

$$s_j = f_j(t - t_0) \tag{1}$$

representing the actual state value at time t, and a standard deviation

$$\sigma_j = \sqrt{\frac{1}{N}\sum_i \left(d_j(t) - S_j(t_i - t_0)\right)^2} \tag{2}$$

for the noise level. The noise level σ_j represents the data quality of a dataset $d_j(t)$ that provides important insights into its data quality and operational status. The time dependent trend $\{f_j(t - t_0), \sigma_j\}$ is obtained in ML framework by training the time dependent function $f_j(t - t_0)$ with the dataset $d_j(t)$. Data training for the datasets with Gaussian probability distribution follows the least square fitting routine with respect to the parameter set $\boldsymbol{W}$:

$$\underset{W}{argmin} \sum_i \frac{1}{2}\left(D_j(t_i) - F_j(t_i - t_0, \{S_k\}, W)\right)^2 \tag{3}$$

where $D_j(t)$ and $F_j(t - t_0, \{S_k\}, W)$ are the corresponding training set for datasets $d_j(t)$ and the data model for the time dependent function $f_j(t)$ in the data training space. A transformation between $D_j(t)/F_j(t - t_0, \{S_k\}, W)$ and $d_j(t)/f_j(t)$ is generally needed for data models within the ML framework and with actual datasets with arbitrary scales. The data training function finds a parameter set *W* so that the error function, Eq. 3, is minimized. This type of problem is generally referred to as a regression problem in ML.

There are many data models in the ML framework for time dependent trends, which can generally be categorized in two groups: linear and nonlinear data models. Data training for linear models is generally much more efficient than data training for nonlinear models, while the nonlinear models, such as for neural networks, generates more accurate training outcomes especially for a dataset with a high noise level. The implemented data models include the Fourier expansion model[2] and neural networks[3,4,5]. The Fourier expansion data model is a linear model, where the time dependent function $F_j(t-t_0, W)$ has a linear relationship with the parameter set $\boldsymbol{W}$. A data training algorithm for the linear model is the linear least square fit algorithm. The Fourier expansion model is limited to data patterns with dominant low frequency components. Neural network models are nonlinear models, and data training algorithms are generally based on the gradient decent approach. The local adaptive gradient decent algorithm[6] is implemented for neural networks, which provide significant improvements in accuracy as well as efficiency over the standard gradient decent approach within training outcomes. The universal approximation theorem shows that the neural networks model generally provides a good description for all data patterns including the non-continuous ones, however, the data training algorithm for neural networks is always less efficient than that for linear data models. Therefore, the linear model is generally preferred for data patterns with dominant low frequency components to improve data training efficiency. Multiple data models, including both linear and nonlinear models, are implemented for datasets within the same system to ensure data training efficiency and accuracy. More detailed discussions of data training for the Fourier expansion model and neural networks are provided in Ref. [5].

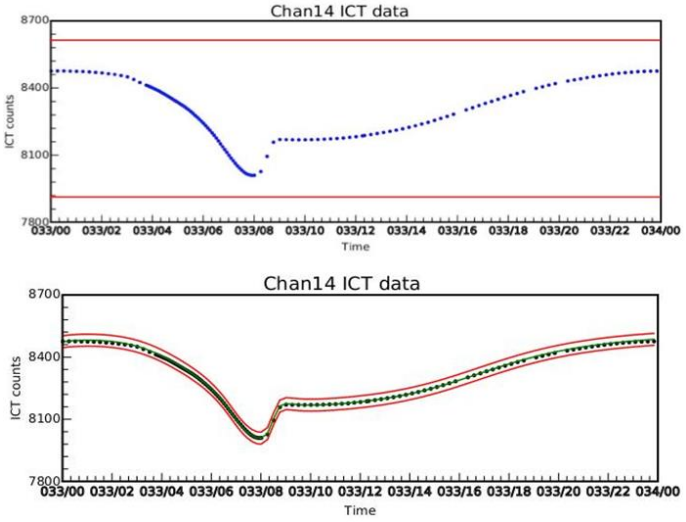

Figure 3 The top plot shows the data monitoring with the static limit and the bottom plot represents the dynamic monitoring based on the ML outputs. The red lines in the bottom plot are the data bound defined by Eq. 4.

The time dependent trend for a dataset with a Gaussian probability distribution defines a data bound with values that satisfy the following relationship:

$$\left|f_j(t_i - t_0) - d_j(t_i)\right| < N\sigma_j \tag{4}$$

where *N* is a user defined threshold parameter. There are generally two threshold parameters, N^W and N^E. These parameters are defined for warning and error assessments respectively, which correspond to the red and yellow limits in an existing telemetry data monitoring application/system. Figure 4 highlights the difference in data monitoring between the static limit monitoring approach and the dynamic monitoring in the ML approach. The narrow data bound determined by the time dependent trend is much more sensitive to deviations above data noise levels. Thus, deviations from the existing data pattern in a dataset can be detected much earlier in a ML framework than from static limit monitoring. Some deviations in data patterns may still be measured within the defined static limits, however these data points cannot be detected by a typical limit monitoring function.

A data point with a value outside the data bound defined in Eq. 4 is regarded as an outlier. While an isolated outlier does not impact the data quality, consecutive outliers may indicate a persistent data pattern change due to a potential anomaly. An anomaly in a dataset corresponds to the **data pattern change** in ML framework, which is represented by the persistent consecutive outliers in a dataset. An outlier for a dataset is quantitatively defined as[7]

$$O\left(d_j(t_i)\right) = N\left(\frac{f_j(t_i) - d_j(t_i)}{\sigma_j} > N\right) \tag{5}$$

where the quantity $\left(\frac{f_j(t_i)-d_j(t_i)}{\sigma_j} > N\right)$ is a Boolean variable, and *N* is a threshold value for an outlier. These persistent outliers form an outlier cluster along the time axis corresponding to the data pattern change. The metric that measures the data pattern change is defined as:

$$\chi_j^O = \sum_i \left(\frac{\delta_i^W}{T}\right) + \frac{N^E}{N^W}\sum_i \left(\frac{\delta_i^E}{T}\right) \quad (6)$$

where δ_i^W and δ_i^E are the period for the warning and error outliers respectively, and N^W and N^E are the warning and error threshold parameters. The parameter T in Eq. 6 represents the time scale of persistent outliers determined by the sampling frequency of datasets. The metric χ_j^O depends on the period of the consecutive outliers, and the quantity $\frac{N^E}{N^W} > 1$ represents the outlier severity. The χ_j^O value has the range $0 \le \chi_j^O < \infty$, and $\chi_j^O = 0$ for the datasets with an isolated outlier with a period of zero. A dataset with a normal operational status should have zero or a small χ_j^O value. The metric χ_j^O is used in both real-time data monitoring and post-training analysis for anomaly detection and data quality assessment.

There are two additional metrics[7], temporal and spatial changes, used in the post training analysis in addition to the pattern change metric in Eq. 6, which measures data quality based on data training outputs. The temporal change metric measures changes in standard deviation $\boldsymbol{\sigma_j}$ of a dataset between two consecutive training sessions. The spatial change represents the relative quality of a dataset by comparing its standard deviation $\boldsymbol{\sigma_j}$ with those that have the same scale and similar data patterns. The spatial change metric is very useful for sensors within the same spectral channels for sensor quality evaluations. More detailed discussions of these metrics are presented in Ref. [7].

4. A FLEXIBLE, SCALABLE AND EXTENSIBLE ML PLATFORM

A ML system in an operational environment has considerable challenges, which include: large numbers of datasets, very large data volumes, diverse data types, and defective datasets with outliers that affect data training outcomes. Data training algorithms in operational environments must be very efficient while maintaining the accuracy of data training outcomes. Flexibility in selecting different data models for datasets with different data patterns is critical. Furthermore, the application of a ML framework to different domains requires different post training algorithms. A scalable, extensible and flexible architecture for a ML system in space missions is essential. The enterprise architecture for ML systems separates the common logic from the mission-specific and data pattern-specific logic. The ML platform provides common software services and infrastructure, while mission specific and data pattern specific logic are implemented as plug-and-play components. The ML platform also provides processing management to allocate sufficient computing resources for ML processes that are very computationally intensive. A well-designed ML framework allows engineers to focus on ML algorithms specific to the domain datasets, eliminating or reducing the need to expend resources for logistics and infrastructure functions that are required for setting up complex ML applications. A well-designed and robust ML platform will significantly reduce the development cycle, risk, and cost of a ML application. Figure 4 shows the ML enterprise architecture based on the ML architecture model with component (top), function, service, and infrastructure (bottom) layers, which provides well-defined functions, interfaces, and a simple and structured operational concept.

The ML architecture model in Figure 1 provides well-defined input/output content for each component, which ensures that a standard API can be defined. The service, function, and infrastructure layers in the enterprise architecture provide the common logic for all machine learning applications, while the component layer provides the flexibility to select the algorithm component for specific data patterns, and the extensibility to meet the mission specific requirements. The data input components provide the flexibility to access data in different formats from raw telemetry data and instrument calibration data. The post-training analysis functions implement clustering techniques within the data quality metrics space. The domain specific algorithms for long term trending or predictive maintenance can also be implemented in the post training analysis processes. Different missions or domains may

have different requirements for the implementation of data quality metrics and clustering techniques. Report generation components ensure the needed flexibility to meet specific and/or unique requirements in content and format.

The common ML services include:

- Class registry service links the component names with the actual class object that can be invoked by the processes in the function layer. Each component is registered in the class registry, and each datasets defined in the application database have attributes that identify which component is being used in ML processes.
- Database provides data definitions and attributes needed for data training including the component names for algorithm and post training analysis.
- Configuration service provides overall global configurations such as the data path for the data archive locations, and mission specific configurations can be defined and accessed by each component via the configuration service.
- Event message service provides consolidated event message handling, which can be accessed by components to publish event messages within the framework.

Figure 4 The ML Enterprise Architecture with component (top), functional, service, and infrastructure (bottom) layers.

The common infrastructure provided by the ML platform includes

- The archive provides permanent storage for data training output. Data models used in ML are represented by a set of parameters so that data models can be archived and reconstructed.
- The client service provides the capability to show data training results, data monitoring status, and post-training analysis results in various formats, such as data plots, timeline bar chart, and html files. A common GUI interface is also provided for manual data training, critical for a ML system during deployment.
- Ground system interface enables a ML system to publish event messages to a common message bus within the ground system enterprise and to provide ML services to other applications allowing the existing ground system to leverage capabilities of the ML application and framework. Examples of ground system enterprise standards/capabilities include the Goddard Mission Services Evolution Center (GMSEC) reference architecture[8] and the future US Air Force Enterprise Ground Services (EGS) standard.

The enterprise architecture in Figure 4 has been implemented in AIMS[5,7]. The APIs for the components in Figure 4 have been defined and implemented in AIMS, and the component deployment procedure has been developed.

5. ML FRAMEWORK APPLICATION PORTFOLIO

The initial application of the AIMS ML approach focused on instrument datasets from the GOES spacecraft. AIMS is deployed for monitoring the instrument calibration performance of the Advanced Baseline Imager (ABI) on GOES-R. The ABI instrument has 6 visible and 10 infrared (IR) channels with 7000 active detectors. There are multiple variables or data points defined for each detector used in instrument calibration, which leads to more than 30,000 datasets to be trained and monitored daily by AIMS. Figure 5 shows an example of anomaly detection by AIMS. The dataset shown in Figure 5 is the detector internal calibration target (ICT) counts of an infrared channel, which is used as the input for instrument data calibrations. Anomalies in this dataset will directly impact instrument data quality. The two red lines in the plot are the upper and lower data bounds based on Eq. 4. The detector became anomalous around 10:00Z on 364 in 2017, where the values become flat. This represents a detector that is no longer responsive to temperature changes in the remote sensing area of the globe that is being measured, which leads to a stripping phenomenon of ABI-captured images. This anomaly can be detected in real-time or near real-time depending on

instrument data latency. The sensor data pattern change (from nominal) would not be detected through static limit monitoring, since these data values within the anomalous period are still within the static limits. The anomaly is also detected through post training analysis since the changes within the data pattern generates highly elevated values in both spatial and temporal change metrics.

It may take hours or days to find the root cause of an anomaly using a typical engineering analysis (human and non-ML computing capabilities), while it takes only minutes or less for AIMS to find and determine which sensor failed.

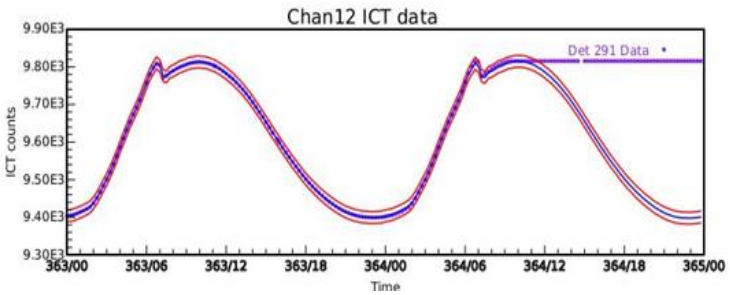

Figure 5 The ICT Counts for the detector 291 in ABI channel 12. The purple points are the actual data values, and the blue line is the prediction from the data model. The two red lines are the upper and lower data bound defined in Eq. 4

The application of an ML framework and applications such as AIMS on polar orbiting satellites, sand low Earth orbit (LEO) satellites is ongoing. Figure 6 shows the AIMS data training output for the NOAA Suomi NPP power system, which is a LEO satellite with very different orbit characteristics compared to GEO missions (such as GOES-R). All three datasets are implemented with the same simple neural network with two hidden layers. The data training output (blue line) shows remarkable accuracy with the actual data points (red dots). The key difference in the daily data training operation between the datasets for GEO and LEO satellites is the reference time $\boldsymbol{t_0}$ in the time dependent function $\boldsymbol{s_j = f_j(t - t_0)}$. The reference time $\boldsymbol{t_0}$ must correspond to the same point in a given data pattern. The GEO satellite datasets generally show a diurnal pattern so that the reference time can be set at the start of day, while LEO satellite datasets, such as the NPP dataset, show a repeatable pattern in one satellite orbit period, which is around 1.7 hours. The time for the equator crossing point is generally selected as the reference time for LEO satellites, which can be obtained generally from the spacecraft ephemeris telemetry or the two-line element propagation calculation.

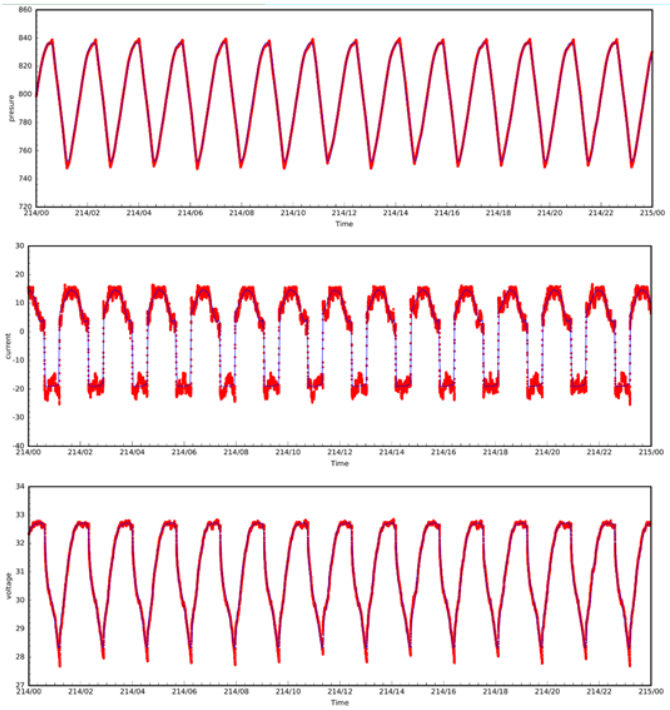

Figure 6 Data training output for datasets in NPP power system. The red dots are the telemetry data, and the blue lines are the data training output. The datasets are the pressure (top), voltage (middle), and the current (bottom).

The temporal change and outlier cluster metrics[7] can be used for data quality evaluation and anomaly detection in the post training analysis for telemetry data. In addition to data quality metrics, the domain specific knowledge is generally needed to evaluate the operational status (nominal or anomalous) of a given subsystem, such as the power or navigation systems. The ML framework and application for analyzing and monitoring LEO satellites spacecraft health and safety is in progress.

6. SUMMARY AND OUTLOOK

A robust and flexible ML framework brings fundamental advances in system resilience and maintaining the health and safety of spacecraft and its onboard instruments; datasets are monitored via data pattern changes rather than static limit violations. Data quality metrics provide quantitative measurements of data pattern changes, and they become actionable information in anomaly detection, data quality assessments, and system operational status evaluations. The ML architecture model based on the theory of the situational awareness for dynamic systems provides well-defined functions/capabilities, operations concepts, and interfaces between ML processing blocks. The scalable, extensible and flexible ML enterprise architecture allows engineers to focus on the development of algorithm components for specific applications, which significantly reduces ML application development cost and schedule. A robust and flexible ML framework with a well-documented and functioning API, plug-and-play algorithm platform (ML platform as a service) and deployment approach can be evolved into an industry standard for ML applications.

The ML framework presented here can be extended to missions that contain a very large number of sensors with complex operations and large data volumes. Flight operators, with typical software tools, would find it impossible to retrieve actionable information from these datasets because of their very large data volume with today's telemetry assessment tools (e.g. trending and analysis and custom tools).

Our initial experience from deploying AIMS for NOAA/GOES-R radiometric operations shows that an ML approach significantly improves system resiliency; it enables early anomaly detection, much more rapid turnaround in troubleshooting and sensor quality assessment, which is not possible, or more time-consuming, using a traditional engineering analysis approach. AIMS is a low risk and low-cost ML solution that provides considerable benefits to current and future space missions.

Reference

[1] Mica R. Endsley, "Toward a Theory of Situation Awareness in Dynamic Systems", Human Factors, 1995 The Journal of the Human Factors and Ergonomics Society 37(1):32-64.

[2] Zhenping Li, David Pogorzala, Ken Mitchell, J.P. Douglas, "Adaptive trending and limit monitoring algorithm for GOES-R ABI radiometric parameters" GSICS Quarterly Newsletter, Summer 2015 Issue, http://dx.doi.org/10.7289/V5XK8CHN#page9

[3] Zhenping Li, J.P. Douglas, and Ken Mitchell, "Creating Situational Awareness in Satellite Data Trending and Monitoring", 32nd Space Symposium, April 11-12, 2016, Colorado Springs, Colorado. http://www.spacesymposium.org/tracks/technical-track/papers.

[4] Zhenping Li, "Creating Situational Awareness in Spacecraft Operations with the Machine Learning Approach", 2016 AMOS Conference, September, 20-23, Maui, Hawaii. http://www.amostech.com/TechnicalPapers/2016/Poster/Li.pdf.

[5] Zhenping Li, "Developing Machine Learning Solutions for Spacecraft Ground Systems", Proceedings of 6th International Conference on Space Mission Challenges for Information Technology (SMC-IT), Alcala de Henares, Spain, September 2017. https://smcit.ecs.baylor.edu/data.html.

[6] M. Riedmiller, "Advanced Supervised Learning in Multi-Layer Perceptrons –From Backpropagation to Adaptive Learning Algorithms", Computer Standards & Interfaces, 16 (1992), 265, http://dx.doi.org/10.1016/0920-5489(94)90017-5.

[7] Zhenping Li, Ken Mitchell, Biruh Tesfaye, and Data Pogorzala, "Monitoring GOES-R Advance Baseline Imager(ABI) Radiometric Performances with a Machine Learning System", CALCON 2018 Conference Proceedings, June, 2018. Logan Utah. To be published in July 2018.

[8] See https://gmsec.gsfc.nasa.gov/ for the detailed information.