A Reusable Framework for Fault Detection and Isolation in Small Satellites

Jubilee Prasad Rao, John Pace, Jesse Williams, Ryan Mackey, Liang He, *Senior Member, IEEE,*

Abstract—Nearly half of all small satellites launched between 2000 and 2016 have experienced partial or complete failures. Detecting the anomalies and faults responsible for such failures and responding to them rapidly may help increase the success rates of future small satellite missions. However, developing and implementing platform-specific and comprehensive fault management solutions can be cost-prohibitive to most small satellite teams. To enable such teams to achieve this capability quickly and cost-efficiently, we have developed a reusable and fully data-driven framework and associated algorithms to detect and isolate anomalous behaviors. This work is supported by NASA and utilizes correlations between system operational variables and Machine Learning (ML) techniques to generate real-time estimates of expected nominal behavior. Large differences between expected and observed behavior captured through sensor measurements may indicate the presence of anomalies or faults. Since this approach monitors sensor streams for new behavior, a faults database is not required, and anomalies or faults not previously known are also expected to be detected.

To automate this framework, a set of algorithms are developed that use a small amount of normal operational data from a satellite system (or a subsystem) to train the required models. The algorithms utilize physical dependencies between the system's operational variables that are extracted using a Dynamic Time Warping (DTW) technique and ML regression models. The system's battery metrics, current and voltage, are considered the roots of trust to diagnose other system operational variables selected based on the DTW correlation strengths. Battery metrics are selected as they can be independently and reliably measured. Fluctuations in a rolling window of battery current and voltage measurements are extracted into features such as window average, window maximum, window minimum, and several others. These features (predictors) and the individual sensors' readings (targets) are then utilized for training the ML regression models. During a mission, the same battery features are extracted in real-time and fed to the trained ML models to estimate sensors' measurements expected during nominal system behavior. Then, the cumulative error between predicted and observed measurements and its slope are calculated. An anomaly flag is raised when these two values cross dynamic thresholds computed based on their recent values and some preset weights. Due to the one-to-one nature of the independent mappings from battery metrics to each operational variable, the anomaly is also simultaneously isolated to the sensor itself or the subsystem where it is located. The framework also includes automated testing of the trained ML models and anomaly detection parameters selected by artificially injecting different types of anomalies. The injected anomalies relate to loose connections, abrupt sensor failure, sensor drift, data corruption, and others. In this work, the implementation of this framework on datasets generated from laboratory tests on a CubeSat platform is discussed. Results show nearly 90% average detection rate and less than 1% average false positives rates for many analog operational variables strongly correlated to battery metrics.

Keywords—small satellites, anomaly detection, batteries, data-driven methods, reusable framework, sensor faults, cyber-intrusion, data corruption

1 INTRODUCTION

Small satellite technology is revolutionizing the space industry with a variety of missions spread across scientific, defense, and commercial applications [1]. Ensuring that these systems operate reliably with minimal interruptions is a concern among mission developers and operators. Anomalies or unexpected events can occur in the different satellite subsystems due to various environmental factors, hardware or sensor faults, software bugs, and maybe even cyber intrusions [2]. These may lead to immediate or eventual failure of subsystems and also entire missions.

Accurately and promptly detecting and mitigating such anomalies is very critical to the successful execution of missions and managing operational costs. Traditional methods for anomaly detection rely on predetermined thresholds or simple rule-based systems [3]. More recently, physics model-based fault detection methods [4] are also popular where the output from the simulation of a subsystem is compared against observed behaviors to detect discrepancies and the associated faults. Developing these methods requires extensive testing and knowledge of the different subsystems and expected values during operation. Still, these fall short of capturing complex and evolving anomalies. There also exist circumstances in which anomalies are very subtle and cause the affected operational variable to change only slightly which do not breach the preset thresholds. In addition to the technical limitations of these methods, resources and time constraints make it challenging to develop accurate thresholds and rules for each platform [5]. Developing and implementing platform-specific and comprehensive fault management solutions can be especially cost-prohibitive to most small satellite teams.

• *J. Pace, J. Prasad Rao, and J. Williams are with Global Technology Connection Inc, Atlanta, GA, USA, R. Mackey is with JPL, NASA, CA, USA, and L. He and J. Marinelli are with the University of Colorado Denver, CO, USA.*

It has been reported that nearly half of all small satellites launched between 2000 and 2016 have experienced partial or complete failures [6]. Detecting the anomalies and faults responsible for such failures and responding to them rapidly may help increase the success rates of future small satellite missions. To achieve this, several data-driven approaches have been proposed and/or have been implemented for satellites. One such method utilized operational data and its telemetry logs from a satellite to extract reboot labels and identify outliers that were observed before such reboots. Then, using statistical methods, their timestamps were used to predict the probability of future faults [7]. One approach utilized Recurrent Neural Network (RNN) [8] and several time series inputs to detect anomalies. A general-purpose data-driven monitoring system was developed [9] that used clustering techniques to calculate a distance metric that indicated how different a new vector representing the current state of a system was from nominal behavior. Another method utilizing machine learning also involved a classifier method after the data was reduced using principal component analysis [10]. Another machine learning-based method [11] utilized an unsupervised artificial neural network method for dimensionality reduction followed by a clustering approach to detect outliers. Then, a supervised approach is utilized to identify the small set of faults included in the data. The methods described here utilized multiple operational variables or their time series data as input, needed reboot/fault labels from existing methods, and were difficult to interpret for operators to generate mitigation strategies.

To overcome these challenges, and enable small satellite teams to achieve this capability cost-efficiently, we propose a reusable and fully data-driven framework and associated algorithms to detect and isolate anomalous behaviors. The uniqueness of this approach lies in abstracting a system as a graph of nodes (operational variables) that are connected to others through edges. Here, an edge connecting two nodes represents a strong correlation and also an ML model that can predict the behavior of one node by using features from only the other node. This approach makes this framework very viable for the interpretation of anomalies. Overall, this approach utilizes correlations between system operational variables, feature extraction methods, and Machine Learning (ML) techniques to generate real-time estimates of expected nominal behavior [12]. Large differences between ML model predictions and observed behaviors may indicate the presence of anomalies or faults. Since this approach monitors sensor streams for new behavior, a faults signature database is not required, and anomalies or faults not previously known can also be detected.

To automate this framework, a set of algorithms are developed that use a small amount of normal operational data from a satellite system (or a subsystem) to train the required models. The algorithms utilize physical dependencies between the system's operational variables that are extracted using a Dynamic Time Warping (DTW) technique and ML regression models. The system's battery metrics, current and/or voltage, are considered the roots of trust to diagnose other system operational variables selected based on the DTW correlation strengths. Battery metrics are selected as they can be independently and reliably measured but a different variable can also be selected instead. Features (predictors) are extracted from battery measurements and different individual operational variables (targets) are utilized for training the ML regression models. The trained ML models generate their estimated values assuming nominal system behavior. These predictions for each operational variable are compared directly with their measured values using DTW technique. An anomaly flag is raised when the DTW distance-guided warning counter crosses a dynamically generated threshold. Due to the one-to-one nature of the independent mappings from battery metrics to each operational variable, the anomaly is also simultaneously isolated to the operational variable. The framework also includes automated testing of the trained ML models and anomaly detection parameters selected by artificially injecting different types of anomalies. The injected anomalies relate to loose connections, abrupt sensor failure, sensor drift, data corruption, and others. In this work, the implementation of this framework on datasets generated from laboratory tests on a small satellite platform is discussed.

2 Dataset

The dataset used to demonstrate this framework is provided by Arogtec Inc. Argotec has experience in small satellite hardware development, testing, and flight heritage including the recently successful ArgoMoon and LiciaCube missions. The data that is being utilized to demonstrate the proposed framework was generated in Argotec's test facilities, and the data and the setup utilized are described below.

2.1 Setup

The setup is composed of a satellite simulator and a ground station system. The satellite simulator involves two kinds of components: a satellite built with ground model subsystems and EGSE (Electronic Ground Support Equipment) simulating the environment or substituting missing subsystems to achieve a nominal behavior of the rest of the satellite.

The following is a brief description of the subsystems involved in the system architecture:

Battery: A device composed of several cells provides energy to the satellite when the solar panels are not pointing at the sun.

Solar Panel Array (SPA): Solar panel provides power to the satellite. SPA is simulated with a power supply

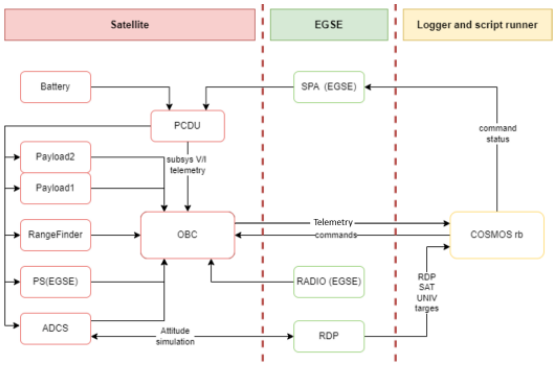


Fig. 1. System architecture to simulate satellite operation

and does not exactly match the current and voltage dynamics observed in space.

Power Converter and Distribution Unit (PCDU): The PCDU distributes power to different satellite subsystems at different required voltage levels, protects the circuits against shorts, and provides telemetry to the On-Board Computer (OBC).

Payloads 1,2: Scientific equipment (cameras) installed on the satellite.

Range Finder: Device to measure the distance to nearby objects in space.

Propulsion System (PS): A propulsion system device is simulated and its behavior is not distinguishable from the original flight model except for the gas pressure contained in the tank (a static value will be provided despite the different temperature conditions).

Attitude Determination and Control System (ADCS): ADCS is the device responsible for satellite orientation. This is similar to the flight hardware but does not have mounted wheels responsible for inertia compensation.

Realtime Dynamics Processor (RDP): EGSE dedicated to the environment simulation and sensor signal simulator are directly connected to the ADCS.

Radio: The radio device is simulated using an embedded board. Power consumption is not simulated by the system setup. This subsystem does not have a wide signal dynamic, and it is advised to consider only data recorded during the long-run sessions of the satellite.

COSMOS rb: Telemetry and Command manager. To achieve a wide range of signal dynamics, the test runner functionality will be used with a set of scripts sending commands to the satellite.

A couple of faults are injected during the tests, which include having an open circuit or a loose connection on a battery temperature sensor as shown in Fig. 2. However, a large number of anomalies are required for reliable results. Hence anomalies are injected artificially into the dataset as described in the following sections.

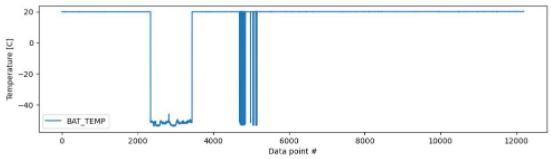


Fig. 2. Battery temperature sensor faults injected during testing

3 Currently Employed Fault Detection Methods

Understanding the current fault detection, isolation, and recovery (FDIR) practices in satellites is key to developing this framework. Standard FDIR practices followed in small satellites are described here. Generally small satellites integrate a software module dedicated to failure identification and recovery tasks. This functionality is implemented by monitoring telemetry data from sensors connected to the On-Board Computer (OBC) (or shared by the subsystems connected), comparing sensor values to predefined thresholds, and finally executing a recovery action.

FDIR is implemented in different subsystems and depending on the platform and the mission, and different the corresponding operational variables are monitored. Sample FDIR modules implemented on such platforms include FDIR table (monitor voltages, currents, and temperatures at different subsystems), FDIR ADCS (monitors the functions of Attitude Determination and Control System), FDIR EPS (checks power distribution unit functionalities and evaluation of related failures), and others. The proposed framework is expected to be particularly suitable for the evaluation of dynamic and analog operational variables in these tables. The FDIR table module, for example, evaluates the analog variables collected from sensors directly connected to the onboard computer and variables retrieved from reading subsystem status telemetries that are published. The analog measures are then compared with the limits predetermined according to the variable, component, subsystem, and expected characteristics.

Fault Detection:

Fault detection is achieved by comparing and classifying the analog values measured using a set of thresholds into five delineated regions described below and illustrated in Fig. 3.

- MIN ERROR: sensor value unexpectedly too low;
- MIN WARNING: sensor value between the minimum error state and the nominal state;
- NOMINAL STATE: sensor value as expected (between minimum and maximum warnings);
- MAX WARNING: sensor value between nominal value and a value to high;
- MAX ERROR: sensor value unexpectedly too high.

Error and warning thresholds, shown in Table 1 pertinent to subsystems are determined by the manufacturers

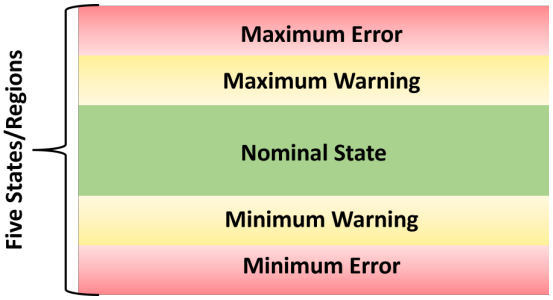


Fig. 3. Possible states for a measured analog variable determined by set thresholds

name	min err	min warn	max warn	max err
spa_temp	-20°C	-10°C	80°C	90°C
obc_temp	-15°C	0°C	40°C	55°C
...	...	...	...	...
psh_itm	0A	0A	2.2A	3.1A

TABLE 1: Example FDIR table showing threshold limits for sample variables

and accepted after extensive testing in different operational conditions. The operational limits (*MIN ERROR*, *MAX ERROR*) are determined in these tests. Exceeding those limits does not imply immediate failure of the component however, their functionalities are not guaranteed. Warning values (*MIN WARNING* and *MAX WARNING*) are then determined based on the variable dynamics and are chosen to be between 10 and 20% over or below the maximum or the minimum nominal values, respectively. Readings in a warning region do not compromise a component's functionality but raise a flag.

4 Methodology

The framework proposed here covers the process from data ingestion and abstraction of a system, to digital twin construction, feature extraction, model training, and anomaly detection methods. Each of these steps are discussed below.

4.1 Abstraction

The first step in this framework is to automate data ingestion from any system/platform and generate an abstracted digital twin using inter-parameter correlation metrics. Correlations thus extracted between any two system operational variables will indicate how well one of them could be used to diagnose the other. This step also provides a visualization of the abstracted system as a layered correlation graph connecting inter-dependant operational variables, in order to support fault explanation and root cause identification analysis. This task also includes identifying and selecting one or more ground truth variable(s) based on system-wide correlation scores and extracting suitable features from them to train a series of machine learning regression models used in the anomaly detection process.

Extracting Correlation Strengths

To accurately capture all system variable relationships for constructing a digital twin, we utilize Dynamic Time Warping(DTW) technique to evaluate a distance metric between all variables under a series of conditions. DTW is selected because a certain degree of latency between physically dependent variables is expected in many systems, and DTW accounts for this latency while still accurately quantifying the correlation between variables.

To construct a correlation mapping of the system, each of the n operational variables has a DTW distance score calculated between all of it's $n-1$ variables. These scores are then min-max normalized to be in the range [-1.0, 1.0]. Certain parameters, like voltage in a battery-powered system, may have their correlations to other variables obscured by a constant increase or decrease in their values over time. To correct this, DTW correlation scores are calculated for not only the original values, but also linear and polynomial detrended versions as well. Additionally, we also check for inverse relationships that may exist, such as between current and voltage, between all system variables. For each variable pairing only the highest of these correlation metrics is kept. An example mapping built using this technique describing all correlations within a small satellite is shown in Figure 9.

Feature Extraction Our framework deploys $n-1$ machine learning models to predict all operational variables using a selected ground truth measurement as input. The ground truth is selected based on it's correlation to all other variables in the system. In the case of the satellite, battery current information is selected as our ground truth variable, and prediction of the remaining variables is performed based on these ground truth readings.

Each learning model deployed corresponds to one operational variable, making run time efficiency a consideration for real-time prediction. For this we utilize a sliding time window that extracts a feature vector for each reading of the ground truth parameter. That is, each new reading of the ground truth results in a feature vector, $F = \{f_1, ..., f_9\}$, characterizing the information collected over the last $T_{now} - T_w$ readings. Each feature vector contains ground truth information for the most recent reading, maximum and minimum amplitudes, amplitude locations within the window, most recent peak and troughs, window average, impulse, and slope. Each feature vector, F_i, corresponds to a target value G_i^m, and model M_i takes F_i as input to predict g_i^m.

$$F = \{f_1, ..., f_9, \} \quad \text{and} \quad G = \{g_1, ..., g_n\}. \tag{1}$$

For a given target variable g_m, model M_m gives a single prediction $\hat{G}_m^i$ from feature vector F_m^i. During normal operation, the predicted $\hat{G}_m^i$ should match its observed reading collected from the system, i.e., G_m^i. To check for anomalies, empirical readings $G_n = \{g_n{}_i^j\}$ are

compared to the model estimated values $\hat{G}_n = \{\hat{g_n}_i^j\}$. In this unsupervised approach, our models are trained only on non-anomalous operational data, and do not rely on any pre-generated database of known fault signatures or behaviors. Because of this, the types of anomalies detectable by our framework are not restricted to known anomalous patterns or known thresholds, but our framework is instead capable of detecting a large range of unknown anomalous behaviors that are within acceptable ranges but may deviate from predicted norms.

4.2 Anomaly Detection

The second framework component is responsible for detecting anomalies observed in operational variables. This involves training of machine learning regression models to perform a one-to-one mapping using extracted ground truth features to predict other system operational variables. The ML models predict expected nominal behavior, which is then compared to actual observations. Anomalies are then labeled based on deviations between measured and predicted values. The optimization of parameters that define when and what type of differences are classified as anomalies, in addition to the quantification of the magnitude and duration of detected anomalies, is also explored here. To demonstrate these capabilities, different types of anomalies/fault models are developed and injected in the available datasets.

For preliminary testing, we demonstrate the model training, prediction, and anomaly detection processes using a small dataset collected from a testbed platform. This motorized propeller system is mounted to a test bench, and collects metrics such as current, RPM, torque, and thrust. In this scenario, the model uses battery current features to be used as model inputs and the node's raw values (thrust, torque, etc.) as targets.

Training of Machine Learning Prediction Models

After testing different ML models, Gradient Boosting [13] models were identified to have the most suitable all-around performance in terms of both runtime efficiency and prediction accuracy. Our framework automates the training and handling of the n models to be used, beginning with generating the correlation mapping, selecting a ground truth variable, extracting feature vectors, and training models on nominal behavior.

We have utilized the testbed data to demonstrate the automated pipeline to generate ML models. Figure 4 shows the predicted and observed raw values with the cumulative error for the testbed's thrust values. It is important to note here that we are using non-normalized features such as window average, peak and trough values of the ground truth node to predict the raw values of the targets.

Anomaly Models

For testing, in order to evaluate detection methods a variety of anomalies added to operational variables to mimic real-world faults. Anomaly types are classified as point, collective, and contextual [14] and are injected individually into different operational variables. These anomalies reflect situations where 1. only one sensor picks up abnormal behavior experienced by a system, 2. the measured/transmitted sensor data differs from true behavior, or 3. when components do not respond to control inputs due to hardware or software failure. In total, there are seven anomaly types injected, all with randomized injection points, anomaly durations, and intensities.

Noise: An array of randomly selected percentage modifiers, in the range of [-70%, 70%], is applied to a window of data to simulate noise added to the signal. This is shown in Fig. 5a.

Constant: Sets the measured data to a fixed value for the anomalous window. The selected value may be either the average value of non-anomalous data, a random constant taken from the valid data range, or set to 0. An example of one such anomaly is shown in Fig. 5b.

Value shift: A coefficient modifier applied to the original data trace. This increases or decreases the data by 20%-80% while maintaining the shape of its original behavior. A decrease shift is shown in Fig. 5c.

Point anomalies: A single anomalous point surrounded by normal data. The value of the anomalous point injected is a random value between 150% and 300% of the parameter's average value over a previous window. See Fig.5d. This could be representative of a loose sensor connection.

Drift anomalies: A gradually increasing coefficient modifier applied to the original data, shown in Fig. 5e beginning at 0% and eventually increasing the data by up to 80% its original value. This represents various anomalies that have a gradual onset.

Detection Methodology

If a trained model gives a perfect prediction of its target variable, then the residual between the two will be zero, which is not realistic. However, it can be assumed that the error rate between the model prediction and measured value will be fairly consistent if averaged over a period of time. Because our detection strategies revolve around discrepancies between measured values and model predictions we must quantify this deviation, however, certain system variable pairs have an inherent latency between their relationship, so methods based on standard error measures would result in high false positive counts. Instead, we base our detection on dynamic time warping measures collected over non-anomalous periods of time, effectively using this DTW distance score as an error metric, and then examine where subsequent periods fall along a distribution.

DTW-Distribution Detection

To quantify how the target predictions, $\hat{G}$, deviate from empirical readings, G, we first use a tumbling time window to compute a dynamic time warping distance score between the model prediction and measured reading. The length of the tumbling time window used is an adjustable parameter in our framework. By examining longer windows between model predictions and

measured values, the effects of short deviations are dampened, which can reduce false positives from short periods of poor model predictions. A shorter window size can capture specific anomaly types such as point anomalies more effectively, but may fail to capture long duration anomalies that are gradually onset. For final deployment, a combination of both short and long duration windows is suggested to capture a wider ranger of fault types. In addition to the window size, the radius considered in the DTW calculation may also be configured. Setting the radius parameter to 1 will effectively compute the Euclidean distance between the two-time series, while higher values allow for greater time warping.

Using the distance scores collected from this tumbling time window, our detection strategy of identifying outlying behavior is centered around the distribution curve of non-anomalous DTW scores. As shown in Fig. 6, when comparing the distributions of DTW scores collected during nominal and anomalous data, the distribution curve for data containing anomalies has a significantly longer right tail. Based on this observation, it can be assumed that if the DTW scores collected over a period of time fall outside of the non-anomalous distribution, then it is likely this period contains an anomaly.

To accurately quantify where a window score lies along the distribution, DTW scores are collected during a known non-anomalous period and over 100 different distributions are fitted to the data. From this we are able to determine a priority list of distributions that most accurately captures the distribution curve across all variables based on the sum of their squared error, as shown in Fig. 7.

Detection under this method first begins by fitting this list of distributions to DTW distance scores collected over a known non-anomalous period of time to quantify nominal behavior. Then, for subsequent windows examined each distance score is measured against the full width at half the maximum peak value of the nominal distribution curve. If the DTW distance over a period of time exceeds a given distance from the distribution peak, in units of this full width at half maximum value(FWHM), it is considered too far from the nominal distribution and thus abnormal. I.e., a collected DTW score is considered to be an outlier of the nominal distribution when $abs(dtwscore - distributionpeak) > c * fwhm$. The coefficient c used for this threshold is a configurable parameter within the framework.

A warning counter is then increased based on the distance between the DTW score of this abnormal window and the nominal distribution peak, that is the warning counter is incremented by $(((dtwscore - distributionpeak)/threshold)/fwhm)$. This approach ensures that farther outliers are given a greater weight and are more quickly classified to be anomalous. Similarly, if subsequent periods are within the nominal distribution, this warning counter is decreased based on how close to they are to the nominal distribution

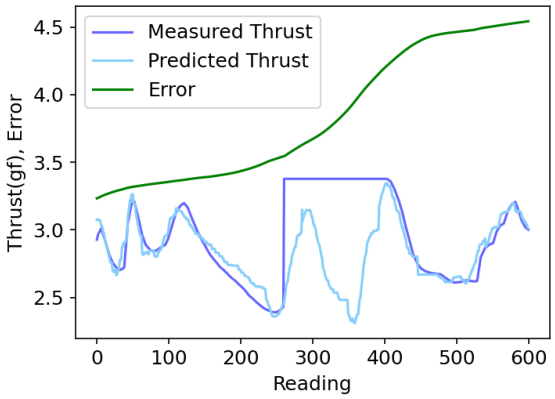
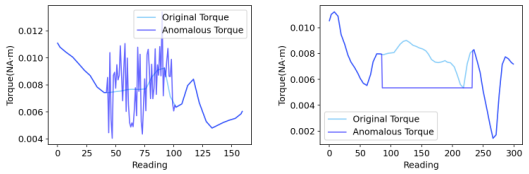
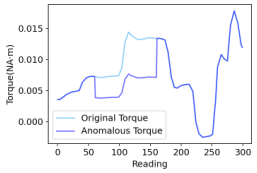


Fig. 4. Cumulative error between model predicted and measured thrust values of the testbed.

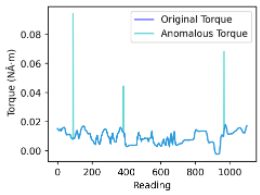


(a) Injected Noise

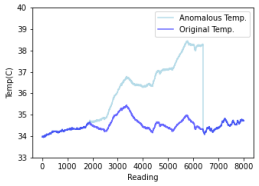
(b) Constant value anomaly



(c) Shift anomaly



(d) Point anomaly



(e) Drift anomaly

Fig. 5. Example anomaly types injected into true data.

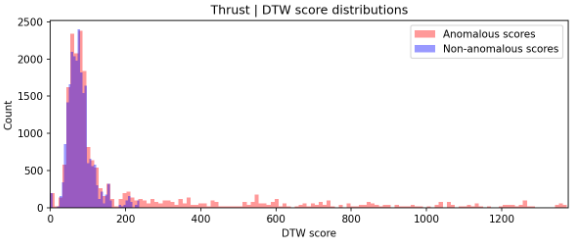


Fig. 6. Distributions of DTW window scores for anomalous and non-anomalous data.

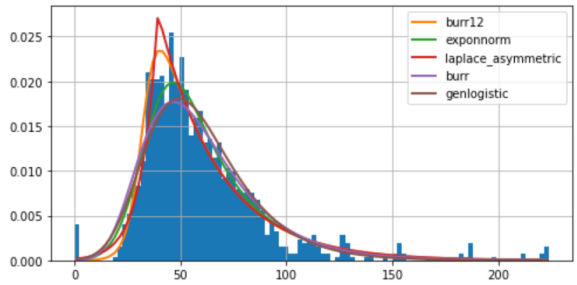


Fig. 7. Most accurately fitted distributions to DTW scores in Torque.

peak, $warnings = warnings - ((windowsize * threshold * 2) + (0.3 * (FWHM/(dtwscore - distributionpeak)))$. This gives a decay to the warning counter and prevents it from gradually accumulating and flagging false positives. Anomalies are then flagged when this warning counter exceeds a warning threshold, set as a configurable percentage of the window size being considered. This method also dynamically adapts itself to changes in system behavior over time, such as slow gradual voltage drops in battery-based systems, by refitting distributions and updating nominal thresholds during extended periods of non-anomalous behavior.

The configuration of the parameters for this detection method determine the types of anomalies that may be detected, detection rate, number of false positives, and detection latency. An extensive grid search has yielded well-performing default values for the configuration, and results are discussed in following section. Fig. 8 demonstrates the behavior of this detection method for three separate anomalies(constant value, value shift, and noise anomalies) injected into the testbed data, plotting the DTW score of each window and the warning counter behavior during the deviation between predicted and measured values.

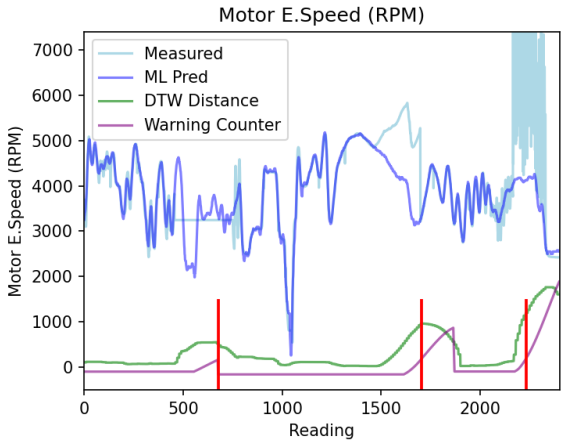


Fig. 8. Detection behavior for three anomalies. Anomaly detection time is marked by vertical red lines.

5 Results

In this section, we discuss the results generated by deploying the framework and algorithms discussed in the previous section on satellite data.

5.1 Abstraction Results

After ingesting the satellite datasets and identifying the analog operational variables in the satellite dataset, heat maps of DTW correlation strengths are generated. Figure 9 shows the correlation scores between all operational variable pairs of the small satellite. The subsystem or category of the operational variables is labeled in the figure as VTM/ITM (voltage and currents), OBC (onboard computer), ADCS (attitude determination and control system), PS (power system), and a Radio subsystem. These scores have been generated after implementing de-trending, inverse-checking, and data normalization methods. Several dark blue blocks indicate strong correlations between different operational variables. In Fig. 10, these scores are arranged in the descending order of their correlation strengths with respect to battery current. A bar chart at the top of the figure shows these values. Using battery current as the ground truth parameter, anomaly detection results are expected to be superior for those with higher correlation strengths.

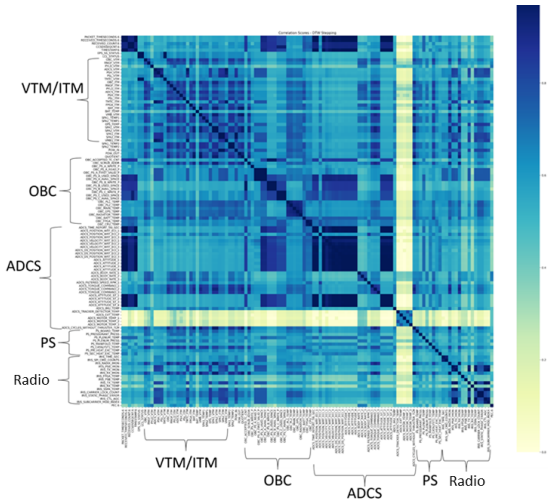


Fig. 9. Heatmap showing correlation scores using DTW technique

Figure 11 shows the top 20 operational variables that are the most correlated with the battery current variable. These include several temperature variables such as SPA temperatures, SPA voltages, power measurements, and ADCS torque commands.

5.2 ML Prediction Results

The operational variables with strong correlations to battery current are selected for training ML models.

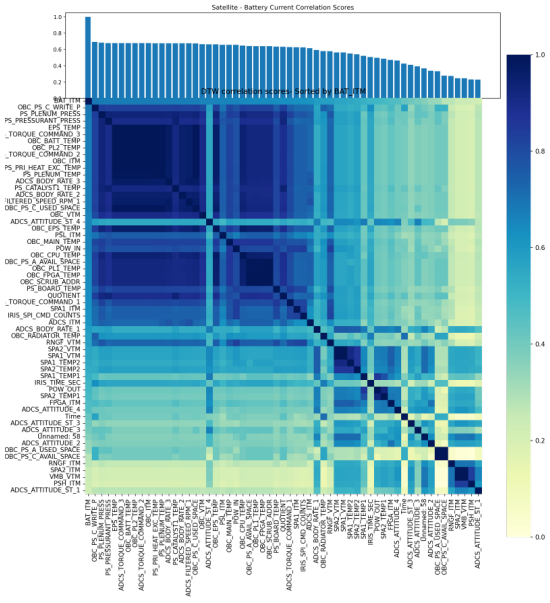


Fig. 10. Heatmap showing DTW correlation scores and operational variables arranged according to descending order of correlation strengths with battery current

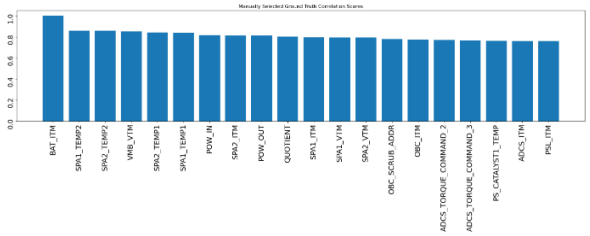


Fig. 11. Top 20 operational variables correlated to battery current in satellite data

Features are then extracted from the battery current readings, and separate Gradient Boosting ML models are trained for each operational variable to be monitored. ML predictions for two sample operational variables (SPA temperature and power output) are shown in Figures 12 and 13. The prediction accuracy is highly dependent on the correlation strength and the amount of training data available which covers the relationship between the ground truth and the selected variable.

Figure 14 shows the Mean Average Percentage Error (MAPE) values of 60 ML models for different operational variables selected.

5.3 Anomaly Detection Results

After the ML models are trained, anomalies are randomly injected into the different operational variables. DTW distribution-based methodology was applied to detect the injected anomalies with a nominal amount

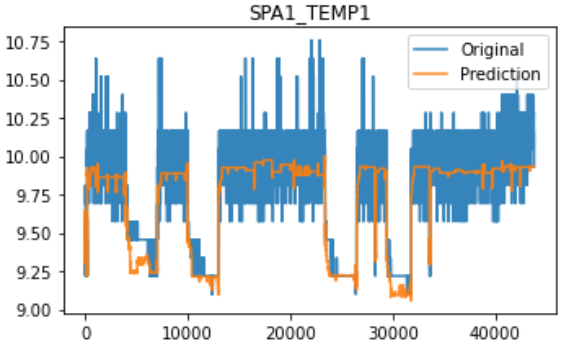


Fig. 12. ML prediction of satellite SPA temperature variable compared to observed behavior

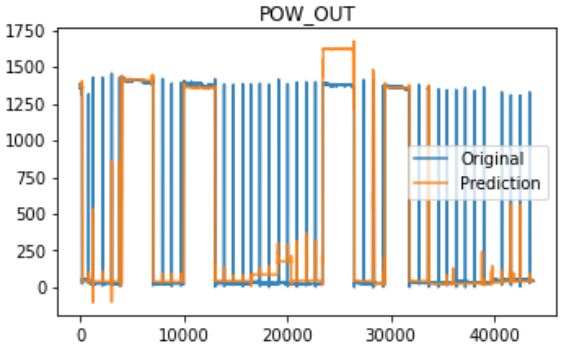


Fig. 13. ML prediction of satellite power output variable compared to observed behavior

of tuning of the detection parameters. The true positive and false positive detection rates on all 60 operational variables selected for monitoring are shown in Fig. 15. The results show that the operational variables that had the largest MAPE values, such as ADCS body rate and ADCS torque commands, also showed poor detection performances. In addition, a few variables with low MAPE values also showed poor detection performance. Further analysis with the help of a subject matter expert may be required to understand the underlying reasons.

To downselect the operational variables for monitoring, a detection rate threshold could be utilized. Optimally selecting the variables with low false positive rates and high detection rates would be beneficial for satellite health monitoring operations. This will reduce the burden on the operators to unnecessarily analyze false positive detections while capturing most anomalous behaviors. Figure 16 shows the operational variables with a detection rate of greater than 60%. In this case, the average detection rate is found to be 81.8% with an average false positive rate of only 2.0%.

In Fig. 17, only the operational variables with anomaly detection rates greater than 80% are shown. In this case, the average anomaly detection rate is 89.1% and the

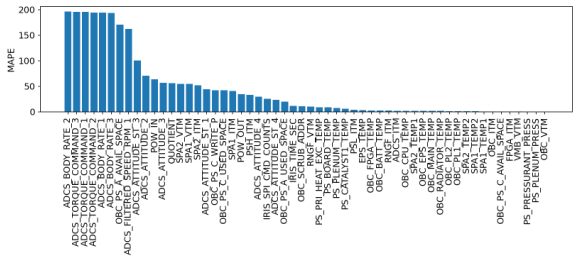


Fig. 14. Mean average percentage errors of ML models predicting different satellite operational variables

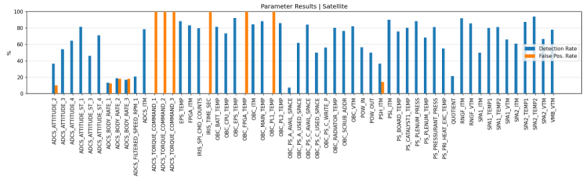


Fig. 15. Anomaly detection results on all selected satellite operational variables.

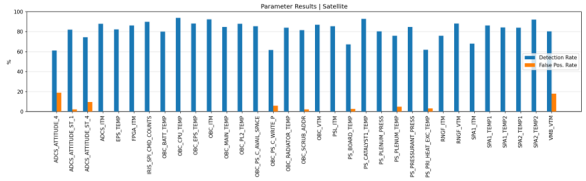


Fig. 16. Anomaly detection results on satellite operational variables with greater than 60% detection rate.

average false positive rate is found to be only 0.9% for the number of anomalies injected and passed through detection.

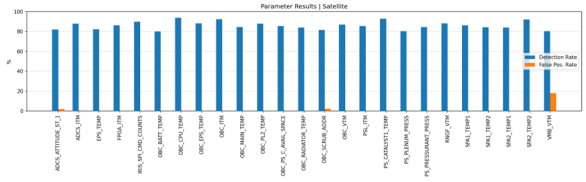


Fig. 17. Anomaly detection results on satellite operational variables with greater than 80% detection rate.

Figure 18 shows the number of operational variables that would be monitored for different detection rate cutoffs. The figure shows detection rate cutoffs from 50% to 100% and the number of variables changes from 38 variables at 50% to 25 variables at an 80% cutoff.

6 Conclusion

In this paper, a reusable and fully data-driven framework for anomaly detection in small satellites is proposed. By leveraging correlations between operational variables, feature extraction processes, and machine

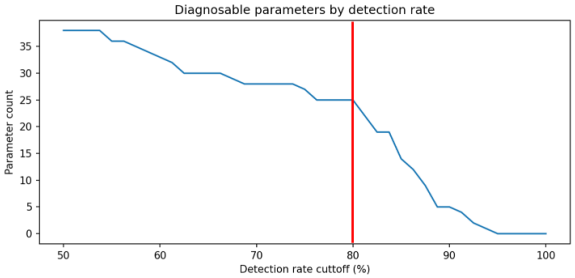


Fig. 18. Number of satellite operational variables that can be monitored for different detection rate cutoff scenarios

learning techniques, this framework enables the rapid detection and isolation of anomalous behaviors. The framework's platform-agnostic nature allows for cost-efficient implementation on most small satellite platforms. It can be deployed either onboard a vehicle to support autonomy or at a ground station to improve operational efficiency and mission success rates.

Through implementation and testing on satellite test datasets, we have demonstrated the effectiveness of our framework. Correlation scores were generated using Dynamic Time Warping (DTW) technique which revealed strong relationships between battery current and various operational variables. ML models trained to learn these relationships achieved low prediction errors for different monitored variables and contributed to reliable detection of injected anomalies. The anomaly detection results showed average detection rates of nearly 90% and an average false positive rate of less than 1% for 25 operational variables which showed strong correlation with battery current. Future work includes further refining the framework, expanding the datasets for evaluation, and collaborating with industry partners to deploy and validate the framework in operational small satellite missions.

References

[1] J. R. Kopacz, R. Herschitz, and J. Roney, "Small satellites an overview and assessment," *Acta Astronautica*, vol. 170, pp. 93–105, 2020.

[2] D. M. Harland and R. Lorenz, *Space systems failures: disasters and rescues of satellites, rocket and space probes*. Springer Science & Business Media, 2007.

[3] F. SalarKaleji and A. Dayyani, "A survey on fault detection, isolation and recovery (fdir) module in satellite onboard software," in *2013 6th International Conference on Recent Advances in Space Technologies (RAST)*. IEEE, 2013, pp. 545–548.

[4] K. Djebko, F. Puppe, and H. Kayal, "Model-based fault detection and diagnosis for spacecraft with an application for the sonate triple cube nano-satellite," *Aerospace*, vol. 6, no. 10, p. 105, 2019.

[5] L. M. Fesq, "Current fault management trends in nasa's planetary spacecraft," in *2009 IEEE Aerospace conference*. IEEE, 2009, pp. 1–9.

[6] S. A. Jacklin, "Small-satellite mission failure rates," Tech. Rep., 2019.

[7] J. Melville, J. B. Harley, M. Lopez, M. Crabtree, and S. Lacy, "Methods for data-centric small satellite anomaly detection and fault prediction," 2022.

[8] K. Hundman, V. Constantinou, C. Laporte, I. Colwell, and T. Soderstrom, "Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding," in *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, 2018, pp. 387–395.

[9] D. Iverson, R. Martin, M. Schwabacher, L. Spirkovska, W. Taylor, R. Mackey, and J. Castle, "General purpose data-driven system monitoring for space operations, 2009 aiaa infotech@ aerospace conference," *Seattle, WA, Apr*, 2009.

[10] S. Fuertes, G. Picart, J.-Y. Tourneret, L. Chaari, A. Ferrari, and C. Richard, "Improving spacecraft health monitoring with automatic anomaly detection techniques," in *14th international conference on space operations*, 2016, p. 2430.

[11] S. Ramachandran, M. Rosengarten, and C. Belardi, "Semi-supervised machine learning for spacecraft anomaly detection & diagnosis," in *2020 IEEE Aerospace Conference*. IEEE, 2020, pp. 1–10.

[12] J. Pace, J. P. Rao, J. Williams, and L. He, "Unsupervised anomaly detection using batteries in electric aerial vehicle propulsion testbed," in *Annual Conference of the PHM Society*, vol. 14, no. 1, 2022.

[13] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "Lightgbm: A highly efficient gradient boosting decision tree," *Advances in neural information processing systems*, vol. 30, pp. 3146–3154, 2017.

[14] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Comput. Surv.*, vol. 41, 07 2009.