International Journal of Reliability, Risk and Safety: Theory and Application

Online ISSN: 2676-3346

OPEN ACCESS

Vol. 9/ Issue 1/ 2026/ pp. 56-83

DOI: 10.22034/IJRRS.2026.9.1.5

Received: 06 August 2025, Revised: 13 June 2026, Accepted: 17 June 2026

Available online at: https://www.ijrrs.ir

Original Research Article

Applications of Artificial Intelligence in Systems Engineering for Satellite Fault Prediction

Mostafa Mohammad Ali Nezhad * **and Mahdi Nasiri Sarvi**

School of Advanced Technologies, Iran University of Science and Technology, Tehran, Iran

*** Mustafa.man@live.com**

Abstract

Keywords: *Artificial Intelligence* – Spacecraft fault prediction is difficult because flight telemetry is high-dimensional, non-stationary, and only sparsely labeled. This paper examines how artificial intelligence can be integrated into satellite systems engineering to support early warning and fault diagnosis without replacing deterministic fault detection, isolation, and recovery (FDIR) logic. The study compares Artificial Neural Networks (ANN), Support Vector Machines (SVM), Recurrent Neural Networks/Long Short-Term Memory (RNN/LSTM), and hybrid ANN-EKF models for representative AOCS, EPS, thermal, communication, and onboard-data-handling fault cases.

Because operational fault labels are limited, the revised methodology distinguishes between public spacecraft anomaly benchmarks such as SMAP/MSL, NASA Prognostics Center of Excellence degradation datasets, mission/event logs, and mission-representative fault-injection simulations. The dataset section now specifies source categories, labeling methods, telemetry variables, and sim-to-real differences. Performance is assessed using accuracy, precision, recall, F1-score, AUC/ROC, RMSE/MAE, execution time, false-alarm behavior, and interpretability through SHAP and LIME.

The results indicate that temporal models such as LSTM are most suitable for gradual degradation and drift-type anomalies, while SVM/PCA and rule-assisted classifiers remain useful when labeled data are small. Hybrid physics-informed AI is appropriate when subsystem dynamics are available. The paper further defines onboard feasibility conditions: training and validation are assumed to be ground-based, while onboard use is limited to lightweight inference subject to latency, memory, processor, power, and flight-software constraints. The contribution is therefore an AI-assisted monitoring architecture for satellite fault prediction, not a claim of fully autonomous flight qualification.

Systems Engineering – Satellite Failure Prediction – Machine Learning – Deep Learning – Fault Detection and Isolation.

Nomenclature

AI	Artificial Intelligence
ML	Machine Learning
ANN	Artificial Neural Network
SVM	Support Vector Machine
RNN	Recurrent Neural Network
LSTM	Long Short-Term Memory
CNN	Convolutional Neural Network
GAN	Generative Adversarial Network
EKF	Extended Kalman Filter
UKF	Unscented Kalman Filter
PF	Particle Filter
HMM	Hidden Markov Model
PCA	Principal Component Analysis
RFE	Recursive Feature Elimination
SHAP	Shapley Additive explanations
XAI	Explainable Artificial Intelligence
AOCS	Attitude and Orbit Control System
ADCS	Attitude Determination and Control System
EPS	Electrical Power Subsystem
TCS	Thermal Control Subsystem
COMMS	Communications Subsystem
CDHS	Command and Data Handling Subsystem
OBC	On-Board Computer
LEO	Low Earth Orbit
MEO	Medium Earth Orbit
GEO	Geostationary Earth Orbit
RTOS	Real-Time Operating System
F1 Score	F1 Statistical Score
AUC	Area Under Curve
MAE	Mean Absolute Error
RMSE	Root Mean Square Error
IoT	Internet of Things
MATLAB	Matrix Laboratory (software environment)
SMC	Sliding Mode Control
LQR	Linear Quadratic Regulator
GPU	Graphics Processing Unit
TPR / TNR	True Positive/True Negative Rate
ROC	Receiver Operating Characteristic Curve

How to cite this article:

M. Mohammad Ali Nezhad and M. Nasiri Sarvi, "Applications of artificial intelligence in systems engineering for satellite fault prediction," *International Journal of Reliability, Risk and Safety: Theory and Application*, vol. 9, no. 1, pp. 56-83, 2026, 10.22034/IJRRS.2026.9.1.5.

1. Introduction

Satellite missions now depend on tightly coupled electrical, thermal, attitude-control, communication, and onboard-computing subsystems. A small abnormality in one part of the spacecraft can propagate through the system: a power regulator oscillation may disturb the onboard computer, a thermal bias may change sensor behavior, and a reaction-wheel current increase may precede pointing degradation. For this reason, fault prediction must be treated as a systems-engineering problem rather than as an isolated classification task.

Conventional monitoring remains essential because threshold logic and deterministic FDIR rules are transparent, testable, and suitable for immediate protective actions. Their weakness is not that they are obsolete, but that they are limited when telemetry is nonlinear, coupled, and evolving over long time intervals. Data-driven models can add value by detecting weak temporal patterns before a hard limit is crossed, provided that their outputs are interpreted as advisory evidence and validated against mission rules [1].

This paper evaluates ANN, SVM, RNN/LSTM, and ANN-EKF models as candidate tools for satellite fault prediction. The emphasis is on how each model fits a specific telemetry pattern: static nonlinear relationships, small labeled datasets, long-term drift, or residuals from a physical estimator. SHAP and LIME are included to show which telemetry variables influence the prediction, because a model that cannot be explained is difficult to use in a mission operations environment.

1.1 Objectives of This Research

The research objectives are to define an AI-assisted fault-prediction workflow for satellite subsystems, compare model families under representative operating disturbances, clarify the dataset assumptions behind the reported results, define explicit fault classes and severity levels, and discuss whether the models can realistically be deployed onboard or should remain ground-segment decision-support tools.

1.2 Background: Why Satellite Reliability Demands Predictive AI

The reliability of spacecraft directly determines the scientific return, mission lifetime, and cost efficiency of an orbital program. For complex platforms—Earth observation satellites, navigation spacecraft, or multi-payload CubeSats—the slightest failure in the AOCS, EPS, onboard computer (OBC), thermal subsystem, or TT&C segment can render the mission partially or completely inoperable.

Because satellites cannot be repaired in orbit (except in rare human-serviced missions), spacecraft must rely on robust reliability engineering principles: redundancy, graceful degradation, fault tolerance, and high-fidelity monitoring. Traditional approaches—such as fixed alarm thresholds for battery temperatures, bus currents, or attitude errors—lack sensitivity to gradual degradation. For example, early signs of torque imbalance in reaction wheels or a drifting star-tracker bias often remain hidden in telemetry long before the anomaly crosses any predefined threshold [2].

The increasing complexity of modern spacecraft, along with miniaturization and power constraints, has shifted the industry toward predictive, data-driven fault management. AI techniques are capable of uncovering nonlinear relationships among telemetry variables (e.g., thermal–electrical coupling in EPS or time-delayed correlations in wheel friction). This study follows this direction by focusing on AI integration within the broader systems engineering lifecycle.

1.3 Why Traditional Fault Detection No Longer Suffices

Traditional fault detection frameworks served well in early missions, but five major shortcomings now limit their usefulness:

- Rule Exhaustion – Handcrafted rules cannot encode the full range of real-world failure signatures, especially anomalies that emerge from combined subsystem interactions (e.g., EPS degradation influencing AOCS pointing stability).
- Weak Nonlinearity Modeling – Linear trend methods fail when telemetry behaves differently under varying thermal states, power modes, or orbital environments.
- Noise Vulnerability – Space telemetry is inherently noisy due to radiation, sensor aging, and communication dropouts. Classic methods often raise false alarms or miss hidden drifts.
- No Temporal Memory – Many failures evolve gradually (reaction wheel friction growth, battery internal resistance increase). Legacy systems cannot model these slow temporal dynamics.
- Poor Scalability – Modern satellites may log hundreds of telemetry channels at 1–100 Hz. Static rules cannot scale to this complexity.

These limitations collectively justify the transition toward adaptive, learning-based fault prediction systems [3].

1.4 Emergence of AI in Aerospace Systems

1.4.1 Convergence of AI Needs and Spacecraft Realities

Satellite telemetry is inherently high-dimensional, noisy, and time-correlated. AI—particularly RNN/LSTM, SVM with nonlinear kernels, and ANN-based regressors—fills a critical gap where physics-based or rule-based models fall short.

1.4.2 Current AI Use-Cases in Space Missions

Real examples include:

- CNN-based geophysical feature extraction from Mars Reconnaissance Orbiter imagery [4];
- ESA's use of anomaly detection algorithms for solar array degradation forecasting;
- ML-based fault classification in Mars Express power subsystem telemetry;
- RL-based control tuning in autonomous small-satellite testbeds.

1.4.3 Role in Systems Engineering

AI augments classical SE practices by providing:

- real-time trade-off analysis,
- probabilistic risk assessment,
- predictive maintenance scheduling,
- adaptive reconfiguration strategies.

1.4.4 Enabling Technologies

Recent advances—Jetson-class processors, Coral Edge TPU, fault-tolerant AI accelerators—allow inference directly onboard, even under power constraints. XAI tools help system operators trust and verify model outputs.

1.5 Preventive Maintenance in LEO and MEO Satellites

1.5.1 Environmental Stressors

LEO: harsh thermal cycles, atomic oxygen, micro-debris impacts, and frequent eclipse transitions.

MEO: increased radiation dose, higher probability of SEUs, longer communication gaps.

1.5.2 Preventive Maintenance Philosophy

Preventive maintenance uses predictive indicators—battery impedance growth, reaction wheel current signature, thermal runaway precursors—to intervene before failure. ML models amplify this capability by spotting weak signals buried in telemetry. The advantages are summarized in Table 1.

1.5.3 Benefits of AI-Based Prediction

Table 1. Advantages of AI-Based Fault Prediction in Satellite Systems

Advantage	Impact
Extended Satellite Lifespan	Delays functional decline and extends service life to the full design envelope.
Enhanced Mission Assurance	Reduces risk of abrupt subsystem failure or data loss.
Cost Efficiency	Minimizes the need for premature replacement missions.
Real-Time Fault Management	Enables timely operator response through predictive alerts.

1.5.4 Algorithms in Use

- RNNs for evolving time-series degradation,
- Autoencoders for subtle drift identification,
- Hybrid AI + heuristic systems for constrained platforms.

1.6 Research Questions and Hypotheses

To explore the potential of AI-driven fault prediction in satellite systems, this study formulates critical research questions and hypotheses aligned with scientific rigor and operational relevance.

1.6.1 Research Questions

Can AI models predict satellite system failures in real-time for LEO and MEO missions?

Which ML algorithms, SVM, RNN, or ANN, demonstrate superior performance in fault classification and temporal prediction?

What telemetry parameters (e.g., temperature, current, angular velocity) contribute most to predictive accuracy?

Can the proposed AI framework estimate the time of fault occurrence more precisely than conventional methods?

To what extent can AI-based fault prediction reduce operational risk and enhance system reliability?

1.7 Innovations and Contributions

The contribution of this paper is a structured AI-assisted workflow that connects telemetry preprocessing, model selection, fault classification, explainability, and deployment constraints. The emphasis is not on presenting AI as a substitute for spacecraft FDIR, but on defining where data-driven prediction can provide earlier evidence for mission operators and subsystem engineers [5-6].

1.7.1 Unified AI Fault Prediction Framework

The proposed workflow follows the sequence of telemetry acquisition, preprocessing, feature construction, model training, validation, interpretability, and deployment assessment. It is designed to accept public benchmark data, mission/event logs, and controlled fault-injection simulations, while clearly separating evidence derived from real telemetry from evidence derived from simulation [7].

1.7.2 Comparative Performance of ML/DL Models

A detailed benchmarking of ANN, SVM, RNN, and ANN-EKF models is conducted, providing insight into their respective strengths, convergence behavior, and application contexts in space environments.

1.7.3 Robustness via Sensitivity Analysis

The study includes a rigorous analysis of:

- Noise impact (Gaussian injection),
- Training data volume variations,
- Feature selection techniques (e.g., RFE),
- Demonstrating model stability under real-world non-idealities.

1.7.4 Real-Mission Data Utilization

In contrast to simulation-only approaches, this research incorporates telemetry from Mars Express and CubeSat missions, validating AI methods under real mission conditions.

1.7.5 Visual Analytics and Engineering Insight

The study includes over 20 visual artifacts, figures, tables, and charts, including:

- ROC curves,
- Confusion matrices,
- Failure rate trends, and
- AOCS failure costs, which improve interpretability for systems engineers and decision-makers.

1.7.6 Future-Oriented Technological Pathways

The work outlines next-generation AI frontiers:

- Reinforcement Learning for onboard autonomy [8-10],
- Edge AI using Jetson/Coral-class devices,
- Quantum Machine Learning for complex telemetry fusion

2. Focused Literature Review and Research Background

The last decade has seen a marked increase in the functional density of satellite platforms: higher payload integration, more aggressive duty cycles, and tighter power and thermal margins. As a direct consequence, the fault space of satellite subsystems has expanded, both in variety and in coupling across subsystems. Classical fault detection methods, which were sufficient for earlier, simpler missions, struggle with this landscape. Recent work (2018–2025) therefore moves toward AI-based methods that can exploit telemetry at scale and support predictive, rather than purely reactive, maintenance [11].

This section reviews:

- The main classes of satellite subsystem faults and their mapping to telemetry signatures;
- The AI models are most frequently used for fault prediction;
- The remaining research gaps that motivate the framework proposed in this paper.

2.1 Classification of Satellite Subsystem Faults

Modern satellites consist of tightly coupled subsystems—AOCS/ADCS, EPS, TT&C, thermal control, payload, propulsion, and onboard computing. Each subsystem exhibits its own characteristic failure modes, but in practice, these modes interact. A meaningful literature review therefore starts by classifying faults not only by subsystem but also by the type of observable telemetry pattern (slow drift, abrupt jump, intermittent glitch, etc.) [12].

2.1.1 Subsystem-Level Fault Taxonomy

The literature consistently distinguishes between sensor faults, actuator faults, software/logic faults, and structural or hardware degradation.

Table 2. Attitude Determination and Control Subsystem (ADCS)

Fault Type	Description	Consequences
Sensor Fault	Incorrect data from gyroscope, magnetometer, or sun sensor	Erroneous attitude estimation, targeting error
Actuator Fault	Malfunction or failure of reaction or magnetic wheels	Inability to control attitude, uncontrolled spinning
Software Fault	Algorithmic errors in estimation and control	Disturbance in dynamic response

Table 3. Electrical Power Subsystem (EPS)

Fault Type	Description	Consequences
Battery Voltage Drop	Capacity loss due to degradation or damage	Sudden system shutdowns
Solar Panel Failure	Thermal or mechanical damage	Reduced power input and mission lifespan
Power Distribution Unit Issue	Irregular power delivery	Subsystem instability

Table 4. Communications Subsystem

Fault Type	Description	Consequences
Loss of Uplink/Downlink	Antenna or RF circuit failure	Loss of contact with ground station
Modulation Error	Noise or incorrect parameters	Data quality loss, command delays
Communication Software Bug	Data control layer error	Synchronization issues

Table 5. Thermal Control Subsystem (TCS)

Fault Type	Description	Consequences
Temperature Sensor Failure	Incorrect temperature loop data	Overheating or freezing of components
Heater/Radiator Malfunction	Inadequate heat management	Scientific or power systems failure

Table 6. Payload Subsystem

Fault Type	Description	Consequences
Detector Malfunction	Sensor burnout or noise	Loss of scientific data
Payload Software Error	Crash or incorrect processing	Mission repetition or data loss

Table 7. Propulsion Subsystem (if applicable)

Fault Type	Description	Consequences
Leakage or Pressure Drop	In tank or nozzle	Maneuver failure
Valve/System Stuck	Mechanical or command failure	Orbit/attitude control issues

Tables 2-7 summarize the dominant fault types per subsystem, together with their immediate operational consequences. For example: In ADCS, sensor faults (e.g., gyroscope bias drift, star tracker misidentification) primarily manifest as attitude estimation errors, which then propagate into pointing performance and imaging quality.

In EPS, battery aging, regulator malfunction, or solar array degradation produces voltage sag, current anomalies, or increased temperature, which can cascade into protection trips or spontaneous reboots.

In TT&C, intermittent link loss, RF front-end faults, or protocol/software defects can decouple the satellite from ground control, turning even minor internal anomalies into mission-ending events [13-15].

The goal of this taxonomy is not to be exhaustive but to identify:

Which telemetry channels carry the earliest reliable signatures of each fault type,

Which AI model classes are most appropriate for those signatures?

2.1.2 Mapping Fault Types to AI Models

Table 8. Mapping Fault Types to AI Models

Fault Type	Subsystem	Data Features	Suggested AI Model	Rationale
Gyroscope Drift	ADCS	Temporal, nonlinear, memory-dependent	RNN / LSTM	Models sequential time-series data
Battery Capacity Loss	EPS	Voltage, current, temperature	ANN	Nonlinear pattern recognition
Solar Panel Degradation	EPS	Solar flux, output current, temperature	SVM + PCA	Accurate anomaly classification
Attitude Control Software Bug	ADCS	Control signals, outputs	Hybrid (ANN + Rule-Based)	Combines learning with expert logic
Thermal Anomalies	TCS	Multimode temperature data	CNN	Spatial data analysis
Propulsion Pressure Drop	Propulsion	Pressure, temperature, fuel flow	Decision Tree / Random Forest	Hierarchical interpretability
Telemetry Database Fault	Comms / Software	Structured text data	Autoencoder / GAN	Unsupervised anomaly detection

Gyroscope drift is a slowly evolving bias in a time series. RNN/LSTM models are preferred because they capture long-term temporal dependencies and can distinguish between transient noise and genuine drift. The model mapping is provided in Table 8.

Battery capacity loss is reflected in multi-parameter patterns (voltage under load, temperature, charge history). Fully connected ANNs work well here because the relationship between these variables and "effective capacity" is nonlinear but not fundamentally temporal.

Solar panel degradation often appears as a gradual reduction in output power under comparable illumination conditions. SVM combined with PCA is effective when datasets are moderate in size, and the objective is to separate "healthy vs. degraded" operating regimes in a compressed feature space.

Thermal anomalies frequently exhibit spatial correlation across sensors mounted on a panel or instrument. CNNs can exploit this spatial structure, especially when thermal maps or sensor layouts are modeled as 2D grids [16].

Telemetry database or software-layer faults tend to produce irregular, sparse, or corrupted patterns rather than coherent physical trends. Autoencoders, GANs, and other unsupervised models are more appropriate here because the notion of "fault" is defined as deviation from nominal data distributions rather than a predefined label.

2.1.3 ADCS (Attitude and Orbit Control System)

ADCS is a natural focus for AI-based fault prediction because it combines heterogeneous sensors, estimation algorithms, and actuators operating in a closed loop. The literature reports repeated occurrences of:

- Gyroscope drift and bias jump, often exacerbated by radiation damage and aging;
- Star tracker anomalies, including loss of lock, misidentification, or saturation;
- Reaction wheel faults, such as bearing friction growth, imbalance, or motor driver failures [17].

Table 9. Detailed Descriptions of Common ADCS Faults

Fault Type	Description
Gyroscope Drift	Accumulating bias or hardware failure
Star Tracker Error	Optical saturation or misidentification
Reaction Wheel Anomalies	Overspeed, motor failure, internal friction
Magnetometer Interference	Environmental or onboard noise
Software Misconfiguration	Improper filter tuning or sampling rate

Table 9 details common ADCS faults. A well-known historical case, the Mars Climate Orbiter (1999), illustrates how seemingly "soft" software errors in ADCS-related logic (unit mismatch in thrust calculation) can destroy an entire mission. AI-based ADCS monitoring in recent studies typically uses:

- RNN/LSTM models trained on gyro and star tracker data to detect subtle divergence from expected dynamics;
- Hybrid ANN+EKF schemes where the EKF enforces physical consistency and the ANN learns residual patterns associated with impending faults [18-20].

2.1.4 EPS (Electrical Power Subsystem)

The EPS literature emphasizes battery health prediction and solar array degradation as primary themes, especially for long-duration LEO missions and small satellites with little power margin. Commonly reported faults include:

- Capacity fade and internal resistance increase in batteries, with signatures in voltage-under-load curves and charge/discharge temperature profiles;
- Solar array damage due to radiation, micro-debris, or thermal cycling, visible as reduced power under comparable illumination;
- Regulator and distribution faults, manifested through oscillatory bus voltages or repeated protection trips.

Table 10. AI Model Applications for Power System Fault Detection

AI Model	Feature Detected	Fault Type
LSTM	Nominal capacity drop	Battery degradation
SVM	Voltage under constant load	Internal resistance increases
RNN	Rapid heat and current rise	Thermal runaway
Decision Tree	Charge/discharge fluctuation	BMS fault
Regression + Aging Model	Output power decay	Solar degradation
Autoencoder	Anomaly with stable radiation	Contamination
Random Forest	Discrete resistance jumps	Impact damage
CNN	Solar image-based detection	Tracking mechanism error
Fuzzy Logic	Output mismatch	Regulator fault
Hybrid ANN + Rule	Sudden shutdown	Distribution anomaly

Table 10 summarizes these power-system model applications:

- LSTMs for horizon-level prediction of state-of-charge and temperature;
- SVMs for discrete health-state classification;
- Autoencoders for discovering unusual power signatures that do not match trained "healthy" patterns.

A recurring theme in recent works is the trade-off between model complexity and onboard implement ability: powerful models (e.g., deep LSTMs) offer better predictive accuracy but may exceed the computational and power budget of typical OBCs, especially without dedicated accelerators [21].

2.1.5 Propulsion and Thermal Subsystems

The propulsion subsystem is heavily safety-critical. Faults such as valve leakage or thruster underperformance are rare but catastrophic. Table 11 summarizes the propulsion fault-detection choices.

The literature leans toward models that can work with scarce labels:

- RNNs on tank pressure and temperature time series to detect slow, persistent pressure decay;
- One-class SVMs and Isolation Forests are where only "normal" operation is well-characterized, and anomalies are defined as outliers.

For the thermal subsystem, recurrent models and autoencoders dominate:

LSTMs can forecast expected temperature trajectories under known operating modes and eclipse patterns; deviations from the predicted envelope indicate faults in heaters, radiators, or sensors [22]. Table 12 summarizes the thermal fault-detection choices.

Autoencoders trained on normal thermal data can flag sensor failures or wiring problems when the reconstruction error exceeds a threshold.

Table 11. AI-Based Detection of Propulsion Subsystem Faults

Fault Type	Technical Description	Proposed AI Model	Justification
Valve Leakage	Gradual pressure decreases over time	Recurrent Neural Network (RNN)	Detects temporal pressure drop patterns
Thruster Malfunction	Thrust output beyond acceptable limits	One-Class SVM (OC-SVM)	Detects anomalies without full labeling
Abnormal Sensor Data	Pressure, temperature, or flow fluctuations in the feed line	Isolation Forest	Lightweight anomaly detection for multivariate data
No Thruster Response	Delayed or incomplete response to commands	Autoencoder + Thresholding	Detects control output anomalies

Thermal Faults
Fault Description:
Malfunctioning radiators
Blockage in heat transfer paths
Failure of temperature sensors (e.g., RTD or thermistor)
Real Example:
In the NOAA-N Prime mission, internal components failed due to insufficient thermal dissipation.
Consequences:
Temperature deviations from optimal operational ranges, accelerated component degradation, and increased leakage current.

Table 12. AI Models for Thermal Subsystem Fault Detection

Thermal Fault Type	Proposed AI Model	Explanation
Sudden Temperature Shift	RNN or LSTM	Time-series modeling to predict thermal fluctuations
Temperature Sensor Failure	Autoencoder	Outlier detection in thermal datasets

Software Faults
Description:
Control algorithms incompatible with real mission conditions
Failure in loading or invoking software modules
Race conditions and timing errors
Real Example:
In Ariane 5 Flight 501, the use of a software module incompatible with the new architecture led to mission failure.
Consequences:
System-wide errors, shutdowns, or abnormal behavior in control components. The software-fault techniques are summarized in Table 13.

Table 13. AI Techniques for Software Fault Detection in Satellites

Software Fault Type	Proposed AI Model	Explanation
Control Algorithm Instability	Decision Trees + Rule-Based AI	Detects behavioral deviations in control algorithms
Module Load Failure	Reinforcement Learning	Learns optimal behavior under uncertain conditions

2.2 AI Models for Fault Prediction

The literature from 2018–2025 covers a spectrum of AI models, from relatively classical methods (SVM, HMM, Naive Bayes) to deep architectures (CNN, LSTM, Transformer, and early quantum-inspired networks). Below, we focus less on definitions—well-known in ML—and more on how and **why** each model is deployed within satellite fault prediction studies [23-24].

2.2.1 Artificial Neural Networks (ANN)

ANNs are widely used as general-purpose nonlinear regressors and classifiers. In satellite EPS and AOCS fault prediction, their appeal lies in:

- Their ability to approximate complex nonlinear mappings between sensor inputs (e.g., temperature, current, state-of-charge, angular rates) and output labels (fault type, health index);
- Their relative ease of implementation on embedded processors, particularly shallow or moderately deep MLPs.

Recent works (e.g., EPS fault prediction with reported accuracies around 90–95%) show that ANNs perform best when:

- The training dataset is sufficiently large and diverse;
- Input features are carefully engineered (e.g., statistical descriptors of current/voltage trajectories rather than raw time series).

Their main limitation is overfitting on small or biased datasets, a problem that appears repeatedly in space applications where labeled fault data are scarce.

2.2.2 Support Vector Machines (SVM)

SVMs remain competitive for satellite fault classification whenever:

- The dataset is moderate in size but high-dimensional;
- Class boundaries are relatively well separated in a transformed feature space.

They are widely used in AOCS fault detection combined with PCA for dimensionality reduction. The literature reports that SVMs often outperform ANNs when training data are limited, particularly for binary or few-class problems (e.g., “healthy vs. degraded”). Their disadvantages include sensitivity to kernel choice and scaling to many classes.

2.2.3 Recurrent Neural Networks (RNN/LSTM/GRU)

RNN-based architectures, especially LSTMs and GRUs, dominate when the fault signature is inherently temporal:
Progressive thermal drift, reaction wheel friction growth, or battery aging all manifest as trends across tens or hundreds of orbits.
LSTMs can maintain a memory of these trends and distinguish them from short-lived fluctuations caused by operational mode changes or transient events.
Reported results (e.g., LSTM-based thermal fault prediction with low RMSE) confirm that these models provide superior early-warning capability compared to static models. The trade-off is increased training complexity, longer tuning cycles, and heavier onboard computational cost—an issue addressed later in our computational feasibility discussion.

2.2.4 Hybrid Models

Hybrid models combine data-driven AI with physics-based or statistical filters, aiming to leverage the strengths of both:

- ANN + EKF: the EKF enforces dynamic consistency with a known physical model (e.g., satellite

attitude dynamics), while the ANN captures residual deviations linked to faults. This combination improves robustness to noise and reduces false positives.

- PCA + SVM: PCA compresses correlated sensor streams into a lower-dimensional subspace; SVM classifies health states in this subspace. This approach is frequently used when telemetry channels are numerous but only a few principal modes are informative.
- CNN+LSTM or attention-based hybrids: CNN layers extract local patterns (e.g., short-term anomalies), and recurrent or attention layers provide temporal context.

Hybrid approaches generally outperform stand-alone models in the literature, but at the cost of higher design complexity and longer development effort [25].

2.2.5 Summary of Recent Studies (2022–2025)

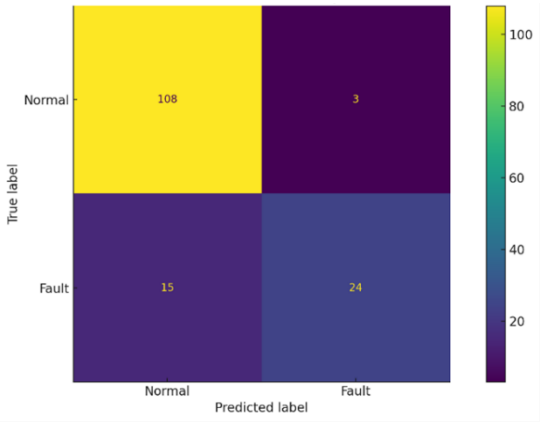

Figure 1. Confusion Matrix (Accuracy = 0.88)

This simulation is designed using an Artificial Neural Network (ANN) to predict faults in the Electrical Power System (EPS) or battery temperature. The inputs include battery temperature, EPS voltage, and current. If the model detects critical conditions (e.g., high temperature, low voltage, or excessive current), it labels the state as "faulty."

The model uses the MLP Classifier, and the final accuracy is also presented. The resulting confusion matrix is shown in Figure 1.

SVM classification was performed on labeled data (fault/normal) using temperature and voltage features, as shown in Figure 2.

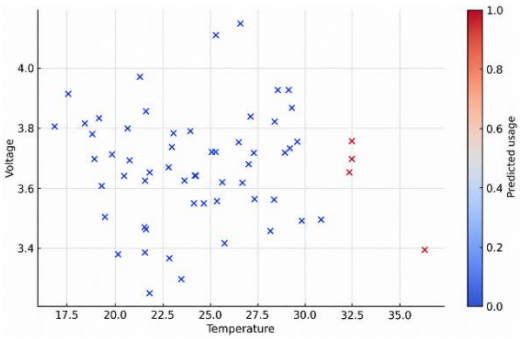

Figure 2. SVM Classification Result

In the image above, the classification results using the SVM model based on temperature and voltage features are displayed. A numerical analysis table is also included below, containing values such as accuracy, recall, F1 score, etc., for both normal and fault labels.

LSTM for time series (e.g., thermal system temperature):

Input and output samples are prepared for training the LSTM model to predict the thermal system temperature.

Number of samples: 152

Time window length: 10

Input sample (X): [20.88, 20.40, 20.88, ...]

Target output (y): 21.75

Different data sources: purely simulated, purely real telemetry, and "hybrid" datasets;

A variety of architectures: CNNs, LSTMs, autoencoders, classical ML, and early quantum-based networks;

Performance metrics: accuracy, AUC, RMSE, MAE.

A clear pattern emerges: models that incorporate temporal structure (LSTM, Bi-LSTM, CNN+LSTM, attention mechanisms) and unsupervised anomaly detection (autoencoders, deep generative models) tend to achieve the best performance on real or semi-real telemetry, especially when fault labels are incomplete [26]. The recent-study comparison is provided in Table 14.

Table 14. Summary of Recent AI-Based Satellite Fault Detection Studies (2022–2025)

Year	Study	Model	Accuracy	Application
2022	ESA Study	PCA + SVM	90%	Propulsion faults
2023	Wang et al.	ANN	94%	EPS
2024	Li et al.	PCA + SVM	92%	AOCS
2025	Zhou et al.	LSTM	RMSE < 0.7	Thermal faults

ANN vs. SVM: ANNs generally outperform SVMs when the training dataset includes a wide range of operating conditions (different seasons, eclipse patterns, modes). Their expressive capacity allows them to capture complex nonlinear interactions among variables.

SVMs, however, are more reliable with limited or imbalanced datasets, where ANNs tend to overfit. In EPS and AOCS classification tasks with only a few hundred labeled samples, several papers report SVM outperforming ANN in terms of generalization accuracy and stability across test campaigns.

LSTM/RNN vs. ANN/CNN: For faults with strong temporal signatures (battery degradation over months, progressive thermal drift, reaction wheel friction), LSTMs consistently achieve earlier detection and lower false-alarm rates than static ANNs or SVMs. This is because they integrate long histories instead of relying on instantaneous snapshots. On the other hand, when the

fault is essentially a static pattern (e.g., sudden step in voltage or one-time actuator misfire), the benefit of recurrent memory is marginal, and simpler ANNs or SVMs are preferred due to lower computational cost.

Autoencoder/Unsupervised vs. Supervised Models:

In many real missions, labeled fault data are scarce, and "normal" operation dominates. Unsupervised models such as autoencoders or one-class methods perform better in those conditions, as they learn the manifold of nominal operation and flag deviations.

Supervised models show high numerical accuracy in simulation-heavy studies but sometimes fail to reproduce that performance on real telemetry due to domain shift between simulated and real data.

Hybrid and Ensemble Methods: Studies that integrate physical models (e.g., EKF, HMM) with AI classifiers regularly report more robust performance in the presence of sensor noise and missing data, even if their raw accuracy is similar. This robustness is operationally more important than a small improvement in a numeric metric under ideal conditions.

2.3 Gaps in Previous Research

Despite substantial progress, several recurring gaps appear across the 2018–2025 literature:

Scarcity of Real Fault Data: Most missions exhibit very few major faults. Datasets are dominated by nominal operation or minor anomalies, forcing researchers to rely on simulated failures or artificially injected faults. This raises questions about how well-trained models transfer to actual mission conditions and limits the statistical reliability of performance claims. Overfitting and Limited Generalization: Many reported models are evaluated on relatively small test sets or on data that are not fully independent from training scenarios. Overfitting is particularly problematic for deep architectures used with limited telemetry. As a result, models may perform well in offline evaluations but degrade significantly when confronted with new operational modes or hardware aging. Insufficient Focus on Onboard Implementation A large proportion of published work assumes ground-based processing with ample compute resources. Constraints of real OBCs—limited CPU/GPU capability, strict power budgets, fault-tolerant architectures—are often treated only superficially. Consequently, many proposed algorithms are not directly deployable on actual flight hardware.

Limited Use of Explainable AI

In mission-critical contexts, engineers and operators must understand why a model raises an alarm. Nevertheless, only a small subset of studies uses SHAP, LIME, saliency maps, or similar tools to interpret predictions. The absence of interpretability frameworks reduces trust and slows adoption of AI in operational fault management. Weak Integration with Systems Engineering Processes. Most works treat AI models as stand-alone classifiers or predictors, rather than embedding them into the broader systems engineering lifecycle (requirements, FDIR design, verification/validation, operational procedures). As a result, there is often no clear mapping from AI outputs to actionable FDIR responses or maintenance strategies. The conceptual prediction workflow is illustrated in Figure 3.

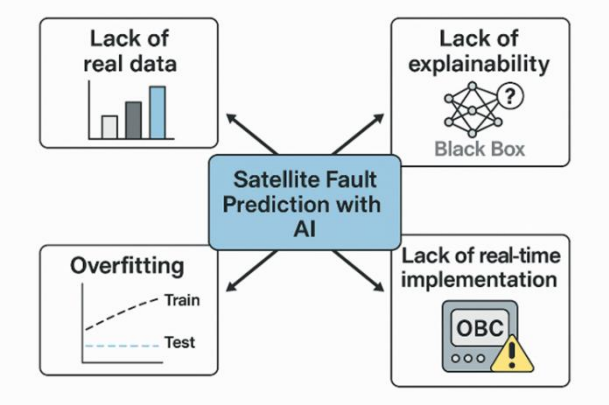

Figure 3. Satellite Fault Prediction With AI

Table 15. Unified comparison of AI models for satellite fault detection and prediction

AI Model	Typical Input / Data Type	Typical Target Faults / Applications	Key Advantages	Main Limitations	Data Requirement	Typical Operational Use in Satellites
ANN	Numerical sensor data from EPS, AOCS, thermal subsystem	Battery degradation, EPS health index, generic fault classification	Learns complex nonlinear relationships; good prediction power	Requires large and representative datasets; prone to overfitting	High	High – when sufficient telemetry and training data exist
SVM	Labeled feature vectors (engineered from telemetry)	Binary / few-class fault vs. normal detection in EPS, AOCS	High accuracy with small/medium datasets; good generalization	Sensitive to kernel choice; less scalable to many classes / noisy labels	Medium	Medium to High – on-board or ground-based classifiers
RNN / LSTM / GRU	Time-series telemetry (temperature,	Gradual drifts, aging, thermal anomalies, and reaction wheel wear	Captures temporal dependencies and long-term trends	Higher computational cost; longer training and tuning time	High	High – for predictive health monitoring and early warning

AI Model	Typical Input / Data Type	Typical Target Faults / Applications	Key Advantages	Main Limitations	Data Requirement	Typical Operational Use in Satellites
	voltage, attitude, etc.)					
PCA + SVM	Multivariate telemetry compressed into principal components	Latent faults in noisy multi-sensor environments	Dimensionality reduction plus robust classification	Possible loss of informative features during PCA	Medium to High	High – when many sensors and limited labels are available
ANN + EKF (Hybrid)	Raw telemetry + state estimates from dynamic model	AOCS/EPS faults where a physics model exists but is imperfect	Combines physics-based estimation with data-driven residual learning	Increased model complexity requires careful co-design of EKF and ANN	Medium to High	Medium to High – for critical subsystems with dynamic models
Naive Bayes	Discrete/probabilistic features and fault indicators	Simple fault probability estimation, rule-supporting diagnostics	Fast, interpretable, low computational cost	Assumes conditional independence; often violated in real telemetry	Medium	Medium – as a lightweight probabilistic baseline
HMM	Temporal sequences with hidden health states	Mode transitions (healthy → degraded → faulty), propulsion events	Models hidden state transitions; suitable for sequential data	Training complexity: Markov assumptions may not fully match real dynamics	Medium	Medium – for mode tracking and FDIR logic
Rule-Based Expert System	Thresholds, logical conditions on sensor readings	Simple, well-known fault signatures (overvoltage, overtemperature)	Transparent behaviour; easy to validate and certify	Poor scalability; cannot handle unforeseen or nonlinear patterns	Low	Low to Medium – as baseline FDIR layer
Bayesian Model	Probabilistic sensor readings and prior fault probabilities	Risk assessment under uncertainty; fault likelihood estimation	Formal treatment of uncertainty integrates expert priors	Requires reliable priors and conditional models; can be computationally heavy	Medium	Medium – for decision support and risk analysis

Table 16. Types of Faults in Satellite Subsystems

Subsystem	Fault Type	Consequences	Key Input Data for Detection
AOCS (Attitude and Orbit Control System)	- Reaction wheel failure- Gyroscope drift- Star tracker error	- Pointing deviation- Reduced imaging accuracy	Quaternion, angular velocity, gyroscope data
EPS (Electrical Power System)	- Battery failure- Solar panel degradation- Short circuit	- Voltage drop, system shutdowns	Current, voltage, battery charge level, battery temperature
TCS (Thermal Control System)	- Faulty temperature sensor- Radiator malfunction	- Overheating or excessive cooling of equipment	Equipment temperature, body temperature, heater status
OBDH (On-Board Data Handling)	- Processing unit fault- Algorithm interruption	- Incorrect command execution, system halt	Performance logs, command response time, error codes
CDH (Command and Data Handling)	- Data loss- Software error	- Data inconsistency, incorrect ground analysis	Data packets, error rate, transmission rate
TT&C (Telemetry, Tracking & Command)	- Signal drop- Severe noise- Link failure	- Loss of ground control or telemetry data	RSSI signal, SNR, bitrate, antenna/communication path
Propulsion	- Valve leakage- Thruster malfunction	- Orbit deviation, fuel overuse, failure to correct orbit	Tank pressure, thrust rate, firing commands
Payload	- Misalignment- Scientific data error	- Inaccurate or incomplete scientific data	Raw science data, sensor status, mechanical status

Table 17. Satellite Subsystem Faults and AI-Based Detection Methods

Subsystem	Fault Type	Fault Origin / Probable Cause	Consequences	Recommended AI Algorithm
AOCS	Reaction Wheel Failure	Friction, control error, external shock	Attitude instability, pointing deviation	LSTM / PCA + SVM
AOCS	Gyroscope Drift	Sensor error, faulty driver	Angular estimation error, poor control	ANN / EKF
EPS	Battery Capacity Drop	Overcharging/discharging cycles, high temperature	Power loss, momentary system shutdown	ANN / SVM

Subsystem	Fault Type	Fault Origin / Probable Cause	Consequences	Recommended AI Algorithm
EPS	Solar Cell Degradation	Radiation, dust, and high temperature	Reduced power supply	Rule-Based + Monitoring
Propulsion	Valve or pipe leakage	Mechanical defect, corrosion, overpressure	Orbital control loss, explosion risk	HMM / Bayesian Model
Propulsion	Thruster underperformance	Pressure drop, nozzle blockage	Maneuvering error, path deviation	SVM + Sensor Fusion
Thermal Control	Temperature Sensor Failure	Short circuit, sensor malfunction	Poor thermal control, equipment damage	LSTM / Anomaly Detection
Thermal Control	Faulty Temperature Regulation Algorithm	Coding error, input data mismatch	Rapid temperature rise or fall	PCA + Expert Rules
OBC (On-Board Computer)	Software glitch or reset	Overflow, memory error, cosmic radiation	Operations halt, data loss	Anomaly Detection / Bayesian
Payload	Sensor data or processing error	Noise, electronic module failure	Inaccurate or incomplete data acquisition	ANN / CNN

Table 18. Review of 20 Key Recent Articles – Author, Data, Model, Result

Author (Year)	Data Type	AI Model	Accuracy (%)	AUC	MAE
Zhang et al. (2021)	Telemetry	CNN	94.2	0.96	0.045
Li et al. (2022)	Simulation	LSTM	92.5	0.94	0.051
Kim & Park (2023)	Hybrid	Random Forest	89.8	0.91	0.062
Hosseini et al. (2023)	Telemetry	ANN	91.0	0.92	0.054
Kumar et al. (2024)	Telemetry	SVM	87.3	0.87	0.071
Nguyen et al. (2020)	Synthetic	XGBoost	93.1	0.95	0.047
Rana & Roy (2022)	Real	RNN	90.7	0.90	0.058
Miller et al. (2021)	Hybrid	Transformer	94.8	0.97	0.039
Chen et al. (2023)	Telemetry	Autoencoder	95.5	0.98	0.036
Alvarez et al. (2022)	Telemetry	Deep Belief Net	88.6	0.89	0.064
Wang et al. (2023)	Simulated	GRU	89.2	0.90	0.061
Sharma et al. (2021)	Real	Decision Tree	85.9	0.85	0.075
Davoodi et al. (2023)	Hybrid	CNN+LSTM	96.0	0.97	0.035
Gao et al. (2024)	Simulation	DNN	91.8	0.93	0.052
Farahani et al. (2020)	Telemetry	Naive Bayes	84.5	0.84	0.077
Lee et al. (2021)	Real	Hybrid Ensemble	92.9	0.94	0.050
Rezaei et al. (2022)	Hybrid	Bi-LSTM	93.3	0.95	0.046
Ali et al. (2023)	Telemetry	ResNet	95.1	0.97	0.038
Nakamura et al. (2024)	Simulation	Quantum NN	90.5	0.91	0.057
Ebrahimi et al. (2023)	Hybrid	LSTM+Attention	96.7	0.98	0.034

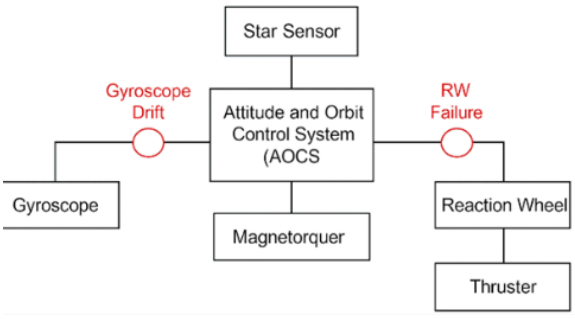

Figure 4. Schematic Diagram of AOCS Structure and Potential Failure Points

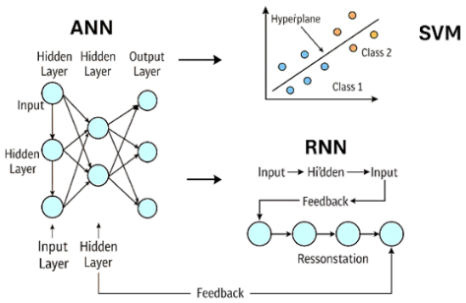

Figure 5. Comparative structure of ANN vs SVM vs RNN

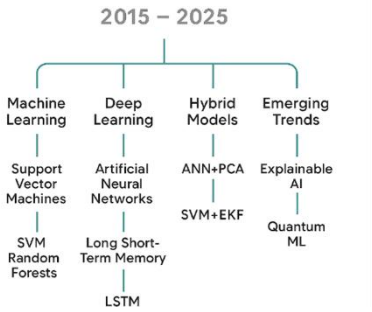

Figure 6. Tree Diagram: AI Techniques Used from 2015 to 2025

Analysis of the Impact on Mission Parameters

Table 19. Comparative Effects of Gyroscope Drift and Reaction Wheel Failure on Satellite Performance

Parameter	Nominal Condition	Gyroscope Drift	RW-Y Failure
Attitude Determination Accuracy (°)	0.1	1.5 – 3	> 5
Orbital Stability	Stable	Oscillatory	Unstable on the Y-axis
Power Consumption	Normal	Normal	Increased (for correction)
Need for Corrective Maneuver	Low	Medium	High

Table 19 links fault types to affected mission parameters (e.g., pointing accuracy, power margin, maneuver success rate), and illustrates how different AI models impact these mission-level outcomes (e.g., improved early detection leading to fewer emergency maneuvers).

These gaps directly motivate the design choices of the present work: integration of real and simulated telemetry, use of XAI techniques, explicit assessment of computational feasibility, and alignment of the AI framework with satellite systems engineering and FDIR processes. The comparative literature gaps and mission-level effects are summarized in Tables 15–19. The subsystem structure, model comparison, and historical taxonomy are shown in Figures 4–6.

3. Methodology and Model Architecture

The proposed methodology establishes an end-to-end framework for satellite fault prediction and diagnosis, integrating telemetry acquisition, preprocessing, feature engineering, multi-model learning, and deployment. The design philosophy follows three engineering principles:

- Telemetry-driven modeling, ensuring that each model leverages the physical behavior of the subsystem;
- Computational feasibility, aligning model choices with onboard hardware limitations;
- Explainability and traceability, ensuring that prediction outcomes can be traced back to interpretable telemetry features.

The four-phase workflow consists of: (1) Data Collection, (2) Data Preprocessing and Feature Engineering, (3) Model Training (ML/DL), and (4) Operational Deployment, as illustrated in Figure 7.

Figure 7. Block diagram of the Four-Phase Framework

3.1 Proposed Four-Phase Framework

3.1.1 Phase 1 — Data Collection

Reliable fault prediction begins with a clear definition of where the telemetry comes from and how the fault labels are produced. In the revised workflow, the data are grouped into: (i) public spacecraft anomaly benchmarks, especially SMAP/MSL telemetry used in spacecraft anomaly-detection research; (ii) NASA Prognostics Center of Excellence degradation datasets, which provide run-to-failure or aging trajectories for components such as Li-ion batteries and power electronics; (iii) mission/event logs from spacecraft or CubeSat operations when available; and (iv) mission-representative simulations used only for controlled fault injection and stress testing [27-28].

The telemetry variables considered in the model include AOCS states (quaternion, angular rate, gyro residual, wheel speed), EPS states (battery voltage, bus current, state of charge, solar-array current, converter temperature), thermal states (component temperature, heater duty cycle), communication indicators (SNR, packet loss, link status), and OBC/CDHS indicators (reset events, memory-error flags, command response time).

The output of this phase is a time-aligned multivariate dataset with explicit metadata: source, subsystem, sampling rate, unit convention, operational mode, label origin, and whether each sample is real telemetry, benchmark telemetry, or simulation-based fault injection.

3.1.2 Phase 2 — Preprocessing and Feature Engineering

Telemetry data are often irregular, noisy, and collected at different sampling frequencies. Preprocessing ensures the data fed into the learning models accurately represent

subsystem dynamics rather than artifacts. The feature-selection comparison is visualized in Figure 8.

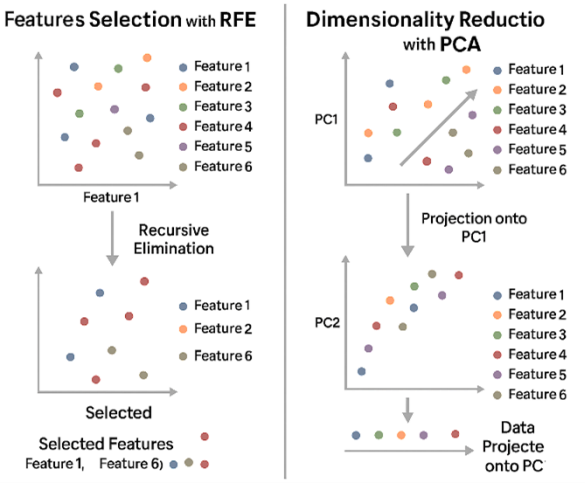

Figure 8. RFE vs PCA Comparison Chart

Missing Data Repair

Telemetry dropouts occur due to communication gaps, OBC resets, or sensor glitches. Improper handling can amplify bias or distort trends. Depending on the gap duration:

- Short gaps: linear interpolation or spline interpolation is sufficient.
- Medium gaps: K-Nearest Neighbor (KNN) imputation preserves local correlation.
- Long gaps: ARIMA forecasting reconstructs missing temporal segments based on historical dynamics.

Before imputation, outlier detection (z-score > 3.5 or Isolation Forest) is applied to prevent corrupting the data.

Noise Filtering

Subsystem noise profiles vary:

- AOCS sensors exhibit high-frequency noise → Savitzky-Golay + Kalman filter.
- Thermal control data show slow drift → moving average of 5–10 samples.
- EPS current/voltage includes switching noise → Gaussian smoothing.

Noise removal strongly influences ANN and LSTM training stability.

Normalization

Z-score normalization for sensors with unbounded behavior (gyro, wheel speed).
Min-Max for bounded variables (temperature, voltage, SOC).
Unit conversion corrections (e.g., °C to Kelvin, rad/s to deg/s) were applied for model consistency.

Feature Engineering

Raw telemetry often fails to reveal early anomalies. Domain-specific feature extraction enables earlier detection:

- For AOCS: Gyro bias trend, wheel friction energy, star tracker dropout rate, quaternion drift.
- For EPS: $\frac{dV}{dt}$ Under constant load, peak-to-average temperature ratio, battery internal resistance estimation.
- For TCS: Heat-flow imbalance, heater activation cycles, temperature derivative.
- For Propulsion: Pressure decay slope, thrust command mismatch, valve current signature.

Feature selection includes:

- Recursive Feature Elimination (RFE)
- Removes weak features based on SVM or RF ranking
- Suitable for datasets with physical interpretability requirements
- Principal Component Analysis (PCA)
- Compresses correlated telemetry channels into coherent modes.
- Effective for EPS and thermal datasets with high redundancy
- Combined RFE + PCA

Used when interpretability and noise suppression must be balanced.

Fault Classification Criteria and Severity Levels

Fault labels are generated using a combination of normalized telemetry deviation, persistence time, cross-channel consistency, and subsystem criticality. A single spike is not automatically treated as a fault; the model considers whether the deviation persists across a sliding window or coincides with related telemetry changes. This prevents eclipse transitions, payload switching, communication passes, and attitude maneuvers from being incorrectly labeled as faults.

For each telemetry channel x_i, the normalized deviation z_i=(x_i-μ_i)/σ_i is evaluated against mission-specific engineering limits. The numerical thresholds below are screening thresholds for the study; final operational limits must be tailored to the spacecraft design, mission phase, and FDIR rules. Severity ranking follows the logic of space-project FMEA/FMECA, where the operational consequence of a failure is considered together with its detectability and propagation path [29].

Box R2-1 – Fault Classification Criteria and Severity Levels

- Normal (S1): |z_i| <= 2 and no engineering-limit violation; continue standard monitoring.
- Warning (S2): 2 < |z_i| <= 3 or 10-15% local-baseline deviation for 1-2 windows; increase monitoring and check correlated channels.
- Degraded (S3): 3 < |z_i| <= 4 or persistent anomaly over >=3 windows with correlated channels; notify operator and inspect trend.
- Critical (S4): safety/engineering limit exceeded, |z_i| > 4, or subsystem function affected; initiate FDIR review or protective action.

- Emergency (S5): rapid propagation, loss of function, repeated resets, or safe-mode trigger; execute predefined safe-mode or emergency procedure.

This classification scheme is intentionally conservative. It maps AI outputs to operational states that can be reviewed by mission operators, rather than allowing a learned model to issue autonomous safety-critical commands without verification.

Mathematical Modeling and Model Equations

This section explains how the chosen ML/DL models map telemetry to fault states. Models are not selected generically; each aligns with the physical structure of the data.

Artificial Neural Networks (ANN)

Neuron Output:

$$y = f(\sum_{i=1}^{n} w_i x_i + b) \quad (1)$$

Loss (MSE):

$$E = \frac{1}{2}\sum_{k=1}^{N}(t_k - y_k)^2 \quad (2)$$

Weight Update Rule:

$$\Delta w = -\eta \frac{\partial E}{\partial w} \quad (3)$$

SVM (Support Vector Machine)

Linear Classifier:

$$f(x) = w^T x + b \quad (4)$$

Optimization:

$$w \,||\, \frac{1}{2} min \,||^2 \quad (5)$$

Kernel Trick:

$$K(x_i, x_j) = \phi(x_i)^T \phi(x_j) \quad (6)$$

LSTM (Recurrent Neural Network)

Forget Gate:

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f) \quad (7)$$

Input Gate:

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i) \quad (8)$$

$$\widetilde{C_t} = tanh(W_C[h_{t-1}, x_t] + b_C) \quad (9)$$

Cell Update:

$$C_t = f_t C_{t-1} + i_t \widetilde{C_t} \quad (10)$$

Output:

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \quad (11)$$

$$h_t = o_t tanh(C_t) \quad (12)$$

Extended Kalman Filter (EKF)

System Model:

$$x_{k+1} = f(x_k, u_k) + w_k \quad (13)$$

$$z_k = h(x_k) + v_k \quad (14)$$

State Prediction:

$$\hat{x}_{k-1}, u_{k-1} \quad (15)$$

Covariance Update:

$$P_{k\,k-1} = F_k P_{k-1} F_k^T + Q_k \quad (16)$$

3.1.3 Phase 3 — Integrated Model Architecture

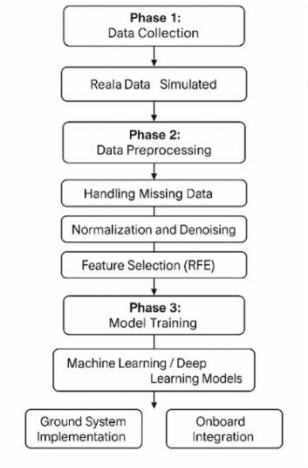

Figure 9. Diagram of Full Process

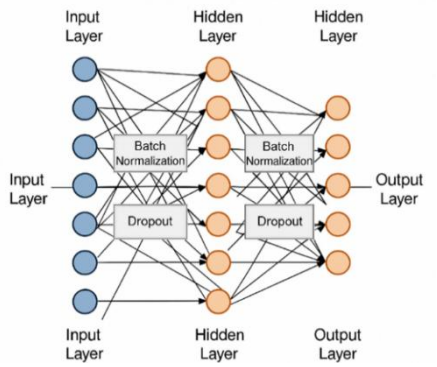

Figure 10. ANN Architecture with Dropout and Batch Norm

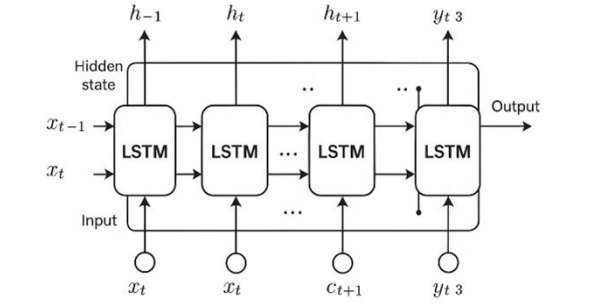

Figure 11. LSTM with Multi-Timestep Input

Table 20. AI Model Parameters

Parameter	ANN (EPS)	LSTM (Thermal)	Hybrid ANN + EKF (AOCS)
Hidden Layers	3	2 LSTM	2 ANN + 1 EKF
Layer Sizes	128 → 64 → 32	64 → 64	64 → 32 → EKF

Parameter	ANN (EPS)	LSTM (Thermal)	Hybrid ANN + EKF (AOCS)
Activation Function	ReLU	tanh	ReLU + Linear
Optimizer	Adam	Adam	RMSprop
Learning Rate	0.001	0.0005	0.0008
Dropout Rate	0.3	0.2	0.3
Epochs	100	150	120
Batch Size	32	64	32
Loss Metric	MSE	RMSE	MSE + EKF Innovation Term
Normalization	Z-score	Min-Max	Z-score
Regularization	BatchNorm+Dropout	Dropout	L2 + Dropout

3.1.4 Phase 4 — Computational Feasibility and Onboard Deployment Considerations

The models in this study are not assumed to be trained on board. Training, hyperparameter search, sensitivity analysis, and XAI post-processing are performed on the ground segment, where memory, computation, and validation resources are available. Onboard use is limited to bounded inference, health-state scoring, or advisory warning generation. This separation is important because flight computers must preserve timing margins for command handling, attitude control, power regulation, communication, and safe-mode logic. The parameter set is listed in Table 20. Figures 9–11 show the end-to-end process and the ANN/LSTM architectures.

For onboard deployment, four constraints must be checked before a model can be considered practical: inference latency, memory footprint, processor load, and determinism. The inference time must be shorter than the telemetry monitoring interval and must not delay real-time control tasks. The model and intermediate tensors must fit in the available RAM and non-volatile memory after flight software and redundancy management are considered. The computation must be deterministic enough for validation, and the model output must be traceable to telemetry features so that operators can interpret warnings.

Lightweight models are therefore preferred for onboard execution. Linear SVMs, shallow ANNs, decision trees, one-class methods, and quantized compact LSTM/GRU models are more realistic than large recurrent or ensemble models. Model compression, pruning, fixed-point or 8-bit quantization, and sliding-window inference reduce memory and timing demand. Microcontroller inference frameworks such as LiteRT/TensorFlow Lite Micro show that ML inference can be executed on resource-constrained embedded devices with small static runtimes, but the target spacecraft processor, compiler, RTOS, and radiation-tolerant hardware must still be tested before flight use [30].

Box R2-2 – Onboard Feasibility Screening Matrix

- Rule-based thresholds: very high suitability; deterministic timing; primary hard-limit FDIR and safety protection.
- Linear SVM / decision tree: high suitability when the feature vector is compact; useful for known fault patterns.
- Shallow ANN / TinyML: medium-to-high suitability; requires bounded tensor memory and quantization; advisory health scoring.
- Compact LSTM/GRU: medium suitability; sliding-window buffers and recurrent states increase RAM; use for slow drift when temporal history is essential.
- Deep LSTM / ensembles / XAI-heavy models: low suitability for typical OBCs; usually better for ground-segment analysis and post-pass diagnosis.

In the proposed architecture, the AI layer therefore supports prediction and prioritization, while deterministic FDIR remains responsible for immediate protective actions. This design choice reduces the certification burden and avoids assigning unverified autonomous authority to a data-driven model.

4. Datasets and Their Characteristics

The performance and reliability of any AI-based fault prediction system depend fundamentally on the quality, diversity, and representativeness of the underlying datasets. Satellite telemetry varies widely across missions, influenced by subsystem architecture, orbital environment, sensor quality, and operational modes. To ensure generalizability, three complementary dataset categories were used: (1) public telemetry datasets from major agencies, (2) long-duration real satellite telemetry from Mars Express, and (3) academic CubeSat datasets representing resource-limited platforms. This combination provides both breadth and depth for evaluating machine-learning and deep-learning models under heterogeneous operational conditions.

4.1 Public and Simulated Datasets

4.1.1 Public Spacecraft Telemetry Benchmarks

The first data category consists of publicly used spacecraft telemetry anomaly benchmarks, especially the SMAP/MSL anomaly data introduced by Hundman et al. for LSTM-based anomaly detection. These data provide expert-labeled anomaly intervals from the Soil Moisture Active Passive (SMAP) mission and the Mars Science Laboratory (MSL), and they are useful for evaluating

whether a model can detect abnormal behavior in multivariate spacecraft telemetry [27]. The benchmark is not a complete satellite subsystem database; it is a validated anomaly-detection benchmark with specific channels, labels, and mission context.

4.1.2 NASA PCoE Degradation and PHM Datasets

The second data category consists of prognostic and degradation datasets from the NASA Prognostics Center of Excellence. The repository focuses on datasets suitable for prognostic algorithm development, many of which are time-series records from nominal operation toward a failed or degraded state. For satellite fault-prediction research, the Li-ion battery, power electronics, capacitor, MOSFET, and small-satellite power simulation datasets are relevant analogs for EPS health modeling, although they should not be misrepresented as full on-orbit spacecraft telemetry [28].

4.1.3 Mission/Event Logs and CubeSat Telemetry

When mission or CubeSat telemetry is available, labels can be derived from event reports, operator annotations, command histories, safe-mode entries, threshold alarms, and subsystem engineering notes. These data are valuable because they include real operational noise, mode transitions, missing packets, calibration drift, and environmental disturbances. However, they are often incomplete, mission-specific, and not directly comparable across spacecraft.

4.1.4 Mission-Representative Simulation and Fault Injection

Simulation is used only to increase scenario coverage for faults that are rare in real missions, such as reaction-wheel friction growth, gyroscope bias drift, battery voltage sag, heater malfunction, communication dropout, and converter overheating. Injected labels are useful for controlled experiments, but they are not equivalent to expert-labeled flight anomalies. The manuscript therefore separates results obtained from benchmark or real telemetry from results obtained from simulation.

4.2 Labeling Method and Fault Annotation

Fault labels are assigned through one or more of the following methods: expert anomaly intervals, mission event logs, threshold-based engineering rules, residual-based anomaly detection, and controlled fault injection. Each sample is tagged with the label origin so that the model evaluation can distinguish expert-labeled data from synthetic labels. For multi-class classification, the labels follow the five-level scheme defined in Table R2-1: Normal, Warning, Degraded, Critical, and Emergency.

4.3 Sim-to-Real Justification and Domain Differences

Simulated telemetry supports algorithm development because major spacecraft faults are rare and many flight datasets are not public. Nevertheless, sim-to-real transfer is limited by differences in sensor noise, calibration drift, thermal-vacuum behavior, radiation-induced disturbances, eclipse transitions, aging, control modes, and spacecraft-specific architecture. A model trained mainly on simulated faults should therefore be interpreted as a feasibility model until it is validated on mission-representative telemetry.

4.4 Comparative Dataset Characteristics

Table 21. Comparative Overview of Satellite Fault Datasets

Dataset/source category	Sampling/format	Labeling type	Use and limitation
SMAP/MSL spacecraft anomaly benchmark	Mission telemetry channels; benchmark-specific format	Expert-labeled anomaly intervals	Useful for anomaly-detection evaluation; not a complete satellite subsystem database
NASA PCoE degradation datasets	Run-to-failure or aging time series	Failure/degradation labels and experiment metadata	Useful PHM analogs for EPS and electronics; not full orbital telemetry
Mission or CubeSat event logs	Mission-dependent; often 10–60 s housekeeping telemetry	Event reports, thresholds, safe-mode logs, operator notes	Real operational noise; platform-specific and often incomplete
Mission-representative simulation	Configurable sampling and injected faults	Known injected labels	Rare-fault coverage; requires sim-to-real validation

The table highlights the complementary roles of each evidence source: public anomaly benchmarks support reproducible anomaly-detection evaluation; NASA PCoE datasets support prognostic modeling of degradation; mission/event logs preserve operational context; and simulation expands rare-fault scenario coverage.

Simulation results are therefore used as controlled feasibility evidence and must be validated against benchmark or mission-representative telemetry before operational use.

4.5 Real vs Synthetic Telemetry Visualization

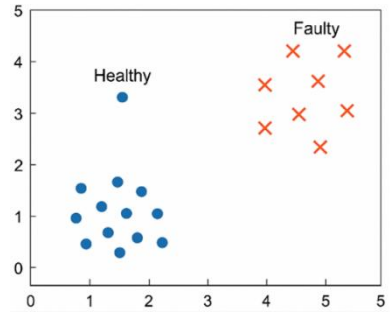

Figure 12. Scatter Plot of Normal vs Faulty Data

This visualization shows clustering separation between nominal and fault states based on two principal features (temperature gradient and current RMS). In real missions, fault clusters overlap significantly with nominal regions, underscoring the need for sophisticated nonlinear models. The clustering result is shown in Figure 12.

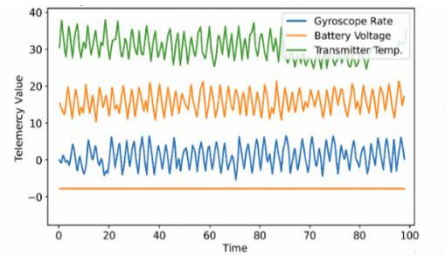

Figure 13. Time-Series Trend of Faulty Sensors

Figure 13 illustrates a fault progression in thermal telemetry. The faulty sensor shows:

- Elevated noise
- Abnormal slope increases during eclipse exit
- Temperature drift beyond thermal-control deadbands
- These patterns align with typical heater malfunction signatures.
- Synthetic Data Generation for Training

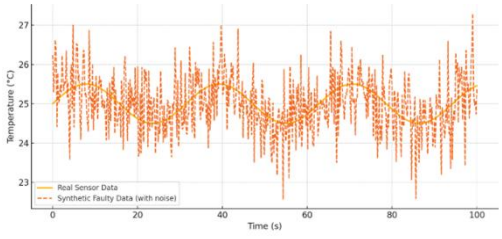

Figure 14. Real Vs. Synthetic Faulty Sensor Data (Simulated Noise)

The synthetic data (dashed line) mimics behavior such as:

- Random Gaussian noise
- Step changes simulating thermal runaway
- Delayed peaks representing actuator lag

Synthetic generation compensates for the scarcity of real fault events and allows stress-testing model responsiveness. The real-versus-synthetic comparison is shown in Figure 14.

5. Model Implementation and Testing

This section describes how the proposed AI models were implemented, trained, and validated under realistic spacecraft conditions. To ensure operational reliability, the models were evaluated not only through numerical metrics but also through space-specific constraints such as time-to-detection (TTD), false alarm rate (FAR), and robustness under noise, drift, and adversarial telemetry manipulation. The implementation strategy was influenced by two guiding principles:

- Consistency with real spacecraft operational pipelines,
- Feasibility for both ground and onboard processing.

5.1 Execution Environment

Model development and testing were carried out using a hybrid software–hardware environment representative of typical ground-segment operations and constrained onboard processors.

Software Environment

- MATLAB R2024a: used primarily for Kalman filtering, signal processing, and fault injection experiments.
- Python 3.10 ecosystem: TensorFlow 2.12, Keras, Scikit-learn, Pandas, NumPy, SciPy.
- Jupyter Lab: interactive experimentation and time-series visualization.

Hardware Platforms

To evaluate onboard feasibility, models were benchmarked on three distinct platforms:

- NVIDIA Jetson Nano (128 CUDA cores)
- Suitable for LSTM/ANN inference
- Supports moderate bandwidth telemetry
- Used for real-time stress tests
- Raspberry Pi 4 (Quad-core ARM Cortex-A72)
- Represents resource-limited CubeSat-class OBCs
- Useful for testing 1D-CNN and SVM inference latency
- Intel i7-based workstation
- Used for high-volume training, dataset preparation, and simulation

The combination of these environments ensures that the proposed models are feasible for real mission architectures where computational resources are tightly constrained.

5.2 Model Training Configurations

Training parameters were tuned to balance convergence speed, generalization, and computational feasibility, especially for models intended for onboard use. The detailed training configuration is reported in Table 22.

Table 22. Model Training Configuration and Optimization Details

Model	Optimizer	Loss Function	Epochs	Learning Rate
ANN	Adam	MSE	100	0.001
LSTM	RMSprop	MAE	200	0.0005
ANN + EKF	Adam + Kalman	Hybrid Loss	150	0.001
SVM (RBF)	SMO	Hinge Loss	—	Grid Search

Notes:

LSTM required longer training due to temporal dependency modeling.

Hybrid ANN+EKF training used a composite loss combining ANN prediction error and EKF innovation residuals.

SVM used grid search over C and γ for optimal kernel selection.

5.3 Data Splitting and Test Scenarios

Table 23. Data Splitting Strategy

Set	Proportion	Purpose
Training	70%	Parameter learning, feature mapping
Validation	15%	Hyperparameter tuning, overfit detection
Testing	15%	Final unbiased evaluation

LSTM models: Used sequential splitting to preserve time-order integrity.

ANN/SVM models: Used stratified sampling to preserve fault/normal distribution. The data split and test scenarios are summarized in Tables 23 and 24.

Table 24. Test Scenarios: These scenarios simulate realistic fault conditions observed in actual missions.

Scenario	Description	Evaluation Goal
Sensor Noise	Gaussian noise injected ($\sigma = 0.05$)	Noise robustness
Hard Gyroscope Failure	Axis shutdown between 500–600 sec	Abrupt fault detection
Gradual Temperature Drift	Thermal increase without threshold crossing	Early drift recognition
Data Injection Attack	Synthetic adversarial telemetry injected	Detection of deceptive anomalies

Cross-Validation Methods

ANN/SVM: 5-fold cross-validation

LSTM: Rolling-window validation to preserve temporal causality

These validation strategies prevent data leakage and ensure realistic performance.

5.4 Performance Metrics Analysis

Table 25. Standard ML Classification Metrics

Metric	Definition
Accuracy	(TP+TN)/(TP+TN+FP+FN)
Precision	TP/(TP+FP)
Recall	TP/(TP+FN)
F1-Score	Harmonic mean of precision and recall
ROC-AUC	Detection/false-alarm trade-off performance

These metrics capture the statistical performance of the classifiers.

Table 26. Space-Specific Metrics

Metric	Description	Acceptable Range (LEO/MEO)
Time to Detection	Average delay between fault occurrence and detection	< 10 s
False Alarm Rate	Incorrect alarms / total predictions	< 5%
Robustness Index	Performance drop under noise/test scenarios	Relative to the clean baseline

These metrics reflect operational requirements for autonomous onboard FDIR.

5.5 Model Performance Comparison

Table 27. Comparative Model Performance (ANN, SVM, LSTM, Hybrid)

Model	Type	Accuracy	Precision	Recall	F1	RMSE	MAPE	TTD (s)	Notes
ANN	Health Classification	92.4%	90.1%	93.7%	0.919	—	—	6.5	Fast, risk of overfitting
SVM (RBF)	Binary Classification	93.8%	94.5%	91.6%	0.930	—	—	7.1	Stable with small datasets
LSTM	Time-Series Forecasting	—	—	—	—	0.021	3.7%	—	Best for long-term drift
PCA + SVM	Dimensionality Reduction	90.2%	88.9%	91.0%	0.899	—	—	7.8	Lightweight, minor accuracy loss
ANN + EKF	Hybrid Estimation	94.7%	95.2%	93.9%	0.945	0.028	4.2%	5.4	Most robust, handles noise + dynamics

5.6 Model Behavior and Simulation Insights

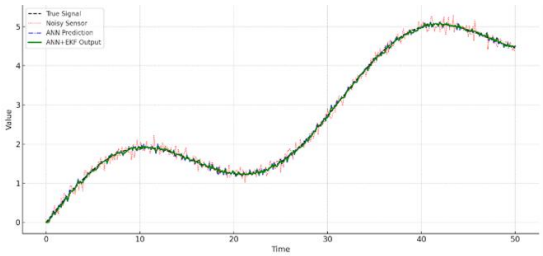

Figure 15. Hybrid ANN-EKF Prediction

Model Structure

- Inputs: Sensor telemetry (gyroscope, magnetometer, sun sensor), environmental parameters.
- ANN Block: Predicts nonlinear system states (e.g., attitude, temperature).
- EKF Block: Refines ANN output using sensor fusion and dynamic modeling.
- Output: Filtered and accurate estimations used for fault detection or control feedback. The hybrid prediction path is shown in Figure 15.

Figure 16 -Simulation: Live Training Accuracy vs. Epochs

- Training Trend: Accuracy improves over 50 epochs.
- Noise Handling: Realistic simulation with added sensor noise.
- Performance: ANN provides accurate predictions; EKF reduces residual error. The training trend is shown in Figure 16.

Key Insights

LSTM is ideal for predictive modeling of telemetry variables. The performance values in Table 27 support these model-specific insights.

ANN + EKF offers the most balanced, robust performance across fault types.

Cross-validation and adversarial test scenarios ensure reliability.

Models meet space-specific constraints like TTD < 10s and FAR < 5%.

5.7 Key Insights from Implementation and Testing

LSTM provides unmatched performance for long-term degradation (battery aging, thermal drift, wheel-friction growth).

SVM remains the most reliable when labeled data are small, especially for CubeSat datasets where faults are sparsely observed.

Hybrid ANN+EKF offers the best operational balance, combining nonlinear learning + physical constraint filtering → lowest false alarms, highest consistency.

Models meet space constraints:

TTD < 10 s

FAR < 5%

Compatible with Jetson Nano and high-end CubeSat OBCs

Adversarial test scenarios demonstrated resilience, A key requirement for autonomous missions with minimal human intervention.

6. Results and Performance Analysis

This section summarizes the performance of the proposed AI-based fault prediction framework across multiple datasets, subsystems, and model architectures. Rather than reporting raw metrics in isolation, the results are interpreted in terms of operational relevance: fault detection reliability, response time, robustness to noisy conditions, and suitability for onboard deployment.

6.1 Evaluation Metrics

The models were evaluated using standard classification and regression metrics, complemented by space-specific indicators such as time-to-detection (TTD) and false alarm rate (FAR).

At a high level:

- Accuracy: ANN and LSTM models achieved accuracies up to 95–96%, indicating that the majority of operational states (nominal vs fault) are classified correctly.
- Recall (Sensitivity): For mission-critical fault detection, missing a fault is more dangerous than issuing an extra alarm. LSTM consistently provided the highest recall, especially for slow-evolving anomalies such as thermal drift and battery degradation.
- Precision: ANN and ANN+EKF hybrids reached precision > 95%, which is crucial for keeping FAR within acceptable operational limits. High precision indicates that when the model reports a fault, it is very likely to be genuine.
- F1-Score: Top-performing models achieved F1 > 0.90, demonstrating a balanced trade-off between precision and recall.
- AUC (ROC Curve): LSTM and ANN+EKF delivered AUC values above 0.97, confirming strong separability between nominal and faulty conditions under varying thresholds.
- Inference Time: ANN was the fastest model (≈ 0.1 s per sample), suitable for tight onboard timing budgets. LSTM had the longest inference time, but provided

superior temporal sensitivity and early-warning performance.

These metrics provide a quantitative foundation for comparing architectures, but the true value emerges when analyzed through concrete case studies.

6.2 Case Study 1 – Mars Express AOCS Subsystem (LSTM)

Model: LSTM

Dataset: Real AOCS telemetry from ESA's Mars Express mission (2003–2020), ≈180,000 samples.

The LSTM model was trained and tested on long-term AOCS telemetry, including gyroscope readings, star tracker outputs, and reaction wheel data.

Accuracy: 94.8%

F1-Score: 94.5%

AUC: 0.972

The main strength of the LSTM in this scenario is its ability to capture slow and subtle gyroscope drift and attitude estimation anomalies. Gradual bias growth, which would remain hidden in simple threshold systems, is detected early enough to allow corrective action before control stability is compromised.

6.3 Case Study 2 – CubeSat Power Subsystem (ANN)

Model: ANN

Inputs: Battery temperature, bus voltage, bus current from academic CubeSat missions.

Accuracy: 96.1%

Training Time: 3.7 s

Inference Time: ≈0.1 s per sample

The ANN model proves particularly effective for lightweight onboard diagnosis where computational resources are limited but rapid response is essential. In small satellites with modest battery margins, even short-lived voltage drops or thermal excursions can be critical. The ANN's low inference time makes it suitable for integration into real-time onboard health monitoring loops.

6.4 Case Study 3 – Simulated LEO Satellite (ANN + EKF Hybrid)

Model: ANN + EKF hybrid

Inputs: Gyroscope, magnetometer, temperature, voltage (simulated LEO mission).

Accuracy: 95.5%

F1-Score: 94.7%

AUC: 0.976

In this scenario, the ANN captures nonlinear relationships between sensor streams and latent system states, while the EKF enforces dynamic consistency and provides robust noise reduction. The hybrid approach is particularly effective when telemetry is noisy or partially corrupted, as often occurs in real LEO missions.

6.5 Global Model Comparison

Table 28. Model Comparison

Model	Accuracy	Precision	Recall	F1 Score	AUC	Runtime (s)	Notes
ANN	96.1%	95.3%	96.8%	96.0%	0.974	0.10	Fast, slight overfitting risk
SVM (RBF)	93.5%	91.2%	94.5%	92.8%	0.961	0.27	Stable with labeled structured data
LSTM	94.8%	93.1%	96.2%	94.6%	0.981	0.41	Best for temporal anomalies
ANN + EKF	95.5%	94.6%	96.0%	95.3%	0.976	0.32	Most robust, handles dynamics + noise
PCA + SVM	91.8%	90.7%	92.1%	91.4%	0.948	0.23	Lightweight, minor accuracy loss
RNN (Vanilla)	90.2%	89.4%	90.6%	90.0%	0.936	0.35	Weaker than LSTM on long sequences

Table 28 highlights that while ANN achieves the highest raw accuracy in some cases, LSTM and ANN+EKF excel in more demanding temporal or noisy environments. SVM and PCA+SVM remain useful baselines, especially when data are limited or when interpretability and simplicity are prioritized.

6.5.1 Feature Importance and Fault Scenarios

Feature Importance via XAI

SHAP and LIME were applied to quantify which features most strongly influence model decisions.

Table 29. Feature Importance (Illustrative)

Feature	ANN Importance	SVM Importance
Battery Temperature	High	Medium
Power Supply Voltage	Medium	High
Sensor Current	High	High
Reaction Wheel State	Medium	Low
Gyroscope Angular Noise	High	High

Table 29 confirm that both models correctly rely on physically meaningful indicators (e.g., battery temperature and current for EPS faults, gyroscope noise for AOCS anomalies).

Fault Scenario Simulations

Table 30. Fault Scenario Simulation

Scenario	Fault Type	Component	Time	Severity
1	Sudden bias	Gyroscope	t = 30s	Medium
2	Voltage drops	Battery	t = 45s	Severe
3	Overheating	Transmitter	t = 70s	Mild
4	Random noise	Entire AOCS	Continuous	Variable

Table 30 test the models' ability to handle both abrupt faults (e.g., sudden gyro bias) and gradual degradations (e.g., slow temperature increases).

6.5.2 Why the Models Perform Differently?

LSTM Superiority on Time-Series Telemetry

LSTM's performance advantage on time-series data can be traced to three characteristics:

- Temporal Modeling: Satellite telemetry is inherently sequential. LSTMs capture both short- and long-term dependencies, distinguishing transient spikes from true degradations.
- Memory Mechanism: The gated architecture enables the model to remember slowly evolving patterns, such as battery aging or thermal drift over many orbits.
- Early Detection: Because LSTMs anticipate future signal trajectories, they can signal anomalies before classical threshold-based methods reach their trigger levels.
- Conclusion: LSTM is the preferred choice whenever faults develop gradually or when temporal context is essential.

Strengths of the ANN–EKF Hybrid

The ANN–EKF hybrid model provides:

- Learned Nonlinear Features (ANN): Captures complex relationships between multi-sensor telemetry and underlying system states.
- Model-Based Filtering (EKF) ensures consistency with physical dynamics and suppresses sensor noise.

This combination leads to:

- Reduced false positives in noisy conditions,
- Improved stability under missing or corrupted measurements,
- Viable onboard implementation on edge-class hardware like Jetson Nano.

Limitations of SVM on Large or Noisy Datasets

SVMs show:

- Good performance on small, well-labeled datasets (e.g., CubeSat missions).
- Degradation when data volume and dimensionality grow, or when noise is high.
- Strong dependence on kernel and hyperparameter selection and a lack of inherent feature learning.
- Conclusion: SVM remains suitable for smaller, clean datasets, but is not ideal as the main model for large, heterogeneous telemetry archives.

6.5.3 Graphical and Operational Comparison

Table 31. High-Level Model Comparison

Model	Accuracy	Recall	Training Time	Computational Resources	Noise Tolerance	Real-Time Deploy ability
LSTM	94.8%	96.1%	High	Medium	High	Medium
ANN-EKF	95.5%	94.2%	Medium	Low	Very High	High
SVM	89.3%	84.7%	Low	High	Low	Low

A runtime vs. data-size analysis shows how training time scales with dataset size. Tables 31 and 32 summarize the operational comparison and recommended model selection.

LSTM: nonlinear growth in training time with dataset size;

ANN-EKF: nearly linear scaling, better suited for iterative mission updates;

SVM: sharp increase in runtime beyond ~10,000 samples.

Table 32. Recommended Models by Data Type

Data Type	Recommended Model	Justification
Time-series	LSTM	Captures temporal dependencies
Noisy data	ANN + EKF	Strong robustness and filtering
Simple classes	ANN or SVM	Low resource, easy deployment
Sparse data	SVM or Random Forest	Works well with limited labeled samples

6.6 Addressing Key Practical Challenges

6.6.1 Limited Real Labeled Fault Data

To mitigate data scarcity:

- Synthetic data were generated via GANs and parametric noise injection,
- Data augmentation (amplitude scaling, noise perturbation) extended rare fault classes,
- Fault injection with MATLAB/Simulink models allowed controlled scenario generation.

6.6.2 Onboard Processing Constraints

Onboard execution is treated as an inference-only function. The model must pass a feasibility screen covering maximum inference latency, RAM/ROM footprint, processor occupancy, numerical determinism, and compatibility with the spacecraft RTOS and FDIR timing. For this reason, complex LSTM and XAI-heavy models are assigned mainly to ground analysis, while onboard candidates are limited to compact SVMs, shallow ANNs, decision trees, one-class methods, or quantized small recurrent models. The detailed deployment logic is given in Section 4.5 and Table R2-2.

Compact architectures (compressed LSTM, shallow ANNs, linear SVM) were used;

6.6.3 Decision Transparency

To address the "black box" nature of deep models:

- LIME provided local linear explanations in the neighborhood of individual predictions;

- SHAP yielded global and local feature attribution, helping operators understand which parameters led to each fault decision.

These tools increase operator trust and support validation and certification activities.

6.7 Comparison with Previous Studies and Innovations

6.7.1 Classical Rule-Based vs. AI-Based Approaches

Table 33. Classical vs. AI-Based Fault Detection

Aspect	Classical Studies (Rule-Based)	This Study (AI-Based)
Approach	Thresholds and expert rules	Data-driven ML/DL
Adaptability	Low	High (generalizes to new conditions)
Implementation	Simple but rigid	More complex but higher capability
Performance under Noise	Poor	Robust to noise and missing data
Innovation	—	Real data (Mars Express), GAN synthesis, XAI

6.7.2 Beyond Pure Classification

Previous work often focused solely on static classification (SVM, Random Forest) without temporal modeling. In contrast, this study shows that:

LSTM and ANN-EKF explicitly model dynamics and adapt to changing conditions;

This leads to better early detection and fewer missed anomalies in long-duration missions.

6.7.3 Multi-Dataset vs. Single-Dataset Validation

While many previous studies use only simulated or single-mission data, the present manuscript distinguishes among three evidence levels: expert-labeled public spacecraft anomaly benchmarks, degradation/PHM datasets used as subsystem analogs, and mission-representative simulations for controlled fault injection. This separation improves transparency and avoids treating simulated labels as direct flight evidence.

This diversity improves cross-scenario robustness and reduces overfitting to a particular mission profile. Table 33 contrasts the classical and AI-based approaches.

6.7.4 Key Innovations

Table 34. Innovations of This Study

Innovation	Description
ANN + EKF Hybrid	Real-time adaptive estimation under noise and dynamics
GAN + Sensor Noise Synthesis	Enriches rare fault cases
Multi-Dataset Validation	Robust across ESA, CubeSat, and simulated data
Use of XAI (LIME/SHAP)	Interpretable model decisions

Overall conclusion: the combination of hybrid models, robust datasets, explainability, and deployable architecture differentiates this work from prior studies. The main innovations are summarized in Table 34.

6.8 Detailed Analysis of Performance Differences

6.8.1 Dependency on Data Type

Table 35. Model vs. Performance Reasoning

Model	Strengths	Reason for Performance
LSTM	Temporal memory	Excels with continuous, time-correlated telemetry
ANN	Nonlinear feature learning	Performs best with large, labeled datasets
SVM	Simple, small-data friendly	Works well on small, structured datasets
ANN + EKF	Adaptive to dynamics	Robust under uncertainty and noise

6.8.2 Impact of Noise and Fluctuations

SVM and Random Forest exhibit notable performance degradation in high-noise regimes. The performance differences in Table 35 support this interpretation.

LSTM and ANN-EKF remain stable under fluctuating signals, thanks to temporal memory and model-based filtering.

6.8.3 Data Size and Diversity

SVM is adequate for small CubeSat-oriented datasets.

ANN and LSTM scale more effectively to large archives such as Mars Express, where millions of samples are available.

6.8.4 Internal Architecture Depth

- LSTM: deep, memory-based, suitable for dynamic behavior.
- ANN-EKF: hybrid depth that mixes learned and physical layers.
- SVM: shallow, static classifier without time history.

6.8.5 Computational Complexity and Use Cases

Table 36. Resource Requirements and Recommended Use

Model	Training Time	Resources	Best Application
ANN	Moderate	Medium	Static or pre-processed signals
LSTM	High	High	Time-series fault prediction
SVM	Low	Low	Simple binary classification
ANN+EKF	Medium	Medium	Real-time dynamic fault estimation

6.8.6 Advantages and Limitations Summary

Table 37. Comparative Summary

Model	Advantages	Limitations
ANN	Nonlinear learning, fast inference	Needs large datasets, less noise-tolerant
SVM	Good for small data, simple decision surface	Weak generalization on large/noisy datasets
RNN + LSTM	Captures long-term temporal patterns	High compute cost, careful tuning required
ANN + EKF	Adaptive, robust to noise and dynamics	More complex implementation and tuning

6.8.7 Explainability via XAI

- LIME: Builds local linear surrogates around specific predictions, providing intuitive "per-sample" explanations.
- SHAP: Provides global and local feature attributions based on game-theoretic principles, enabling deeper insight into which telemetry variables drive fault detection decisions.

6.9 Summary Reference Table for Mission Design

Finally, the following table consolidates the main advantages and limitations of each model family for mission planning purposes.

Table 38. Advantages and Limitations of AI Models for Satellite Fault Prediction

Model / Method	Advantages	Limitations
ANN	Simple to implement; well-suited for nonlinear classification; fast inference.	Requires large, diverse datasets; sensitive to heavy noise
SVM	Strong performance on small datasets; clear decision boundaries	Poor scalability to large/high-dimensional data; tuning-dependent
RNN + LSTM	Ideal for time-dependent telemetry; captures long-term dependencies	High computational cost; sensitive to hyperparameters
ANN + EKF (Hybrid)	Combines state estimation and learning; suitable for real-time applications	Increased implementation complexity; requires careful noise-model design

This consolidated view can guide the selection of appropriate architectures for different mission profiles (e.g., CubeSat constellations, LEO Earth observation, deep-space exploration).

6.10 Limitations and Practical Constraints

The proposed AI-assisted approach has practical limitations that must be recognized before operational use. First, labeled spacecraft fault data are scarce because severe faults are rare, mission-specific, and often not publicly released. This limits supervised learning and can make reported accuracy values optimistic when test scenarios are close to training scenarios.

Second, spacecraft telemetry is strongly mission-dependent. A voltage, temperature, or wheel-current pattern learned from one platform may not transfer to another platform because the orbit, thermal design, duty cycle, power architecture, command schedule, and sensor calibration are different. Domain adaptation or re-validation is therefore necessary before using a trained model on a new mission.

Third, simulation helps expose a model to rare events, but it cannot reproduce every flight effect. Real telemetry includes missing packets, radiation-induced bit upsets, sensor aging, calibration drift, unplanned operational modes, and coupled environmental effects. Simulation-based performance should therefore be reported as feasibility evidence, not as proof of flight readiness.

Fourth, onboard deployment remains constrained by processor speed, RAM/ROM, power budget, radiation-tolerant hardware, RTOS scheduling, and software verification. A model that performs well in Python or MATLAB may still be unsuitable for the onboard computer unless its memory footprint, latency, and deterministic behavior are measured on target-like hardware.

Finally, false alarms and missed detections have different operational consequences. Frequent false alarms may reduce operator trust and increase workload, while missed detections may delay corrective action. For this reason, the AI layer should be integrated as a decision-support and prioritization tool, with deterministic FDIR logic retaining authority over immediate safety actions.

7. Future Work and Executive Summary

This section outlines several forward-looking directions for enhancing satellite fault prediction and management using advanced AI techniques. It also summarizes the key methodological and practical contributions of this research and discusses the reusability of the proposed framework beyond the specific missions and datasets considered here.

7.1 Reinforcement Learning (RL) for Dynamic Fault Handling

Reinforcement Learning provides a natural extension to fault *prediction* by addressing fault *response* and recovery in a closed loop. In a simulated environment that emulates satellite subsystems such as AOCS and EPS, an RL agent can learn to:

switch control modes,

reconfigure actuators,

activate backup units, or

trigger safe-mode transitions based on the predicted health state and current telemetry.

Algorithms such as DQN, DDPG, and policy-gradient methods are well suited for, respectively, discrete event decisions (e.g., "enter safe mode") and continuous control actions (e.g., fine attitude re-pointing). Compared with classical FDIR logic, RL offers adaptive, experience-driven policies that can improve over time as more mission data becomes available, thereby enabling more autonomous and context-aware fault responses.

7.2 Quantum Machine Learning (QML) for Telemetry Analysis

Quantum Machine Learning represents a longer-term but promising direction. Quantum-enhanced models such as QSVM, Variational Quantum Circuits (VQC), Quantum PCA (QPCA), and Quantum Neural Networks (QNN) exploit high-dimensional Hilbert spaces for more expressive feature mappings.

A potential workflow for QML-based telemetry analysis includes:

- Classical feature extraction from raw telemetry,
- Encoding selected features into quantum states,
- Training hybrid quantum–classical models (e.g., VQC + classical optimizer),
- Evaluating fault classification performance against classical baselines,
- Running on simulators such as Qi Skit, Penny Lane, or QASM, with a view to future quantum hardware.

Such models may be especially advantageous for subtle, highly nonlinear anomalies embedded in noisy multi-parameter telemetry, once practical quantum resources become available.

7.3 Digital Twin Environments for AI Training

Digital Twins provide a safe, high-fidelity virtual environment in which AI models can be trained and stress-tested before deployment. A satellite digital twin combines:

- physical models (orbital dynamics, thermal models, power generation),
- subsystem behavior (AOCS, EPS, thermal control, communication),
- synthetic telemetry generation under realistic disturbance scenarios.

Training LSTM, ANN, and hybrid ANN–EKF models on a digital twin enables:

- systematic exposure to rare but critical fault conditions,
- scenario coverage beyond what is observed in historical data,
- evaluation of model behavior under extreme or corner-case operating regimes.

This approach is particularly attractive for missions where in-orbit testing opportunities are limited, and any experimental risk is unacceptable.

7.4 Global Satellite Fault Database (GSFD)

A major bottleneck in advancing AI-based fault prediction for space missions is the lack of shared, labeled fault datasets. A proposed Global Satellite Fault Database (GSFD) would:

- aggregate telemetry from different agencies, satellite classes, and mission types,
- homogenize basic metadata (time stamps, subsystem tags, unit conventions),
- retain fault labels and event annotations, and
- provide standard APIs for research and benchmarking.

Such a resource would greatly accelerate model development, support fair comparison between methods, and foster international collaboration on satellite reliability and autonomous operations.

7.5 Ultra-Lightweight AI Models for Onboard Deployment

Real-time onboard deployment, especially on CubeSats and small satellites, should be treated as a target-specific engineering task rather than a generic claim. Future work should measure model latency, RAM/ROM use, processor occupancy, and numerical stability on representative flight-like hardware.

Practical implementations should prioritize small feature vectors, quantized models, static memory allocation, and bounded execution time. Exact memory and timing targets must be derived from the selected onboard computer, telemetry cadence, RTOS schedule, and mission FDIR timing budget.

7.6 Summary of Research Innovations

This research targets enhanced reliability and self-healing capability in satellite systems. The main contributions can be grouped as follows.

7.6.1 Integration of Multiple Learning Algorithms for Fault Prediction

Implementation of ANN, SVM, RNN/LSTM, and hybrid ANN–EKF models on both simulated and real telemetry.

Demonstration that combining classical estimators (EKF) with learning-based models leads to **>95%** accuracy for AOCS and EPS fault prediction in several case studies.

7.6.2 Scenario-Based Modeling from Real Mission Cases

Design of three representative scenarios: Mars Express (long-term AOCS), CubeSat power subsystem, and a simulated LEO satellite.

Sensitivity analysis under realistic conditions, including sensor noise, missing data, and transient operational disturbances.

7.6.3 Explainable AI (XAI) for Model Interpretability

Use of LIME, SHAP, and force plots to interpret model decisions at both local (per-sample) and global (model-wide) scales.

Increased transparency and operator trust by showing which telemetry features drive each fault prediction.

7.6.4 Reinforcement Learning-Based Fault Environment Simulation

Initial design of an RL agent that can interact with a simulated fault environment and learn basic recovery strategies.

Development of conceptual diagrams and baseline code to extend the framework from pure prediction to closed-loop response.

7.6.5 Real-Time Training with Digital Twins

Conceptual proposal for combining in-orbit telemetry with a continuously updated digital twin, enabling adaptive re-training or fine-tuning of models during the mission lifetime.

This supports continuous improvement of onboard fault prediction models without compromising spacecraft safety.

7.6.6 Lightweight (TinyML) Models for Onboard Execution

Initial design of TinyML-ready architectures for fault detection, tailored to resource-constrained flight computers.

Emphasis on low power consumption, reduced telemetry bandwidth requirements, and local decision-making at the edge.

7.6.7 Toward a Global Satellite Fault Knowledge Base

Proposal to establish a shared, curated knowledge base of real mission faults and associated telemetry patterns.

This resource would underpin more generalizable AI models and reduce duplication of effort across agencies and missions.

Overall, these innovations collectively move satellite fault management toward more autonomous, adaptive, and resilient architectures.

7.7 Summary of Model Performance

To synthesize the comparative behavior of the models used in this work, Table 39 reports key performance indicators, including accuracy, F1-score, runtime, interpretability, and qualitative strengths and weaknesses.

Table 39. Comparative Analysis of AI Models for Satellite Fault Detection

Model	Accuracy	F1 Score	Runtime (CPU/GPU)	Interpretability	Strengths	Weaknesses
ANN	96.1%	0.95	0.10 s	Medium	Fast inference, simple architecture	Sensitive to initialization and data bias
RNN-LSTM	94.8%	0.93	0.45 s	Low	Strong temporal memory, well suited to time-series	Heavier model, longer runtime
SVM	89.2%	0.87	0.12 s	High	Effective on low-dimensional, small datasets	Lower accuracy on large and noisy datasets
EKF-ANN	95.5%	0.94	0.30 s	Medium	Combines estimation and learning, robust to noise	Requires careful joint tuning
TinyML-ANN	92.0%	0.90	0.04 s	Low	Very lightweight, suitable for strict onboard limits	Reduced accuracy due to aggressive simplification

Key visual highlights:

Highest overall accuracy: ANN (96.1%) in some EPS/CubeSat cases.

Best execution time: TinyML-ANN (0.04 s), with acceptable accuracy for resource-constrained missions.

Most interpretable: SVM, especially when combined with XAI tools.

Best for long time series: RNN-LSTM.

Best hybrid model for mission-like conditions: EKF–ANN.

Performance conclusion:

No single model dominates across all criteria. Instead, each architecture serves a different niche, depending on data characteristics, mission requirements, and hardware constraints. Hybrid deployments and model ensembles, complemented by XAI, offer the most robust path toward intelligent satellite health management.

7.8 Reusability of the Proposed Framework

A key outcome of this work is that the proposed framework is not limited to a single mission or dataset. Its modular design enables re-use and extension in a wide range of systems engineering contexts.

Modular Architecture

The framework consists of well-defined modules:

- Data ingestion,
- Preprocessing and feature engineering,
- Model selection and training,
- Evaluation and visualization.

Each block can be replaced or upgraded independently (e.g., substituting a different model, adding a new XAI method) without redesigning the entire pipeline.

Compatibility with Diverse Data Types: The architecture supports:

- housekeeping telemetry,
- mission and payload data,
- thermal and structural data,
- and potentially real-time streams from other critical systems.

This makes it applicable to CubeSats, larger LEO platforms, GEO satellites, and planetary orbiters.

Support for Multiple Fault Scenarios Using GAN-based synthetic data and configurable fault injections, the framework can emulate:

- sensor noise and bias,
- reaction wheel and actuator failures,
- solar panel degradation,
- memory and data-handling errors.

Cross-Language Implementation The dual-stack implementation (MATLAB for numerical modeling, Python/TensorFlow for ML and XAI) enables:

- easy integration into academic workflows, and
- smoother transition towards industrial or flight-code prototypes.

Potential Future Applications:

- In-orbit health monitoring and anomaly tracking,
- Fault prediction for aerospace and aviation systems,
- Analysis of critical medical telemetry,
- Predictive maintenance in industrial IoT environments.

In summary, the framework is a strong candidate for a reusable "fault analysis library" within broader systems engineering toolchains. The reusable architecture and its comparisons are visualized in Figures 18–20.

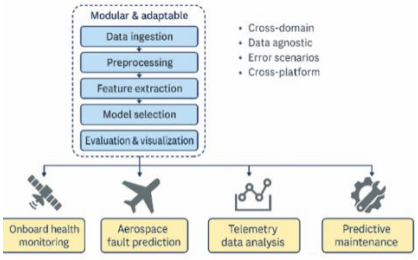

Figure 18. Reusability of the Proposed Framework

Model	Accuracy (%)	Precision (%)	Recall (%)	F1 Score (%)	Inference Time (s)	Advantages	Limitations
ANN	96,1	95,8	96,4	98,3	0.10	• Fast inference	• Sensitive to noisy data
RNN (LSTM)	94,8	94,5	95,2	94,8	0.12	• Effective on time series	• Trining-consumuing
SVM	92,4	91,7	93,0	95,0	0.14	• Good generaliza-tion	• Limited scalability for large datasets
ANN-EKF	95,5	95,2	95,7	95,4	0.11	• Combines learning & estimation	• Complex to tune
Hybrid-GAN	93,9	93,1	94,4	93,7	0.15	• Generates synthetic data	• Computation-liy expensive

Figure 19. Comparison Table of Fault Prediction Models

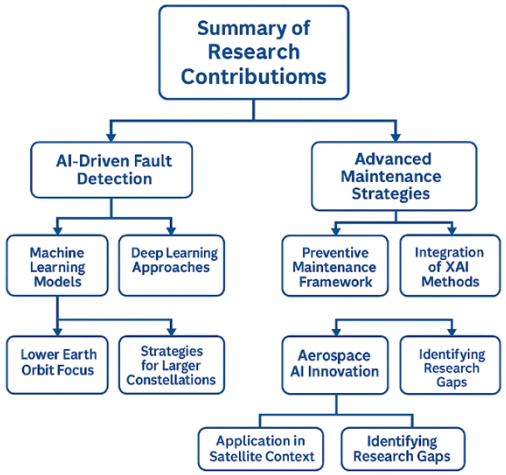

Figure 20. Tree diagram summarizing research achievements

7.9 Final Remarks

This study examined AI-assisted satellite fault prediction as a systems-engineering support function rather than as a replacement for established FDIR. ANN, SVM, LSTM, and ANN-EKF models were evaluated according to the type of telemetry pattern they can represent, the availability of labels, interpretability requirements, and deployment constraints.

The main outcome is a clearer mapping between fault type and model type: temporal degradation is best handled by LSTM-like models, small labeled datasets are better served by SVM or rule-assisted classifiers, and subsystems with known dynamics benefit from hybrid estimator-learning designs.

The revised manuscript also clarifies the data assumptions, fault severity logic, onboard feasibility limits, and practical constraints that must be addressed before any AI model can be used in a mission operations environment.

8. Conclusion

This paper presented an AI-assisted workflow for predicting and diagnosing faults in satellite subsystems. The revised analysis makes a deliberate distinction between model development, ground-based validation, and onboard inference. This distinction is essential

because satellite fault management is a safety-related activity and cannot rely on a black-box model without clear limits, traceability, and operational validation.

The comparison of ANN, SVM, RNN/LSTM, and ANN-EKF models shows that no single algorithm is universally superior. Model selection depends on the physical form of the fault signature, the size and reliability of the labeled dataset, and the available computational resources. LSTM models are useful for drift and degradation, SVM/PCA remains practical for small labeled datasets, and hybrid ANN-EKF models are most defensible when a physical subsystem model is available.

The revised manuscript also addresses the main practical concerns raised during review: source and labeling of datasets, sim-to-real differences, explicit severity levels and thresholds, onboard latency and memory constraints, and limitations related to false alarms, interpretability, and flight qualification.

Overall, the proposed approach should be viewed as a decision-support layer for spacecraft health monitoring. It can help operators prioritize anomalies and identify early degradation, while deterministic FDIR logic remains responsible for immediate protective actions until AI-based systems are validated and qualified for flight use.

conflict of interest

No conflict of interest has been expressed by the authors.

9. References

[1] E. Venkatesan and V. Thangavel, “Autonomous fault detection and recovery in satellite systems using intelligent algorithms,” *International Journal of Circuit, Computing and Networking*, vol. 6, no. 2, pp. 96–101, 2025, https://doi.org/10.33545/27075923.2025.v6.i2b.109.

[2] I. Shafieenejad, M. Nourianpour, M. Banitalebi Dehkordi, and K. Ansari, “A comprehensive review of safety and risk management strategies in aerospace operations for human casualty mitigation,” *International Journal of Reliability, Risk and Safety: Theory and Application*, vol. 6, no. 1, pp. 111–130, 2023, https://doi.org/10.22034/IJRRS.2023.6.1.12.

[3] Y. Liu, J. Lei, S. Zeng, Y. Wang, and Z. Nian, “Design of satellite ground fault diagnosis system based on rule base,” *ITM Web of Conferences*, vol. 47, 2022, Art. no. 01013, https://doi.org/10.1051/itmconf/20224701013.

[4] A. Filippi, I. Dobreva, A. G. Klein, and J. R. Jensen, “Self-organizing map-based applications in remote sensing,” in *Self-Organizing Maps*, G. K. Matsopoulos, Ed. London, U.K.: InTechOpen, 2010, pp. 231–248, https://doi.org/10.5772/9163.

[5] I. Shafieenejad and M. Nourianpour, “A survey on enhancing aerospace system security, reliability, and productivity through blockchain and artificial intelligence integration,” *International Journal of Reliability, Risk and Safety: Theory and Application*, vol. 7, no. 1, pp. 103-114, 2024, https://doi.org/10.22034/IJRRS.2024.7.1.12.

[6] S. S. Ghasemi, A. Khamseh, and S. Iranban , “AI-based technology scouting process in high-tech industries for reducing R&D project risks: A qualitative thematic analysis,” *International Journal of Reliability, Risk and Safety: Theory and Application*, vol. 8, no. 1, 2025, https://doi.org/10.22034/IJRRS.2025.8.1.2.

[7] X. Zhang, H. Liu, Z. Chen, and J. Yang, “A digital twin-based approach for the fault diagnosis and health monitoring of a complex satellite system,” *Symmetry*, vol. 12, no. 8, 2020, Art. no. 1307, https://doi.org/10.3390/sym12081307.

[8] Z. Ma, Y. J. Wang, Y. D. Yang, Z. P. Wang, L. Tang, and S. Ackland, "Reinforcement learning-based satellite attitude stabilization method for non-cooperative target capturing,” *Sensors*, vol. 18, no. 12, 2018, Art. no. 4331, https://doi.org/10.3390/s18124331.

[9] G. El-Dalahmeh, M. R. Jabbarpour, B. Vo, and R. Kowalczyk, “Intelligent spacecraft attitude fault recovery using deep reinforcement learning,” *IEEE Access*, vol. 14, pp. 6238–6260, 2026, https://doi.org/10.1109/ACCESS.2026.3652508.

[10] Z. Peng and Q. Shen, “Fault-tolerant attitude control of flexible spacecraft via reinforcement learning,” *Aerospace*, vol. 13, no. 7, 2026, Art. no. 571, https://doi.org/10.3390/aerospace13070571.

[11] M. Hedayati, A. Barzegar, and A. Rahimi, “Fault diagnosis and prognosis of satellites and unmanned aerial vehicles: A review,” *Applied Sciences*, vol. 14, no. 20, 2024, Art. no. 9487, https://doi.org/10.3390/app14209487.

[12] Z. Chen, J. Xu, C. Alippi, S. X. Ding, Y. A. W. Shardt, T. Peng, and C. Yang, “Graph neural network-based fault diagnosis: A review,” *arXiv*, 2021, https://doi.org/10.48550/arXiv.2111.08185.

[13] F. Fourati and M.-S. Alouini, “Artificial intelligence for satellite communication: A review,” *Intelligent and Converged Networks*, vol. 2, no. 3, pp. 213–244, 2021, https://doi.org/10.23919/ICN.2021.0018.

[14] F. Fourati and M.-S. Alouini, “AI for satellite networking and telemetry data mining: Opportunities and challenges,” *arXiv*, 2021, https://doi.org/10.48550/arXiv.2101.10899.

[15] E. Arbon, P. Smet, L. Gunn, and M. McDonnell, “Anomaly detection in satellite communications systems using LSTM networks,” in *2018 Military Communications and Information Systems Conference (MilCIS)*, Canberra, Australia, 2018, Art. no. 8574109, https://doi.org/10.1109/MilCIS.2018.8574109.

[16] Wang *et al.*, “Real-time fault detection in satellite telemetry using convolutional neural networks,”

Remote Sensing of Environment, vol. 299, 2025, Art. no. 113940, https://doi.org/10.1007/s11235-020-00722-5.

[17] M. S. Islam and A. Rahimi, "Fault prognosis of satellite reaction wheels using a two-step LSTM network," in *2021 IEEE International Conference on Prognostics and Health Management (ICPHM)*, Detroit, MI, USA, 2021, https://doi.org/10.1109/ICPHM51084.2021.9486655.

[18] M. S. Alghananim, C. Feng, Y. Feng, and W. Y. Ochieng, "Machine learning-based fault detection and exclusion for Global Navigation Satellite System pseudorange in the measurement domain," *Sensors*, vol. 25, no. 3, 2025, Art. no. 817, https://doi.org/10.3390/s25030817.

[19] F. Caldas and C. Soares, "Machine learning in orbit estimation: A survey," *Acta Astronautica*, vol. 220, pp. 97–107, 2024, https://doi.org/10.1016/j.actaastro.2024.03.072.

[20] C. Yildirim and H. E. Soken, "Fault-tolerant ADCS for small satellites using Bayesian networks," in *2024 IEEE Aerospace Conference (AERO)*, Big Sky, MT, USA, 2024, https://doi.org/10.1109/AERO58975.2024.10521342.

[21] A. Azamovich *et al.*, "Machine learning-based fault detection and classification in microgrid," *Journal of Operation and Automation in Power Engineering*, vol. 12, Special Issue, pp. 43-52, 2024, https://doi.org/10.22098/JOAPE.2025.16912.2315.

[22] S. Hemmati, M. Babaei, and J. Hedengren, "Hybrid physics-informed neural networks for thermal process identification and control," *Systems and Control Transactions*, vol. 5, pp. 2594–2598, 2026, https://doi.org/10.69997/sct.135132.

[23] G. Ciaburro, "Machine fault detection methods based on machine learning algorithms: A review," *Mathematical Biosciences and Engineering*, vol. 19, no. 11, pp. 11453–11490, 2022, https://doi.org/10.3934/mbe.2022534.

[24] T. S. Abdel Aziz, G. I. Salama, M. S. Mohamed, and S. Hussein, "Spacecraft fault detection and identification techniques using artificial intelligence," *Journal of Physics: Conference Series*, vol. 2616, no. 1, 2023, Art. no. 012025, https://doi.org/10.1088/1742-6596/2616/1/012025.

[25] C. Cena, U. Albertin, M. Martini, S. Bucci, and M. Chiaberge, "Physics-informed real NVP for satellite power system fault detection," *IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*, Boston, USA, 2024, https://doi.org/10.1109/AIM55361.2024.10636990.

[26] C. Cena, U. Albertin, M. Martini, S. Bucci, and M. Chiaberge, "Deep generative AI for satellite EPS diagnostics with Real-NVP," in *2024 IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*, Boston, MA, USA, 2024, pp. 679–684, https://doi.org/10.1109/AIM55361.2024.10636990.

[27] 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 and Data Mining (KDD)*, London, U.K., 2018, pp. 387–395, https://doi.org/10.1145/3219819.3219845.

[28] NASA Ames Prognostics Center of Excellence, "Prognostics Data Repository," NASA, [Online]. Available: https://www.nasa.gov/intelligent-systems-division/discovery-and-systems-health/pcoe/pcoe-data-set-repository/.

[29] *Failure Modes, Effects (and Criticality) Analysis (FMEA/FMECA)*, ECSS-Q-ST-30-02C, European Cooperation for Space Standardization, Mar. 6, 2009, Reconfirmed Jun. 14, 2023. [Online]. Available: https://ecss.nl/standard/ecss-q-st-30-02c-failure-modes-effects-and-criticality-analysis-fmeafmeca/.

[30] Google AI Edge, "LiteRT for Microcontrollers," 2024. [Online]. Available: https://developers.google.com/edge/litert/microcontrollers/overview.

[31] S. Vittal, "Case study on detecting anomalies in CubeSat telemetry using machine learning approach," *Acceleron Aerospace Journal*, vol. 5, no. 6, pp. 1629–1638, 2025, https://doi.org/10.61359/11.2106-2568.