Real-time Sensor Data Streaming for Deployment in Edge AI for Health Index Construction and Remaining Useful Life Prediction

Salama Almheiri[1], Zahi Mohamed [1], Mohamed Ragab [1], Abdulla Alseiari [1], Nouf Almesafri [1]

[1]*Technology Innovation Institute, Abu Dhabi, Masdar City, United Arab Emirates*

Salama.almheiri@tii.ae
Zahi.mohamed@tii.ae
Mohamed.adam@tii.ae
Abdulla.alseiari@tii.ae
Nouf.almesafri@tii.ae

ABSTRACT

This paper introduces a real-time predictive analytics framework that integrates edge artificial intelligence for remaining useful life estimation and health index construction using turbofan engine sensor data. A MATLAB/Simulink model was designed to stream 14 critical sensor signals, derived from the NASA C-MAPSS dataset, into an Opal-RT OP5707XG simulator for real-time emulation. These signals were output as analog voltages, converted into digital values using ADS1115 converters, and processed on an Nvidia Jetson AGX Orin edge-computing platform. A CatBoost regressor, trained on a feature-rich time-series dataset and refined through SHapley Additive Explanations-based feature selection was employed as the predictive model. System performance was benchmarked on two hardware platforms: a mid-tier desktop computer and the Jetson AGX Orin. The mid-tier desktop computer completed training in 18 minutes, while the Jetson required around 3 hours. Inference speed was also faster on the computer at 2.8 ms versus 7.5 ms, though both satisfied the 33 ms requirement for real-time processing of a 30 Hz data stream. The Jetson demonstrated a significant efficiency advantage, consuming 20—40 W compared to 250-350 W for the computer. The framework achieved high accuracy with strong generalization and transparent explainability through SHapley Additive Explanations-based feature selection confirming the feasibility of deploying advanced prognostics on edge AI hardware for real-time health monitoring.

1. INTRODUCTION

The adoption of Artificial Intelligence (AI) in diagnostics, prognostics, and health management of assets has become pivotal in advancing predictive maintenance solutions. Additionally, AI and Machine Learning (ML) algorithms are now being embedded in hardware devices to allow for real-time monitoring while simultaneously generating important information that aids proactive maintenance and optimizes decision making capabilities. Furthermore, one of the primary goals of the Fourth Industrial Revolution is to implement predictive maintenance and condition monitoring in Edge AI devices, which are typically located close to the end devices and carry analytical capabilities and huge processing power (Sharanya et al., 2022).

Research aimed at diagnosing and identifying failures in industrial facilities has primarily been carried out in the 1980s, and it has mainly been conducted based on mathematical and physics-based principles (Park et al., 2018). In predictive maintenance and condition-based maintenance, sensor data was typically collected and transmitted for analysis. Therefore, the transmission of the sensor data was done either through wires or the web. These extended transmission routes resulted in specific issues such as irrelevant information, delays in transmission, and dispatching inaccurate or insufficient data (Bala et al., 2024).

To address these constraints and limitations, edge computing devices were introduced for initial data preparation and real-time decision making which eliminates the need to transmit large quantities of raw data to the cloud, leading to an enhanced overall efficiency of the predictive maintenance framework (Hector & Panjanathan, 2024).

Building on this foundation, prior work has established the evolution of prognostics and health management (PHM) techniques from classical physics-based models to modern data-driven and hybrid approaches. (An et al., 2013) provided an early taxonomy distinguishing model-based, data-driven, and hybrid methods, highlighting their respective advantages and shortcomings. More recently, (Su & Lee, 2024) compiled a comprehensive review of machine learning methods used

in diagnostics and prognostics across multiple PHM data challenges, offering practical insights into feature extraction, prediction accuracy, and benchmark limitations. Together, these works illustrate both the historical depth and ongoing diversification of PHM strategies.

Parallel to advances in algorithms, edge computing has emerged as a transformative enabler for predictive maintenance. By shifting computation closer to the sensor, edge devices overcome the latency and bandwidth limitations of cloud-based approaches. (Artiushenko et al., 2024) demonstrates how resource-efficient Edge AI implementations can accelerate integration into maintenance workflows by providing rapid response times, scalability benefits, and improved data security.

Another line of research has focused on feature engineering and explainability in prognostics. Interpretable models are critical for trust, particularly in aerospace and safety-critical domains. Studies applying SHapley Additive Explanations (SHAP) to RUL prediction confirm that combining optimized preprocessing with explainability can reveal degradation drivers more transparently (Ndao et al., 2025). Similarly, (Khandekar et al., 2024) illustrate how SHAP-based feature attribution in predictive maintenance frameworks strengthens both accuracy and interpretability, thereby enhancing operational trustworthiness.

Finally, several works highlight the challenges of hardware–software integration when deploying PHM models on embedded platforms. Field-tested edge monitoring systems validated via hardware-in-the-loop (HIL) approaches illustrate the complexities of real-time data streaming, synchronization, and analog–digital conversion (Short & Twiddle, 2019). This approach specifically addresses the challenge of limited adaptability and accuracy in existing fault detection frameworks. Recent work on microcontroller-based deployments further emphasizes the resource constraints of embedded devices, showing how techniques such as pruning and quantization are essential for achieving efficient yet accurate prognostic performance in constrained hardware environments (Pandey et al., 2023). While Short & Twiddle focused on model-based fault isolation and Pandey et al. emphasized data-driven diagnostics, both approaches faced difficulties in handling dynamic operating conditions. The current work improves on these by integrating a hybrid architecture that combines data-driven learning with adaptive thresholding, thereby enhancing robustness and real-time performance.

This study introduces a real-time localized predictive analytics tool using simulated sensor signals and incorporating Edge AI functionalities for on-site data processing. The proposed setup in the paper acts as a foundation for developing a predictive analytics tool that employs real-time data from sensors in test rigs compared to prior HIL and Jetson-based studies, the novelty lies in a fully analog end-to-end chain (Opal-RT AO → ADC → Jetson) with synchronized timing measurements, real-time, explainable HI construction on-device using CatBoost feature importances, and a deployment-oriented characterization that couples latency and power to an explicit 30 Hz budget.

2. MATERIALS AND METHODS

The study conducted in this paper proposes a system architecture that includes a MATLAB/Simulink model designed to stream sensor signals which are connected to several elements of a turbofan engine. The model is then configured to be connected to Opal-RT for real-time streaming of the sensor signals through its analog output ports. Furthermore, the deployment of Edge AI in this architecture is demonstrated by connecting the sensor signals to the Nvidia AGX Jetson development kit, where a sophisticated Remaining Useful Life (RUL) prediction model is deployed on the edge AI device for real-time sensor data analysis. The primary focus of the paper is the characterization of the performance of the hardware setup and the efficiency of the data streaming pipeline, rather than an exhaustive exploration of the machine learning model itself. The proposed system architecture is presented in Figure 1.

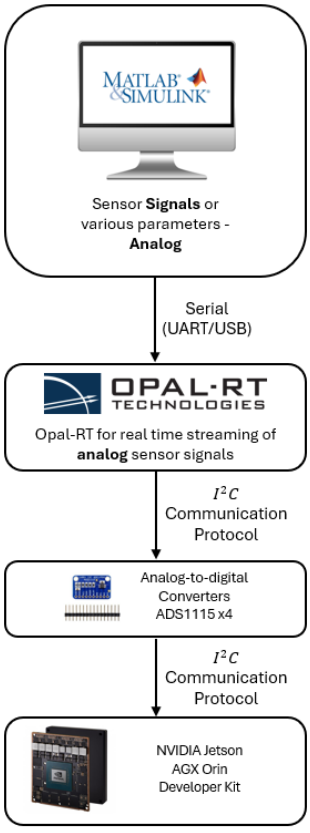

Figure 1. System architecture of the proposed predictive analytics tool using real-time simulation and Edge-AI.

2.1. MATLAB/ Simulink Model

MATLAB/Simulink was used to build a model that streams 14 sensor signals attached to various elements of a turbofan engine. Each sensor measured critical parameters such as temperature, speed, and pressure across the compressor, turbine, and combustor. These signals are used to understand the behavior of the turbofan engine and diagnose it accurately in different operating conditions. The dataset which was used to stream the specified sensor signals was obtained from the Commercial Modular Aero-Propulsion System Simulation data (C-MAPSS) developed by the National Aeronautics and Space Administration (NASA) (Hong et al., 2020), Specifically, the FD001 subset of the C-MAPSS dataset was utilized for this study. This subset represents a fleet of simulated turbofan engines operating under a single steady operating condition and subject to one type of degradation fault, namely the high-pressure compressor (HPC) fault. FD001 serves as the most fundamental configuration among the four C-MAPSS subsets, making it ideal for initial model validation and baseline prognostics development. It provides 100 training and 100 testing engine trajectories, each consisting of multivariate time-series data that capture the gradual degradation process until engine failure.

For modelling purposes, out of the 21 available sensor signals, 14 were selected to be streamed into the model. These sensors were chosen based on their relevance, variability, and contribution to fault progression, while redundant or constant channels were excluded. The selected sensors are listed below:

1. Temperature at Low Pressure Compressor (LPC) outlet
2. Temperature at High Pressure Compressor (HPC) outlet
3. Temperature at Low Pressure Turbine (LPT) outlet
4. Pressure at High Pressure Compressor (HPC) outlet
5. Physical fan speed
6. Physical core speed
7. Static pressure at High Pressure Compressor outlet
8. Ratio of fuel flow to Ps30
9. Corrected fan speed
10. Corrected core speed
11. Bypass ratio
12. Bleed enthalpy
13. High Pressure Turbine (HPT) coolant bleed
14. Low Pressure Turbine coolant bleed

Considering the system architecture and compatibility, the sensor signals streamed through Simulink are represented as analog voltages which needed to be normalized to between 0V to 3.3V in order to be compatible with the voltage rating of the Analog-to-Digital Converters (ADC) and the Edge AI device in use. For this to be implemented, the MATLAB code was developed to load the dataset, identify the 14 sensors of choice, and remove any rows with missing data. As for the normalization of the data, the range was calculated for each sensor column, and the largest range was selected as the reference column, which was then used to scale all other sensor columns to the same voltage range as the Voltage at the Collector (VCC) of the ADCs and the Nvidia Jetson AGX kit. The Simulink model is presented in Figure 2.

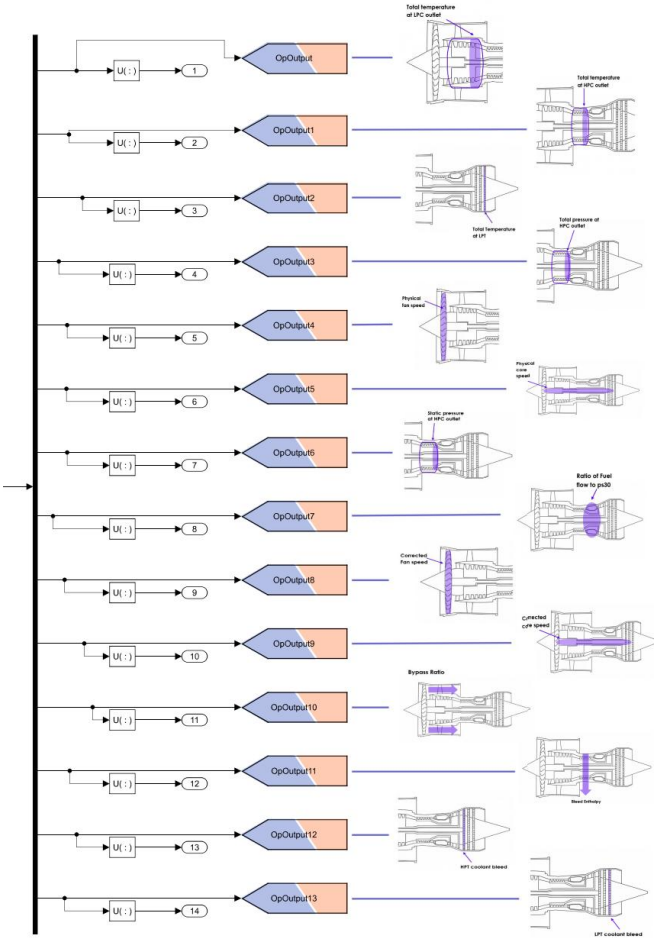

Figure 2. Simulink model of the 14 sensor signals of different elements of the turbofan engine.

2.2. Opal-RT Real-time Simulation

Real-time simulation was configured through Opal-RT to enable real-time streaming of sensor signals. The Opal-RT module used is OP5707XG, a high-end simulator offered by Opal-RT. The model comprised two systems; a Computation system labeled SM_Computation, which includes the deterministic, real-time simulation of the turbofan engine sensor signals, and it is the model that was built in MATLAB/Simulink. And a Graphical User Interface (GUI) system labeled SC_GUI, which is responsible for the monitoring, model design, and Input/Output (I/O) setup. Once the model was built, the Opal-RT board had to be

configured, and that was done by assigning each signal to its corresponding analog output channel. For all 14 sensor signals, the corresponding analog output channels A0 to A13 were utilized, and these channels represent the physical voltage output ports on the simulator that will later on be used to connect the analog sensor signals to the ADCs in use. The analog output channels of the Opal-RT simulator and the physical connection of the 14 channels is shown in Figure 3.

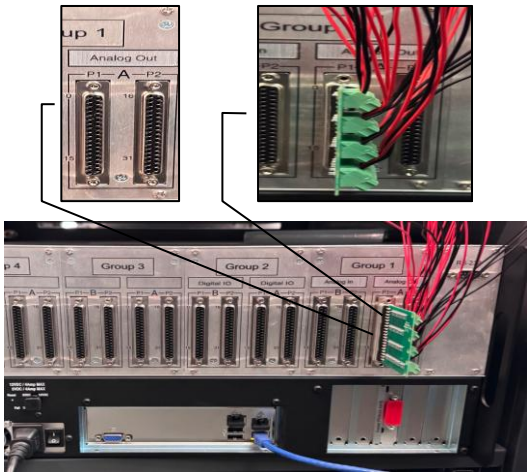

Figure 3. Opal-RT Analog Out channel and physical connection of 14 wires for the sensor signals.

2.3. Analog-to-Digital Converters

The Edge-AI device in the proposed system is the Nvidia Jetson AGX development kit, however, it is incapable of processing analog input signals, therefore, several ADCs had to be connected in order to convert the analog sensor signals to digital values that can be processed and used by the Edge-AI device. In order to proceed with the connection of the ADCs to the Jetson AGX kit, several communication protocols were considered and evaluated based on the compatibility of all components in the system, and it was decided that the Inter-Integrated Circuit (I^2C) protocol will be selected due to its simplicity and adaptability with all components in the system.

The ADC of choice was the ADS1115 module, a 16-bit analog to digital converter with an I^2C interface and a voltage range of 2V to 5V. The selection of the ADS1115 was motivated by its ease of integration, as it employs the simple and widely supported I^2C communication protocol, unlike more complex ADC alternatives. Each module carries 4 analog channels (A0-A3), meaning that the 14 analog signals representing the output of opal-RT required the use of 4 ADS1115 modules in total, allowing for a total number of 16 channels to be used. Furthermore, the VCC pins of all modules were connected to a power supply of 3.3V, the Ground (GND) pins of all 4 modules were tied to a common ground, the Serial Data Line (SDA) pins of all 4 modules were connected together, and the Serial Clock Line (SCL) pins were also connected together.

As for the Address (ADDR) pin of all four modules, the ADDR pins of the ADCs were connected to VCC, GND, SDA, and SCL respectively. This enabled the unique address allocation for each module; 0x48, 0x49, 0x4A, and 0x4B accordingly. The four ADC modules were connected as illustrated in Figure 4.

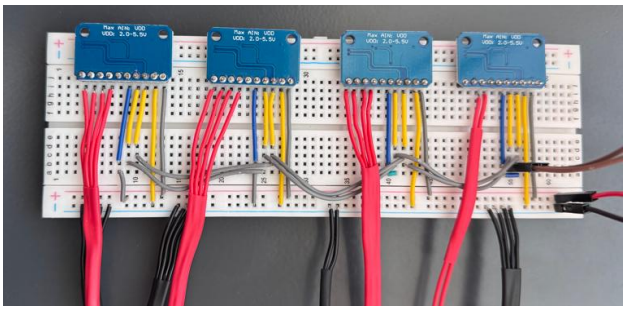

Figure 4. Connection of the four ADS1115 modules and Opal-RT analog output channels.

As for the 14 analog output signals being streamed in real-time from Opal-RT, each analog signal consisted of a positive channel (red wires) and a negative channel (black wires), the positive channel of each signal was connected to an analog channel on the ADC module, and the negative channel of each analog signal was connected to the common ground in the setup. The wire configuration of each pin is illustrated in Table 1.

Pin	Connection
VDD	3.3 V (shared)
GND	Common ground
SDA	Shared SDA line
SCL	Shared SCL line
ADDR	VDD, GND, SDA, SCL 0x48, 0x49, 0x4A, 0x4B
A0–A3	OPAL-RT analog outputs (A0-A13)

Table 1. Pin configuration and connection of ADCs and Opal-RT analog signals.

2.4. Nvidia Jetson AGX Setup

The NVIDIA Jetson AGX Orin Developer Kit was deployed as the edge-computing platform for executing machine learning inference in real time. For the hardware connection of the NVIDIA Jetson AGX developer kit and the four-module ADC configuration. A 3.3V power supply was taken from the NVIDIA Jetson to power the entire system. Furthermore, the SDA, SCL, and GND pins were all tied to

the corresponding pins in the ADC modules. The connection of both components is illustrated in Figure 5.

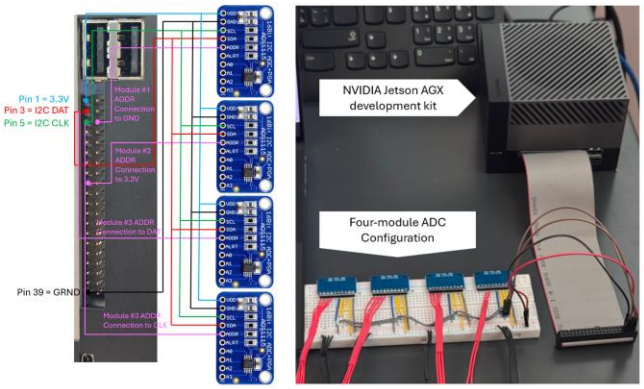

Figure 5. Connection of the four-module ADC configuration and the NVIDIA Jetson AGX development kit.

The device was configured with JetPack Software Development Kit (SDK), which is Ubuntu-based, and includes Compute Unified Device Architecture (CUDA), the Nvidia Deep Neural Network (cuDNN), TensorRT, and relevant Python dependencies. Data acquisition was performed via multiple ADS1115 16-bit ADC converters connected over the I²C bus, supporting up to twelve input channels across three modules, with the fourth module restricted to channels A0 and A1. All ADCs operated in differential mode where applicable, enabling noise rejection in thermocouple-like connections.

The Jetson ingested the digitized sensor data at a fixed sampling rate determined by Opal-RT output frequency. The incoming signal stream underwent synchronous buffering to preserve temporal alignment across channels. Preprocessing routines execute on-device, including z-score normalization, moving average smoothing, and min–max scaling where model-specific requirements dictate.

A dedicated Python pipeline was implemented for feature engineering, leveraging both statistical descriptors (mean, variance, kurtosis, skewness) and temporal descriptors (rolling window slope, first derivative, spectral entropy). The pipeline maintained compatibility with both regression and classification architectures.

To support on-device explainability, SHAP values were computed for model predictions where latency budgets allow (Nohara et al., 2022), primarily for offline model validation. Feature importance scores are cached and employed in the dynamic construction of Health Indices, computed as a weighted dot product between normalized feature vectors and their corresponding importance coefficients. This enabled a univariate health trajectory representation from multivariate sensor data in real time.

The Jetson operates in a fully standalone mode, without dependency on an external MATLAB/Simulink host. The only upstream requirement is the continuous feed of analog sensor signals from OPAL-RT, which emulates the turbofan engine environment. A local Python-based GUI, optimized for the Jetson environment, provides real-time visualization of sensor values, health indices, RUL estimates, and classification outcomes.

2.5. Edge-AI Machine Learning

Machine learning inference on the Jetson AGX Orin comprises both regression and classification pipelines, optimized for low-latency execution. Models are deployed in TensorRT-optimized form where applicable to leverage GPU acceleration. This study uses the FD001 subset of C-MAPSS exclusively. To prevent temporal leakage, we split **by** engine unit (grouped) and preserve chronology (no shuffling). Sliding windows are constructed over each engine's time series using past-only context; specifically, features are computed from windows $W \in \{15,30,45\}$ with stride $s = 1$on standardized signals, and no future information is used in any feature or target formation. Model selection used grouped cross-validation (GroupKFold by engine), and the final test set is a chronological hold-out of engines unseen during training. As the manuscript's scope is the hardware-streaming pipeline, a full exploration of windowing strategies, split variants, and model families is deferred to future work.

2.5.1. Regression Model

For RUL prediction, two model categories were employed:

1. Ensemble Gradient Boosting Regressors (e.g., CatBoostRegressor, XGBoost, LightGBM), which were selected for their ability to handle non-linear feature interactions and heterogeneous data distributions. CatBoost was particularly leveraged for its native handling of categorical features without explicit one-hot encoding, thereby avoiding the curse of dimensionality. Categorical features (e.g., discretized operational modes, maintenance phase flags) were embedded internally via CatBoost's target statistics mechanism.
2. Similarity-based Models, where partial degradation trajectories from the current engine state are matched to a library of historical run-to-failure curves. Matching is performed via distance metrics (e.g., Dynamic Time Warping, cosine similarity) over normalized health index space, returning the RUL associated with the most similar historical patterns.

Hyperparameters for all regression models were tuned via wights ω, Bayesian optimization, searching over learning rate, maximum tree depth, L2 regularization coefficient λ, subsample ratio, and minimum child weight. The L2 penalty is:

$$\mathcal{R}_{\mathcal{L}2}(\omega) = \lambda \sum_{i=1}^{n} \omega_i^2 \quad (1)$$

The optimization objective minimized cross-validated RMSE while maintaining inference latency constraints (<50 ms per prediction). The regression output includes the predicted RUL in operational seconds, accompanied by a confidence envelope derived from bootstrap resampling of the prediction residuals.

2.5.2. Classification Model

The classification pipeline identifies the current operational state and fault class of the turbofan engine. Models tested include:

1. CatBoostClassifier (primary), leveraging identical categorical feature handling to the regression pipeline.
2. Gradient Boosting Decision Trees (GBDT) and Support Vector Machines (SVM) for secondary benchmarks.

A lightweight feedforward neural network for multi-class classification, trained offline and quantized for deployment. The classifier operates on the same engineered feature set as the regression model but includes additional binary indicators (e.g., threshold exceedances, fault mode activation) and categorical mode descriptors. Hyperparameter tuning followed the same Bayesian optimization procedure, minimizing log loss while controlling for overfitting via early stopping and L2 regularization. Model outputs include predicted fault class and associated probability scores.

2.5.3. Health Index Integration

Both regression and classification pipelines utilize the Health Index (HI) time series as the primary prognostic indicator. The HI is computed on the Jetson platform in real time through a CatBoost-driven weighted feature aggregation mechanism. For each feature $x_i(t)$, the corresponding importance weight is obtained from the CatBoost model's feature importance vector FI_i, normalized as:

$$\pi_i = \frac{FI_i}{\sum_{j=1}^{m} FI_j} \ , i = 1, \dots, m \quad (2)$$

The normalized feature vector $\tilde{x}_i(t)$ is then constructed using a standard z-score transformation:

$$\widetilde{x_i}(t) = \frac{x_i(t) - \mu_i}{\sigma_i} \quad (3)$$

where μ_i and σ_i denote the mean and standard deviation of feature i, respectively. The instantaneous Health Index is subsequently derived as a logistic aggregation of the normalized features:

$$\mathcal{H}(\mathrm{t}) = \sigma\left(\sum_{i=1}^{m} \pi_i \, \widetilde{x_i}(t)\right), \sigma(z) = \frac{1}{1 + e^{-z}} \quad (4)$$

which bounds the index within [0,1] for interpretability and consistent scaling across operating regimes. To suppress short-term noise and enhance temporal stability, an exponential moving average is applied, yielding the smoothed health trajectory:

$$\mathcal{H}(\mathrm{t}) = (1 - \alpha)\mathcal{H}(\mathrm{t} - 1) + \alpha\mathcal{H}(\mathrm{t}), \alpha \in (0,1) \quad (5)$$

This formulation ensures that the Health Index remains model-driven, explainable, and adaptive to varying engine operating conditions, providing a stable foundation for both regression-based RUL estimation and classification-based anomaly detection.er5

3. RESULTS AND DISCUSSION

To validate the proposed framework for real-time RUL prediction, a robust machine learning model was developed and assessed on two different hardware platforms: a typical mid-range desktop PC for baseline performance, and an Nvidia Jetson AGX Orin serving as the intended edge-AI deployment system.

A state-of-the-art CatBoost Regressor was employed to predict RUL. The model was trained on a feature-rich dataset derived from the raw sensor signals using a multi-window time-series analysis to capture temporal degradation patterns. An initial model was used with SHAP to perform recursive feature selection, ensuring that only the most impactful features were used for the final model.

The final model demonstrated exceptional performance on the test dataset, achieving an R^2 Score of 0.9957, a Root Mean Squared Error (RMSE) of 4.48 cycles, and a Mean Absolute Error (MAE) of 2.85 cycles. The following sections provide a detailed discussion of these results.

3.1 Experimental Hardware Setup

The performance of the predictive model was evaluated on two distinct hardware platforms: a standard mid-tier desktop Personal Computer (PC) for baseline performance, and an Nvidia Jetson AGX Orin as the target edge AI deployment system. The full system setup is demonstrated in Figure 6.

The mid-tier PC setup featured a 12th Gen Intel Core i7-12700F processor with 12 cores and 20 threads, paired with an NVIDIA GeForce RTX 3060 GPU containing 12 GB GDDR6 memory and delivering approximately 13 TFLOPS of FP16 performance. The system included 32 GB of DDR4 RAM to support the computational workload.

The Nvidia Jetson AGX Orin development kit represented the edge AI platform, equipped with a 12-core Arm Cortex-A78AE v8.2 64-bit CPU and a 2048-core NVIDIA Ampere architecture GPU with 64 Tensor Cores capable of up to 275 TOPS INT8 performance. The device included 32 GB of 256-bit LPDDR5 memory optimized for edge computing applications.

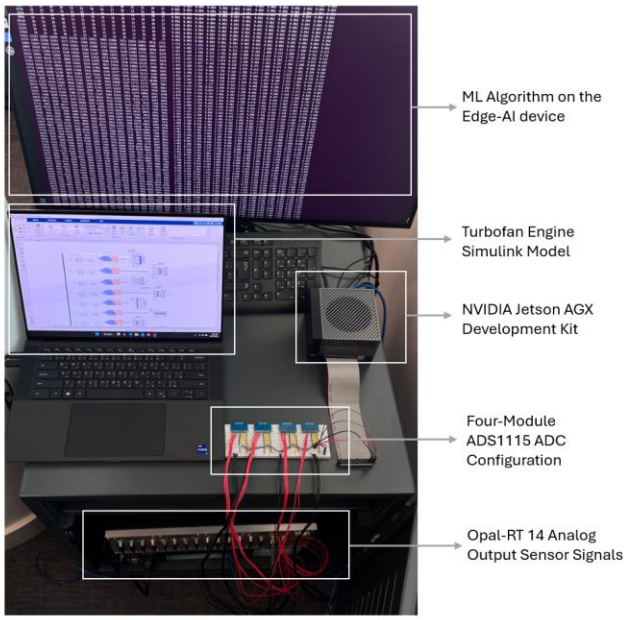

Figure 6. Full system setup.

3.2 Data Streaming and Performance Evaluation

For the real-time evaluation, a data pipeline was designed to stream sensor readings to the inference engine at a continuous rate of 30 rows per second. The primary metrics for comparison were model training time, single-pass inference speed, and power consumption under load.

This study considered; analog out (Opal-RT) → ADC (ADS1115, I^2C) → Jetson; direct digital via DAQ (SPI/I^2C multiplexers); and network streaming (UDP/TCP from the simulator). Analog + ADS1115 offers simplicity, signal-chain visibility, and low integration effort at the cost of extra conversion latency; DAQ-based digital links reduce conversion steps but increase integration complexity and cost; Ethernet streaming offers high throughput and flexible topology but requires timestamp synchronization and packet-loss handling. The selection prioritized repeatable timing, ease of HIL wiring, and low power, aligning with edge deployment constraints.

The real-time budget is set by the stream frequency: for 30 Hz, $T = 1/30 = 33.3$ms per sample. End-to-end inference must finish within this interval (including buffering and I/O) to maintain real-time operation.

Performance evaluation showed distinct trade-offs between the platforms. The mid-tier PC demonstrated superior training performance, completing the model training and optimization process in approximately 18 minutes, while the Jetson AGX Orin was estimated to require 2.5 to 3 hours for the same training task due to its lower computational throughput.

For inference performance (mean ± std over N = 10,000 predictions after a 1,000-step warm-up), the PC achieved an average inference time of approximately 2.8 ms per prediction, while the Jetson AGX Orin delivered inference times of approximately 7.5 ms. Both systems met the real-time requirements for the 30 Hz data stream, as each inference completed well within the 33 ms time budget between incoming data points.

The most significant difference emerged in power consumption. The mid-tier PC consumed between 250-350 Watts under computational load, while the Jetson AGX Orin operated at only 20-40 Watts. This substantial difference in power consumption makes the Jetson particularly suitable for edge deployment scenarios where power efficiency is critical, despite its slower individual inference times.

3.3 Predictive Accuracy and Model Validation

The primary measure of the model's success is its predictive accuracy, as illustrated in Figure 7. The plot of predicted versus actual RUL values shows a near-perfect correlation, with the data points forming a tight, linear cluster along the "Perfect Prediction" line. The exceptionally high R^2 score confirms that the model can explain over 99.5% of the variance in the RUL, indicating a highly reliable and precise predictive capability.

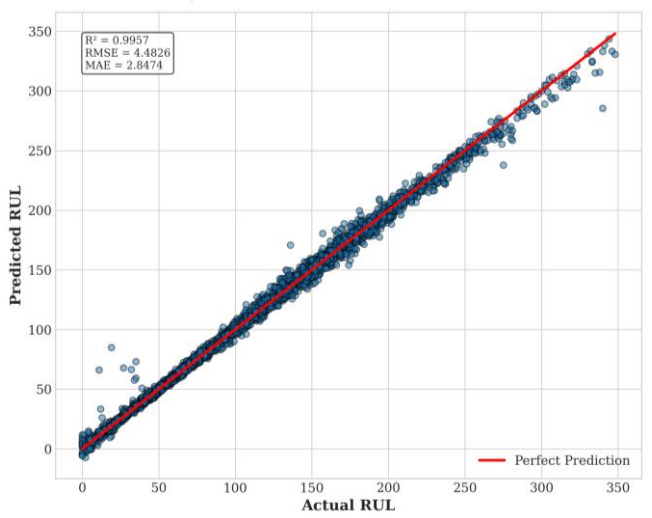

Figure 7. Predicted vs. Actual RUL.

To ensure the model is not systematically biased, a residual analysis was conducted. Figure 8 plots the residual errors against the predicted RUL. The errors are randomly distributed around the zero-error line, showing no discernible patterns. This homoscedastic distribution is a strong indicator of a well-calibrated model, confirming that the prediction error is independent of the magnitude of the RUL.

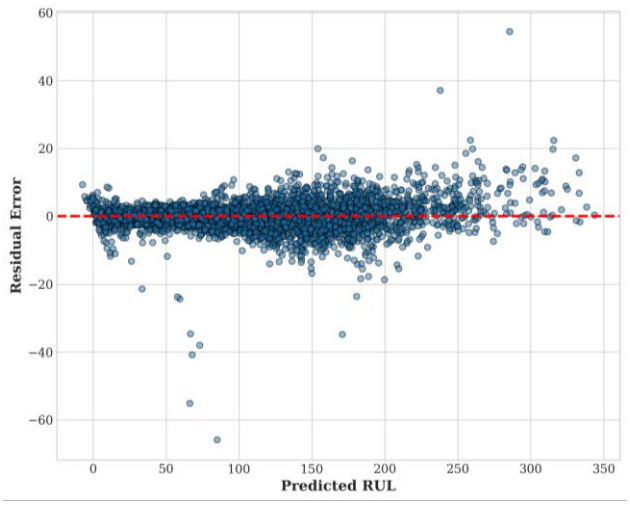

Figure 8. Residuals vs. predicted values.

Further validating this, Figure 9 shows that the distribution of these errors is tightly centered around a near-zero mean (-0.08 cycles), resembling a normal distribution. This confirms the absence of systemic bias and shows that large prediction errors are rare.

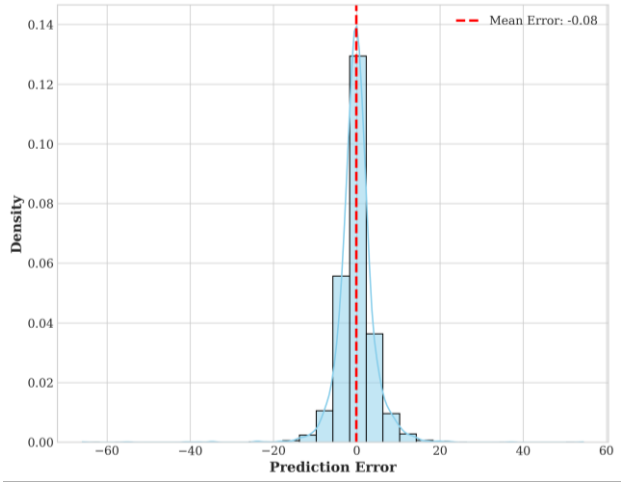

Figure 9. Distribution of prediction errors.

3.4 Model Robustness and Explainability

The model's ability to generalize to unseen data was assessed using a learning curve, presented in Figure 10. The plot shows that the cross-validation score (green line) converges closely with the training score (red line) as more training examples are used. This convergence to a low error value, with a minimal gap between the two curves, demonstrates that the model has achieved an excellent bias-variance trade-off and is not overfitting to the training data. This is a crucial finding, suggesting that the model is robust and its high performance is reliable.

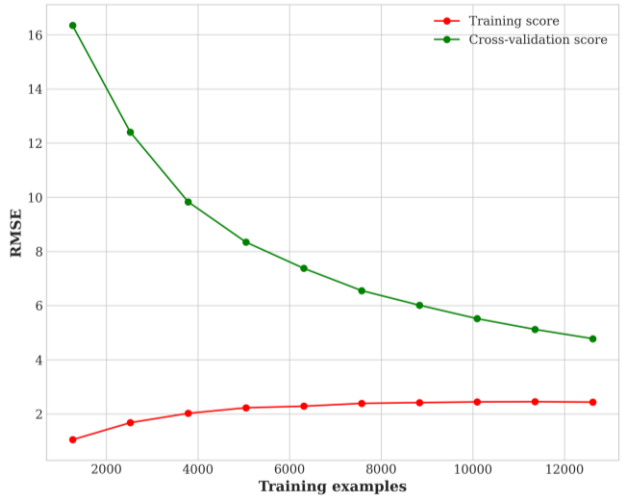

Figure 10. Model learning curve.

Understanding the model's decision-making process is critical for trust and deployment. Figure 11 ranks the top 15 most influential features as determined by the model. It is evident that features engineered to capture the trend (slope) of sensor readings over a long-term window (e.g., sensormeasure11_slope_w45) are the most powerful predictors. This highlights the success of the time-series feature engineering methodology.

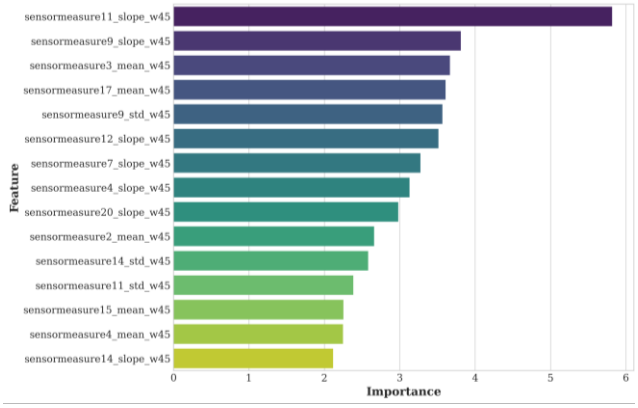

Figure 11. Top 15 feature importances.

The linear relationship between the single most important feature and the RUL is explicitly shown in Figure 12, providing an intuitive confirmation of the physical degradation trend captured by the model.

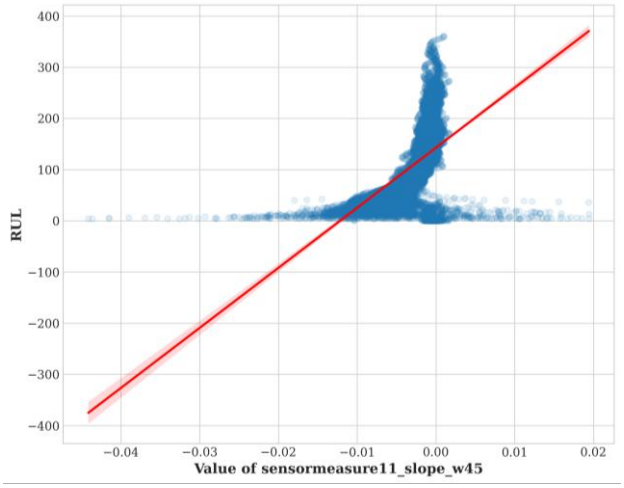

Figure 12. Trend between top features and RUL.

Finally, a SHAP summary plot, demonstrated in Figure 13, provides a deeper layer of explainability. It confirms the feature importance
ranking and reveals the directional impact of each feature. For instance, for the top feature (sensormeasure17_mean_w45), high values (red dots) have a negative SHAP value, meaning they strongly push the model to predict a lower RUL. This granular analysis makes the model's behaviour transparent and aligns with the expected physics of failure.

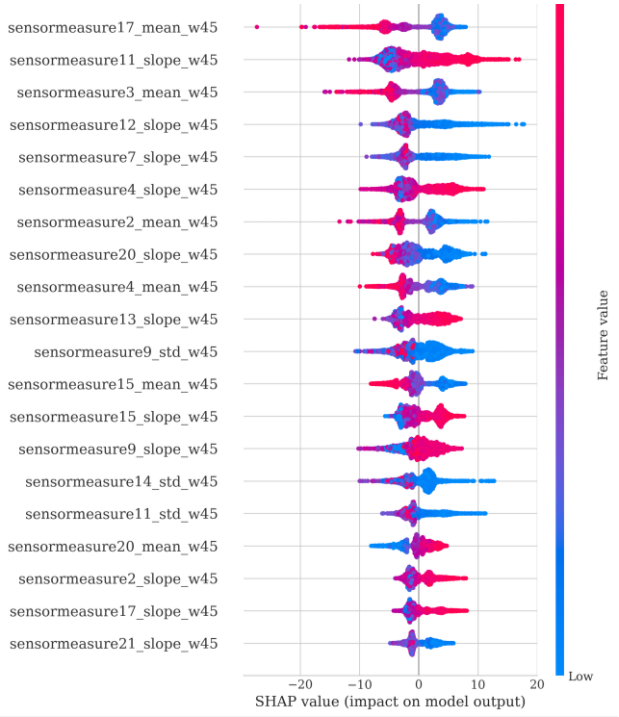

Figure 13. SHAP summary plot.

Edge deployment imposes non-functional constraints such as cost, form factor, and environmental robustness (temperature, vibration). The prototype targets the electrical and timing viability; future engineering validation will integrate industrial enclosures, thermal design, and vibration isolation, and evaluate cost/performance bill-of-materials for target UAV/engine test-rig scenarios.

4. CONCLUSION

This research showcased the utilization of an edge AI device for prognostics on real-time sensor data. A CatBoost Regressor, developed with time-series attributes from unprocessed sensor data, attained remarkable prediction precision with an R^2 of 0.9957 and a mere RMSE of 4.48 cycles. The residuals and error distributions obtained demonstrated no systemic bias, and the evaluation of the learning curve emphasized the model's generalization abilities and resilience. As for the SHAP-based explainability, it detected the degradation-related characteristics that coincide with the anticipated failure physics, therefore enhancing the overall performance of the model.

The experimental comparison and assessment of the Nvidia Jetson AGX Orin against a mid-range desktop PC presented distinct trade-offs in energy efficiency and processing power. Although the PC exceeded the Jetson in both training speed, consuming 18 minutes compared to the Jetson AGX Orin which consumed approximately 3 hours, and an inference latency of 2.8 ms versus 7.5 ms for the Jetson AGX Orin. Both systems satisfied the real-time requirement for processing a data stream of 30 Hz. Significantly, the Jetson performed these outcomes consuming below one-tenth of the energy usage of the PC, showcasing its suitability for edge deployment where portability and efficiency are crucial.

In conclusion, the primary focus of this paper is the performance characterization of the hardware setup and the efficiency of the data streaming pipeline, rather than an exhaustive exploration of the machine learning model itself. As such, the fine-grained details of the model's architecture and hyperparameter optimization are reserved for a future extension of this work. Consecutive work will aim to broaden the assessment to a larger variety of hardware platforms, improve model architectures and hyperparameters, and confirm performance on live engine test rigs.

ACKNOWLEDGEMENT

We would like to express our gratitude to the Technology Innovation Institute (TII) for facilitating the inauguration and development of this research. And we would like to thank Dr Abdulla Alseiari, Director at the Propulsion and Space Research Center (PSRC) at TII for providing his guidance and support throughout this research.

REFERENCES

An, D., Ho Kim, N., & Choi, J.-H. (2013). Options for Prognostics Methods: A review of data-driven and physics-based prognostics. *Annual Conference of the Prognostics and Health Management Society (PHM Society)*.

Artiushenko, V., Lang, S., Lerez, C., Reggelin, T., & Hackert-Oschätzchen, M. (2024). Resource-efficient Edge AI solution for predictive maintenance. *Procedia Computer Science*, *232*, 348–357. https://doi.org/10.1016/j.procs.2024.01.034

Bala, A., Rashid, R. Z. J. A., Ismail, I., Oliva, D., Muhammad, N., Sait, S. M., Al-Utaibi, K. A., Amosa, T. I., & Memon, K. A. (2024). Artificial intelligence and edge computing for machine maintenance-review. *Artificial Intelligence Review*, *57*(5). https://doi.org/10.1007/s10462-024-10748-9

Ghadekar, P., Manakshe, A., Madhikar, S., Patil, S., Mukadam, M., & Gambhir, T. (2024). Predictive Maintenance for Industrial Equipment: Using XGBoost and Local Outlier Factor with Explainable AI for analysis. *Proceedings of the 14th International Conference on Cloud Computing, Data Science and Engineering, Confluence 2024*, 25–30. https://doi.org/10.1109/Confluence60223.2024.10463280

Hector, I., & Panjanathan, R. (2024). Predictive maintenance in Industry 4.0: a survey of planning models and machine learning techniques. *PeerJ Computer Science*, *10*, 1–50. https://doi.org/10.7717/peerj-cs.2016

Hong, C. W., Lee, C., Lee, K., Ko, M. S., Kim, D. E., & Hur, K. (2020). Remaining useful life prognosis for turbofan engine using explainable deep neural networks with dimensionality reduction. *Sensors (Switzerland)*, *20*(22), 1–19. https://doi.org/10.3390/s20226626

Ndao, M. L., Youness, G., Niang, N., & Saporta, G. (2025). Improving predictive maintenance: Evaluating the impact of preprocessing and model complexity on the effectiveness of eXplainable Artificial Intelligence methods. *Engineering Applications of Artificial Intelligence*, *144*. https://doi.org/10.1016/j.engappai.2025.110144

Nohara, Y., Matsumoto, K., Soejima, H., & Nakashima, N. (2022). Explanation of machine learning models using shapley additive explanation and application for real data in hospital. *Computer Methods and Programs in Biomedicine*, *214*. https://doi.org/10.1016/j.cmpb.2021.106584

Pandey, R., Uziel, S., Hutschenreuther, T., & Krug, S. (2023). Towards Deploying DNN Models on Edge for Predictive Maintenance Applications. *Electronics (Switzerland)*, *12*(3). https://doi.org/10.3390/electronics12030639

Park, D., Kim, S., An, Y., & Jung, J. Y. (2018). Lired: A light-weight real-time fault detection system for edge computing using LSTM recurrent neural networks. *Sensors (Switzerland)*, *18*(7). https://doi.org/10.3390/s18072110

Sharanya, S., Venkataraman, R., & Murali, G. (2022). Edge AI: From the Perspective of Predictive Maintenance. In *Applied Edge AI* (pp. 171–192). Auerbach Publications. https://doi.org/10.1201/9781003145158-7

Short, M., & Twiddle, J. (2019). An industrial digitalization platform for condition monitoring and predictive maintenance of pumping equipment. *Sensors (Switzerland)*, *19*(17). https://doi.org/10.3390/s19173781

Su, H., & Lee, J. (2024). Machine Learning Approaches for Diagnostics and Prognostics of Industrial Systems Using Open Source Data from PHM Data Challenges: A Review. *International Journal of Prognostics and Health Management*, *15*(2). https://doi.org/10.36001/ijphm.2024.v15i2.3993

BIOGRAPHIES

Salama Almheiri was born in the United Arab Emirates. She earned her Master of Science in Electrical Power Engineering with distinction from the University of Edinburgh, United Kingdom, in 2021. Salama has two years of experience at the Propulsion and Space Research Center, Technology Innovation Institute, Abu Dhabi, where she currently serves as a Senior Associate Researcher in the Prognostics and Health Management team. Her research focuses on predictive maintenance, edge AI, with applications to aerospace propulsion systems, hydrogen fuel-cell UAVs, and solar powered UAVs.

Zahi M. Omer is a Senior Researcher at the Propulsion & Space Research Center, Technology Innovation Institute, Abu Dhabi. His work sits at the intersection of applied AI, intelligent control, and predictive analytics for energy and propulsion systems. He develops data-driven and model-informed methods for prognostics and health management, including Remaining Useful Life estimation, and leads hardware-in-the-loop development and real-time validation for photovoltaics, rechargeable battery systems, and smart microgrids. His portfolio spans AI-enabled controllers, edge inference pipelines, and reliability-centric analytics. Dr. Zahi holds a Ph.D. in Electrical Engineering from UAE University and has 12 years of combined industrial and academic experience.

Mohamed Ragab received the Ph.D. degree in Computer Science and Engineering from Nanyang Technological University (NTU), Singapore, in 2022, with a focus on developing robust and transferable AI for real-world predictive maintenance. He is currently a Researcher with the Propulsion and Space Research Center, Technology Innovation Institute (TII), Abu Dhabi, United Arab Emirates,

and an Adjunct Researcher with the Agency for Science, Technology, and Research (A*STAR), Singapore. His research interests include industrial AI, transfer learning, machine fault diagnosis, and prognosis. He has been recognized with several honors, including the Finalist Paper Award at the IEEE International Conference on Prognostics and Health Management and the prestigious A*STAR Career Development Award.

Abdulla Alseiari is Director of AI Diagnostics & Prognostics at the Propulsion and Space Research Centre, Technology Innovation Institute (TII), Abu Dhabi. He has over 20 years of experience in predictive maintenance and AI applications across the energy and aerospace sectors, including 18 years with TAQA. His research focuses on Prognostics and Health Management (PHM), digital twins, anomaly detection, and decision-support systems for airbreathing propulsion. Dr. Abdulla is a board member at the International Society for Air Breathing Engines (ISABE) and holds a Ph.D. in Maintenance Engineering & AI from the University of Greater Manchester, an MSc in Maintenance Engineering from Glasgow Caledonian University, and a BSc in Electrical Engineering from Al Ain University.

Nouf Almesafri was born in the United Arab Emirates. She earned her Bachelor of Science in Aerospace Engineering from Khalifa University, Abu Dhabi, in 2022, and her Master of Science in Applied Artificial Intelligence from Cranfield University, United Kingdom, in 2024. Nouf has three years of experience at the Propulsion and Space Research Center, Technology Innovation Institute, Abu Dhabi, where she currently serves as a Senior Associate Researcher. Her research focuses on computer vision, deep learning, and reinforcement learning, with an emphasis on applying advanced AI techniques to aerospace and propulsion systems.