Cloud-Enabled Autoencoder-Based Anomaly Detection for Gas Turbine Faults

Nima Ameri[2], Felipe Montana[1], Oscar Mendoza[1], Visakan Kadirkamanathan[1], Philip Naylor[2], and Will Jacobs[1]

[1] *University of Sheffield, Sheffield, S1 3JD, UK*
f.montana-gonzalez@sheffield.ac.uk
w.jacobs@sheffield.ac.uk
o.e.mendoza@sheffield.ac.uk
visakan@sheffield.ac.uk
a.r.mills@sheffield.ac.uk

[2] *Rolls-Royce Plc, London, N1 9FX, UK*
nima.ameri@rolls-royce.com
philip.naylor@rolls-royce.com

ABSTRACT

Engine Health Monitoring (EHM) is critical to reducing service disruption and operational costs in the aerospace industry. While traditional physics-based algorithms have been effective, the increasing volume and complexity of sensor data necessitate more scalable, automated, and accurate detection methods. This paper discusses the transition of Rolls Royce EHM capability - for civil and commercial aircraft engines - toward a modern Machine Learning (ML) paradigm within the Rolls-Royce Data Science Environment (DSE). Leveraging a cloud-based architecture and its tech stack, we address the challenges of manual recalibration and high false-positive rates. We present a case study of an anomaly detection framework utilizing a two-stage approach: a deep neural network for input-output residual generation followed by an autoencoder with a custom loss function for latent representation. Furthermore, we outline the integration of MLflow to ensure robust experiment tracking, as well as the use of a cloud-based unified data governance framework. This work demonstrates how end-to-end ML lifecycle management maintains model performance and operational trust in a highly regulated environment.

1. INTRODUCTION

Maintaining the operational integrity of commercial aircraft engines requires sophisticated Engine Health Monitoring (EHM) systems. Gas turbine engines (GTEs) are critical components in modern aviation, where their high power-to-weight ratio, efficiency, and reliability have led to widespread adoption. As complex, high-value systems, any unexpected anomaly or fault can cause the grounding of the asset, leading to downtime while the asset is repaired. At the fleet level, GTEs typically operate without redundancy; any unplanned downtime has a cascading effect through a system, often resulting in severe financial penalties, such as airport stand occupancy fees and customer reimbursement. Therefore, accurate and timely anomaly detection is essential for minimizing downtime and optimizing maintenance schedules.

Traditionally, Rolls-Royce has relied on a suite of expert-designed algorithms to ingest sensor data and generate actionable maintenance insights. While these methods are historically successful, the modern aerospace landscape presents two significant opportunities. First, the exponential increase in the volume and variety of available data offers untapped potential that traditional manual methods struggle to exploit. Second, the design and calibration of these legacy algorithms remain labor-intensive, requiring extensive subject matter expert (SME) intervention to transition solutions from development to production.

The task of monitoring aerospace GTEs presents a significant challenge (Fentaye, Baheta, Gilani, & Kyprianidis, 2019), characterized by a number of confounding factors. These engines operate under a wide range of dynamic conditions, including fluctuations in altitude, speed, and ambient temperature, which cause their observed sensor measurements to vary continuously (Verhagen et al., 2023). Furthermore, GTEs

are subject to slow, time-varying degradation over their lifespan—a normal process that also manifests as gradual changes in sensor measurements (Cruz-Manzo, Panov, & Zhang, 2018). The full set of confounding sources of variance can be described as:

- **Build variation:** Manufacturing differences between assets lead to inter-asset variation at the point of entry into service.
- **Non-static engine condition:** The behavior of the engine changes over time due to factors such as age and degradation.
- **Operating conditions:** The engine is affected by measured and unmeasured environmental factors such as temperature and altitude
- **Discrete events:** The engine dynamics can substantially change from one flight to another after an engine intervention.
- **Noisy measurements:** Variability in the conditions under which data points are recorded affects the measurement.

These sources of variation can mask the subtle signatures of a true anomaly or fault, making it difficult to distinguish between normal behavior and a genuine problem. Compounding these technical difficulties is the extreme imbalance inherent in real-world fleet data, where genuine anomalous events typically make up significantly less than 1% of the total dataset.

To address these challenges, there is a strong motivation to adopt advanced machine learning (ML) approaches. However, the adoption of ML in aerospace is not without challenges. Any new system must maintain exceptionally low false-positive rates; spurious detections increase the operational burden on front-line staff and can rapidly erode the user trust necessary for flight-critical decision-making. Rolls-Royce has developed a cloud-based Data Science Environment (DSE) that provides an opportunity to address these requirements: The DSE provides a highly integrated, cloud-based platform for the secure development and deployment of analytics. By leveraging elastic computational resources and commercial off-the-shelf (COTS) capabilities, specifically MLflow for lifecycle management, the DSE enables the scalable collaboration and data management required for highly regulated aerospace applications.

This paper presents a case study of an anomaly detection application that utilizes a robust two-stage ML architecture designed to systematically decouple engine health from sources of variance. The first stage employs a deep neural network (DNN) to model engine input-output behavior, effectively "cleansing" the data by producing residuals that serve as health-indicative features. The second stage utilizes an autoencoder with a custom loss function to learn a latent representation for anomaly detection. The anomaly detection framework is based on previous work by the authors (Montana et al., 2026). Recognizing that model performance naturally degrades over time as engine fleet dynamics shift, we further detail the development of an automated ML pipeline. This pipeline incorporates mechanisms for continuous monitoring, retraining, and controlled promotion to maintain operational effectiveness. This case study serves as a blueprint for the lifecycle management of modern ML models within a governed industrial framework.

2. METHODOLOGY

2.1. Two-Stage Anomaly Detection

We use a two-step framework designed to first systematically isolate engine health from operational variance and then perform anomaly detection. The anomaly detection framework was first proposed in (Montana et al., 2026) and is subject to pending patent applications (Montana, Jacobs, Ameri, Naylor, & Mills, 2024a, 2024b, 2024c). The approach is summarised here. The first step uses an Engine model to generate model residuals by subtracting predicted nominal behavior from observed sensor data, effectively filtering out fluctuations caused by changing operating conditions. In the second step, these health-indicative features are processed by a time-constrained autoencoder, which maps the residuals into a latent space for anomaly detection, see Figure 1.

2.1.1. Step 1: The Engine Model

The first stage of the framework employs a deep feedforward neural network, $E(\cdot;\theta_E)$, designed to capture the nominal dynamics of the engine across a diverse range of operating conditions. By training the model on fleet-level data, the network learns to map engine inputs and measured disturbances to predicted sensor outputs, effectively representing average asset behavior over the fleet. The architecture consists of three hidden, fully connected layers—each with 32 neurons and LeakyReLU activation functions—concluding in a fully connected output layer. This configuration was found to be robust, with performance remaining stable across various architectural iterations. The resulting model serves as a baseline for generating residuals, allowing for the decoupling of environmental and operational effects from the underlying health of the engine.

2.1.2. Training

Training was performed using the Adam optimizer with a learning rate of 0.001 and a batch size of 128, with convergence monitored via validation error on a 75/5/20% data split.

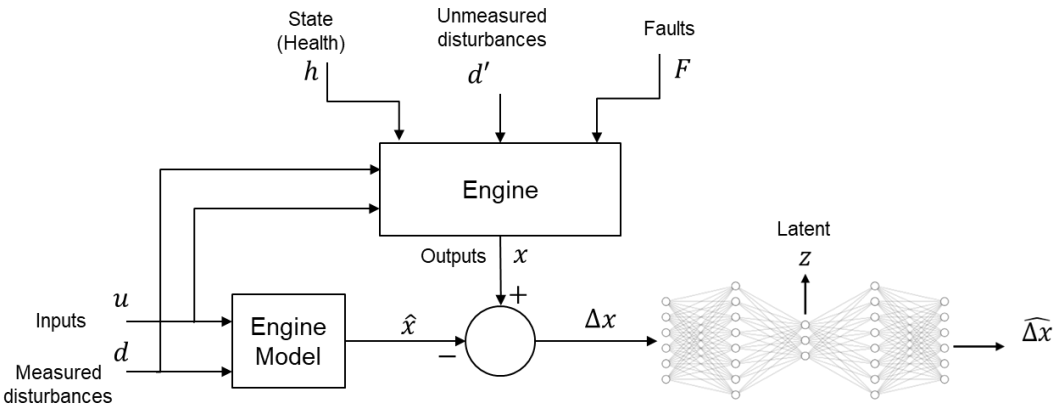

Figure 1. Overview of the two-stage anomaly detection framework. Stage 1 utilizes a multi layer perceptron (a type of DNN) to model nominal engine dynamics and generate health-indicative residuals (Δx) by removing operational variance. Stage 2 processes these residuals through a time-constrained autoencoder (AE) to map engine behavior into a latent space (z) for localized anomaly scoring.

2.2. Step 2: Anomaly Detection with a Time-Constrained Autoencoder

The detection framework moves from the measurement space to a latent space by analyzing the model residuals, $\Delta x = x - \hat{x}$. Because the initial engine model captures the expected nominal behavior, these residuals effectively isolate unmeasurable factors such as faults and degradation. To process these features, we employ an autoencoder (AE) consisting of an encoder $f(\cdot)$, which maps residuals to a lower-dimensional latent representation z, and a decoder $g(\cdot)$, which reconstructs the residuals as $\Delta\tilde{x}$. Both networks are implemented as deep feedforward structures with two hidden layers (size 20 and 32).

2.2.1. Time-Constrained Loss Function

Standard AEs training typically minimizes a reconstruction loss to learn a meaningful representation. However, such an approach ignores the temporal evolution of the engine. In GTE applications, while consecutive flights should exhibit similar behavior, the engine also undergoes a slow nominal degradation over its lifespan. Conversely, faults typically manifest as relatively rapid changes.

To encode this behavior, we use a novel time-constrained loss function, $\mathcal{L}$, which penalizes mapping data from the same engine far apart in the latent space based on their temporal proximity. This is controlled by a weighting function:

$$\lambda_{ik} = e^{-\gamma \Delta t} \tag{1}$$

where Δt is the time difference between flights i and k, and γ is a hyperparameter representing the expected rate of degradation. This constraint ensures that points close in time are mapped close together in the latent space, while the penalty decays as the time between recordings increases. Notably, λ_{ik} is set to zero for data points from different engines or those separated by a maintenance intervention, as these represent distinct behavioral shifts.

2.2.2. Training

The autoencoder is trained using the Adam optimizer with a decaying learning rate and a batch size of 128. To ensure the loss is effectively calculated during training, a custom batch sampler is utilized to provide a sufficient number of data points from the same engine within each training batch. This integrated loss approach results in a feature space that is highly sensitive to anomalous deviations from the expected temporal trend of the asset (Montana et al., 2026).

2.2.3. Anomaly Scoring

To account for the temporal shift in engine behavior, the anomaly score s_i is defined as the squared Euclidean distance between the current latent representation z_i and the mean center c_i of the engine's previous k data points:

$$s_i = \| \, z_i - c_i \, \|^2 \tag{2}$$

By comparing an asset to its own recent history, the detector isolates rapid fault signatures from slow degradation. The detection threshold can be tuned to maximize sensitivity or strictly limit false positives. To prevent spurious alerts following maintenance, the history window k is reset or adjusted based on recorded interventions, ensuring the system remains robust to known operational shifts.

2.3. Cloud Environment Description

The Rolls-Royce Cloud-based DSE has enabled the development of this novel anomaly detection framework by leveraging the following key technologies

- **A code-first analytical platform:** An Apache Spark-based cloud platform that enables organizations to process, analyse, and collaborate on large-scale data and AI projects. It provides an integrated environment for data engineering, data science, and machine learning, offering tools for data storage, processing, and visualization. The platform supports collaborative workspaces, scalable compute resources, and advanced analytics, making it easier for teams to build, deploy, and manage data-driven solutions efficiently and securely across various cloud providers.
- **Data Governance:** EHM data for the fleet are directly available in the DSE and saved at regular intervals. Data is stored using the open source "Delta" data format based on Apache Parquet. This format is optimized for Spark-based distributed computation: This involves generating identity columns, setting table properties, and managing unique tuples; this format is also best fitted for analytic workloads and thanks to Delta transaction log files, it provides atomicity, consistency, isolation, and durability (ACID) transactions and isolation level to Spark. This approach provides also a standardise the way to save and store data for different design trials and use cases.
- **MLflow**: MLflow is a versatile, expandable, open-source platform for managing workflows and artifacts across the machine learning lifecycle. It has built-in integrations with many popular ML libraries, but can be used with any library, algorithm, or deployment tool. MLflow was extensively utilised to manage the various experiments and trials generated during the development phase.
- **Computational resources:** The environment enables the selection of the computing resources and relative runtime available to match the cost and time constraint for the project. Available runtime are also available with pre-built machine learning and deep learning infrastructure including the most common ML and DL libraries. Each cluster can be flexibly configured with the right amount of computational resources and cost required for the specific job allowing both parallel and distributed computation.
- **Code repository:** Code base management is ensured by having a centralised Cloud-based repository enabling seamless multi-user collaboration for code development and governance.

2.4. MLOps and Lifecycle Management

ML models in EHM operate in highly dynamic environments where data distributions shift over time. Studies indicate that upwards of 90% of monitored ML models experience performance decay in production (Vela et al., 2022). In the GTE context, this model decay is driven by Data Drift (changes in input distributions) and Concept Drift (changes in the relationship between inputs and targets) (Lu et al., 2018). Without a robust framework for reproducibility, monitoring, and rollback, advanced AI remains experimental rather than operational.

2.4.1. Drivers of Model Decay in EHM

The requirement for continuous model maintenance is driven by the progressive divergence of the fleet from its original training baseline:

- **Fleet Evolution:** Over a 6 year period, the fleet composition changed by approximately 7% annually due to modification campaigns, production changes, and mean aging. A model trained in 2018, for example, would encounter a fleet by 2024 where only 45% of members match the original training distribution.
- **Operational Drift:** The operational space also evolves; for instance, the route mix has changed by an average of 6% per year over the same period. A static model trained 6 years ago would be required to predict for more than twice the range of routes seen during initial training, introducing the risk of extrapolating into regions of low data coverage.

2.4.2. Automated Updating Pipeline

To address this, we implemented an automated, end-to-end model updating pipeline within the cloud-based DSE. This framework provides a scalable and auditable foundation for continuous maintenance through several key components:

- **Integrated MLOps Stack:** Leveraging MLflow for traceable experiment tracking, Git for distributed version control, and dedicated governed data access.
- **Drift Detection and Adaptive Selection:** We implemented domain-specific drift detection as a trigger for retraining. Rather than simple full-data retraining, the pipeline utilizes an adaptive dataset selection strategy using statistical windowing to ensure the training set remains representative of the 'live' fleet.
- **Retraining and Controlled Promotion:** When drift thresholds are breached, the pipeline automates model retraining and validation. Successful candidates undergo a controlled promotion process, ensuring that only models meeting rigorous performance and safety criteria are transitioned to the inference environment.

This framework transforms the anomaly detection system from a static tool into a resilient, traceable, and reliable lifecycle that maintains high sensitivity and low false-positive rates despite the non-static nature of aerospace data.

2.4.3. End-to-End Lifecycle Architecture

To transition the anomaly detection framework from an experimental tool to a production-grade system, we implemented a five-stage automated lifecycle:

- **Monitoring:** Compares live inference data against a reference baseline to identify drift and performance decay.
- **Dataset Redefinition:** Updates the training set using reproducible data slices to ensure each retraining cycle is based on high-quality, balanced data.
- **Retraining:** Executes scalable, version-controlled training by pulling specific dataset versions and logging all artifacts to MLflow.
- **Validation:** Conducts automated "champion-challenger" testing to ensure newly trained models outperform existing production versions.
- **Promotion:** Registers verified models in the model registry, providing an audited trail for deployment and rollback.

2.4.4. Domain-Adapted Drift Detection

Standard monitoring tools often treat EHM data as independent tabular features, using statistical metrics like the Wasserstein distance (Ferracuti, Freddi, Monteriu, & Romeo, 2022). However, in a GTE context, natural seasonality and temporal trends can trigger "false drift" alarms. To ensure monitoring remains practically relevant rather than just statistically significant, we adopted the following strategies:

- **Residual-Based Monitoring:** Rather than monitoring raw sensor distributions, drift is computed on the residuals of the engine dynamics model. Since the regression model accounts for predictable seasonal and operational variations, the resulting residuals focus drift detection on unexpected deviations in engine behavior.
- **Calibrated Thresholds:** We moved away from default statistical thresholds in favor of calibrated effect sizes (e.g., 0.2σ). This ensures that retraining is only triggered by persistent, practically relevant shifts that are likely to impact model accuracy.
- **Multivariate Triggering:** To reduce sensitivity to isolated outliers, the pipeline requires persistent drift—where variables are consistently flagged over multiple consecutive weeks—or multivariate alerts before a retraining event is triggered.

2.4.5. Dataset Update Strategies

As the fleet evolves with new engines, changing route mixes, and aging patterns, the training data must be updated to prevent model obsolescence. This framework evaluates two primary strategies for long-term alignment:

1. **Adaptive Windowing:** Utilizing a shifting temporal window to ensure the model is always trained on the most recent, and thus most representative, data. This provides effective short-term adaptation but requires careful management to avoid losing rare, still-relevant fault patterns.
2. **Full-Data Retraining:** Incorporating all historical data to ensure maximum coverage of the evolving fleet, providing a broader baseline for the anomaly detection latent space.

By integrating these strategies into a governed MLOps stack, we provide a robust and auditable foundation for continuous model maintenance in the dynamic data context of GTE EHM.

3. CASE STUDY: IN-SERVICE GTE FLEET

To evaluate the effectiveness of the proposed two-stage framework and the associated MLOps lifecycle, a case study was conducted using a fleet of real, in-service Rolls-Royce engines.

3.1. Dataset Characteristics

The dataset comprises snapshot data collected from 138 distinct engines over a duration exceeding three years. These snapshots are recorded during the stable cruise phase of each flight, ensuring a consistent operational baseline for comparison. The resulting dataset reflects the true complexity of a global engine fleet, encompassing the various confounding factors (such as build variation and operational drift) enumerated in Section 1.

3.2. Labels and Operational Context

Within the fleet data, a small subset of points is labeled as "faulty" based on verified maintenance records, with the remaining majority assumed to be normal. In an industrial context, it is acknowledged that "assumed normal" data may contain unidentified anomalies due to the inherent latency or imperfections in in-service feedback. It is important to note that the faults targeted in this study are of an economic and operational nature; they represent degradation or component shifts that impact fuel efficiency and maintenance costs, rather than the safety or airworthiness of the asset.

3.3. Input-Output Selection

To represent the thermodynamic behavior of the core gas path, 13 data channels were selected as inputs and outputs for the anomaly detection framework. The engine's ambient state is characterized by Altitude (Alt), Mach Number (MN), and Ambient Temperature (T_0). While fuel flow is the physical driving input, it is managed via a feedback control loop on the Low-Power Shaft Speed (NL); thus, NL is used as the primary driving input as it remains invariant in the presence of a fault. The selected outputs characterize temperatures and pressures throughout the gas path, including measurements at the IP compressor outlet (P_{26}), HP compressor outlet (P_{30}, T_{30}),

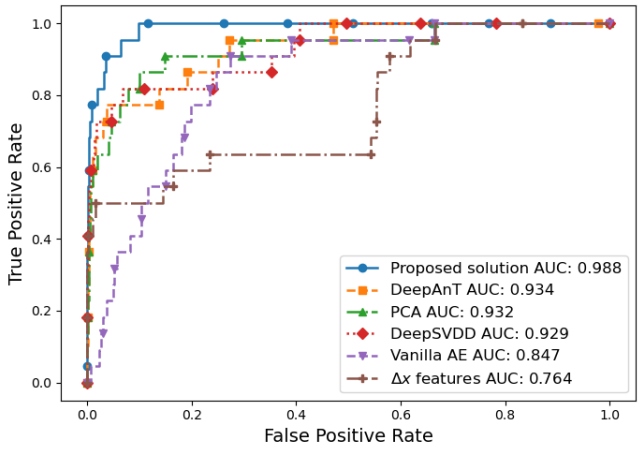

Figure 2. ROC curves for the in-service dataset. The proposed two-stage approach (blue) achieves an AUC-ROC of 0.988, significantly outperforming benchmark methods. The results demonstrate the critical importance of using model residuals (Δx) to prevent anomalies from being masked by operational noise. Recreated from (Montana et al., 2026) with permission.

HP turbine outlet (P_{42}), IP turbine outlet (P_{44}), Turbine Gas Temperature), and LP turbine outlet (T_{50}). Additionally, HP shaft speed (NH) and fuel flow are included to provide a comprehensive profile of the engine's internal performance.

3.4. Application of the ML Pipeline

This real-world data serves as the basis for testing the end-to-end pipeline. The engine model described in Section 2.1.1 was first used to generate residuals across the 138 engines, followed by the time-constrained autoencoder, described in Section 2.2, to map these residuals into the latent space for anomaly scoring. This study specifically examines how the automated retraining and drift-detection strategies described in Section 2.4 maintain the detection performance as the fleet composition and route mixes evolve over a three-year window. However, due to the lack of curated labels spanning the three-year lifecycle, detection performance (e.g., TPR/FPR) cannot be quantitatively assessed over time. Instead, the prediction error of the engine model and the stability of the autoencoder's latent space distributions are used as primary indicators of model health and the effectiveness of the updating strategies.

While substituting ground-truth performance metrics with proxy indicators is a standard and necessary practice in industrial contexts (where verified maintenance labels often significantly lag behind live data) this approach carries inherent limitations. Primarily, maintaining a stable latent space distribution and a low mean absolute error (MAE) on nominal data does not definitively guarantee that the True Positive Rate is preserved for novel, previously unseen fault modes. Furthermore, without continuous labeling, it is impossible to precisely quantify the exact operational cost savings.

4. RESULTS

To demonstrate the efficacy of the two-stage framework and the lifecycle management, we first evaluate the performance of the Engine model. The objective of this initial stage is to isolate engine health by removing the dominant variance introduced by varying operating conditions.

4.0.1. Engine Model Validation

The engine model serves as the foundation of our framework, predicting a nominal response based on the input parameters described in Section 4.2. The model achieves high accuracy on the in-service dataset, with an average Mean Absolute Percentage Error (MAPE) below 1%. This level of precision demonstrates the model's ability to generalize across the fleet despite the presence of measurement noise.

4.0.2. Anomaly Detection Results

The model residuals produced by the validated engine model from Section 4.0.1 are passed to the autoencoder to perform detection.

In a real-world maintenance context, where hundreds of flights are recorded daily, minimizing false alarms is critical. By applying a cost-constrained threshold (fixed at a maximum FPR of 0.02), our proposed method outperforms all others. Despite the low absolute Precision (0.015) caused by extreme data imbalance, the method demonstrated high practical utility:

- **AUC-ROC:** 0.988
- **TPR (Sensitivity):** 0.818
- **FPR (False Alarm Rate):** 0.020

By successfully identifying 81.8% of true faults while maintaining a 2% false alarm rate, the framework provides a significant operational advantage, maximizing fault captures while minimizing unnecessary manual inspections of aircraft on the ground.

Our framework was evaluated against several traditional and state-of-the-art baselines, including PCA (Basora, Olive, & Dubot, 2019), DeepSVDD (Ruff et al., 2018), and DeepAnT (Munir, Siddiqui, Dengel, & Ahmed, 2019). The results indicate a significant performance advantage for the proposed method; as evidenced by the high AUC-ROC values, it maintains higher detection fidelity than all benchmark models on the real-world dataset, see Figure 2."

4.1. Model Drift and Performance Recovery

As expected by the fleet evolution analysis, the base DNN model exhibited a steady increase in Mean Absolute Error (MAE) for key parameters like TGT over the inference period, see Figure 3. This performance decay confirms the presence of data and concept drift. Upon triggering the updating pipeline,

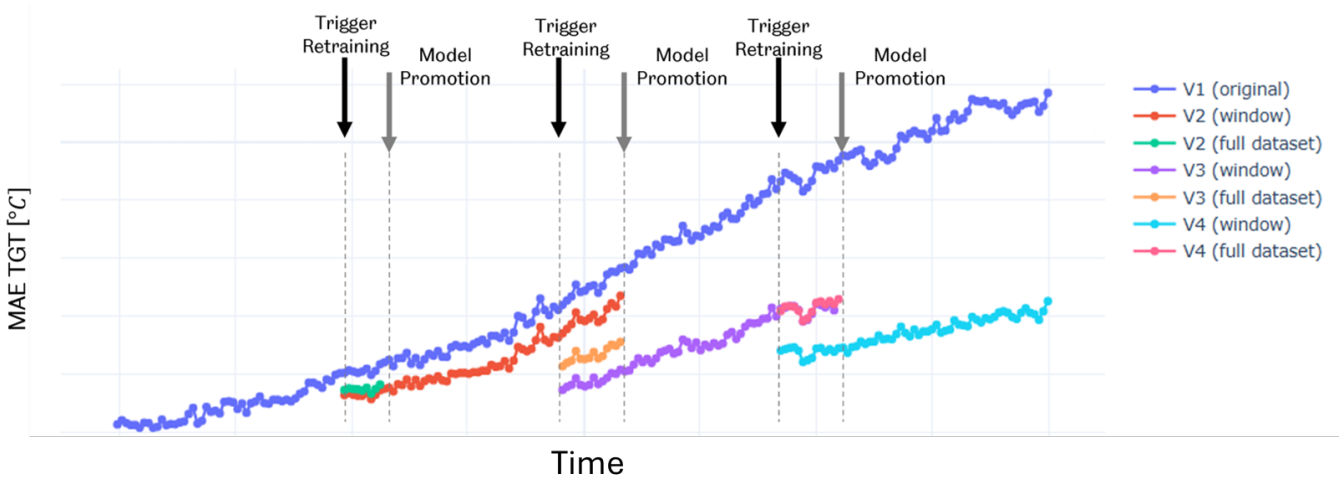

Figure 3. Weekly Mean Absolute Error (MAE) for the TGT prediction. The base model (V1) shows steady performance decay over time due to data drift. Both the "Full-Data" and "Adaptive Window" retraining strategies achieve significant performance recovery, with the adaptive window (green) providing the most localized fit. The time axis has been anonymised.

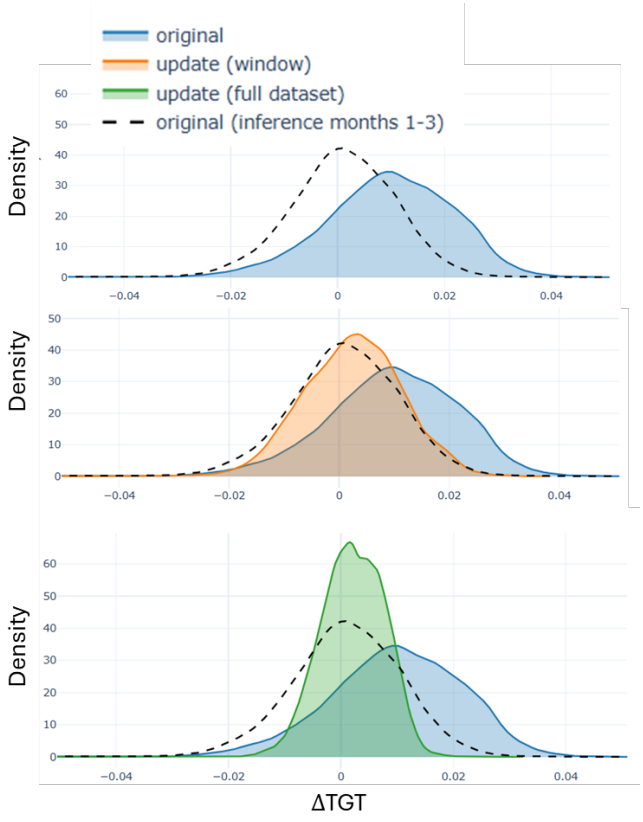

Figure 4. Shift in the Autoencoder reconstruction error distribution over time. The initial inference distribution (dashed) diverges as the fleet evolves (blue), signaling model drift. Following an automated retraining trigger, the distribution is reset toward the nominal baseline, restoring the detector's sensitivity.

both retraining strategies successfully achieved performance recovery, with a significant drop in MAE. In this case study, the adaptive window approach yielded the lowest MAE for TGT, suggesting it effectively prioritized the most recent, relevant fleet signatures. It also gave a significant computational saving with $\tilde{3}5\%$ lower CPU time in comparison to the whole data retraining.

A similar trend was observed in the autoencoder's reconstruction behavior. Comparing the initial three months of inference data to subsequent periods showed a distinct shift in error distributions, signifying model drift, see Figure 4. After retraining, these distributions shifted back toward the baseline, effectively "resetting" the model's sensitivity to nominal behavior. While both strategies recovered performance, full-data retraining provided more consistent reconstruction across the broader fleet history.

5. CONCLUSION

The transition toward data-driven EHM in the aerospace industry requires a balance between sophisticated anomaly detection and rigorous operational governance. This paper has presented the lifecycle of a robust, two-stage deep learning framework designed to isolate engine health from the complex confounding variables inherent in gas turbine operations. By utilizing a DNN to generate model residuals and a time-constrained autoencoder to map these into a latent space, we successfully decoupled engine degradation from environmental and operational noise.

Validation on a real-world fleet of 138 gas turbine engines demonstrated the framework's superior performance under extreme class imbalance. The method achieved an AUC-ROC of 0.988 and a True Positive Rate of 0.818 while maintaining a

strictly constrained False Positive Rate of 0.02. These results confirm that identifying over 80% of true faults is achievable even when anomalies represent less than 0.1% of the total dataset, provided that operational variance is systematically removed.

Crucially, this study highlights that a static ML model is insufficient for long-term deployment in a dynamic fleet environment. Through the Rolls-Royce Data Science Environment (DSE), we implemented an automated MLOps pipeline using MLflow to manage model decay. Our comparison of updating strategies revealed that Adaptive Windowing offers a sustainable approach to model maintenance, reducing computational costs by 35% in comparison to to whole data retraining, while effectively recovering performance lost to data drift.

Specifically, this research advances the Prognostics and Health Management (PHM) domain knowledge in three distinct ways:

- **Robustness against Confounding Variables:** It bridges the gap between theoretical deep learning and practical EHM application by establishing a proven methodology for decoupling operational and environmental noise from true asset degradation/faults.
- **Managing Extreme Class Imbalance:** Anomaly detection is demonstrated in real-world scenarios under extreme class imbalance. By achieving high diagnostic accuracy when anomalies are less than 0.1% of the data, it demonstrates that data-driven PHM systems can be highly sensitive even when learning from data with very few fault examples.
- **Sustaining Models in Production (MLOps for PHM):** It extends the PHM literature beyond initial model deployment to address the critical challenge of model degradation. By validating the Adaptive Windowing strategy, this work provides empirical evidence for how to efficiently combat data drift and sustain diagnostic accuracy over the entire lifecycle of an asset.

In summary, this work provides a blueprint for the end-to-end lifecycle management of ML models in highly regulated industries. The integration of domain-specific drift detection, automated retraining, and controlled promotion ensures that EHM systems remain accurate, traceable, and trusted by front-line engineering teams. Future work will explore the scaling of this framework across broader engine architectures and the further refinement of multivariate triggering to enhance the resilience of the global fleet.

ACKNOWLEDGMENT

This work was supported by Innovate UK [grant number 10112182]

REFERENCES

Basora, L., Olive, X., & Dubot, T. (2019). Recent advances in anomaly detection methods applied to aviation. *Aerospace*, *6*(11), 117.

Cruz-Manzo, S., Panov, V., & Zhang, Y. (2018, Oct). Gas path fault and degradation modelling in twin-shaft gas turbines. *Machines*, *6*(4), 43. doi: https://doi.org/10.3390/machines6040043

Fentaye, A. D., Baheta, A. T., Gilani, S. I., & Kyprianidis, K. G. (2019). A review on gas turbine gas-path diagnostics: State-of-the-art methods, challenges and opportunities. *Aerospace*, *6*(7). doi: 10.3390/aerospace6070083

Ferracuti, F., Freddi, A., Monteriu, A., & Romeo, L. (2022, Jul). Fault diagnosis of rotating machinery based on wasserstein distance and feature selection. *IEEE Transactions on Automation Science and Engineering*, *19*(3), 1997–2007. doi: https://doi.org/10.1109/tase.2021.3069109

Lu, J., Liu, A., Dong, F., Gu, F., Gama, J., & Zhang, G. (2018). Learning under concept drift: A review. *IEEE Transactions on Knowledge and Data Engineering*, 1–1. doi: https://doi.org/10.1109/tkde.2018.2876857

Montana, F., Anderson, S., Mills, A., Naylor, P., Ameri, N., & Jacobs, W. (2026). Deep anomaly detection for gas turbine engines using model residuals and time-constrained autoencoders. *Engineering Applications of Artificial Intelligence*.

Montana, F., Jacobs, W., Ameri, N., Naylor, P., & Mills, A. (2024a, December). *Next generation diagnostic network - a method of training an encoder model.* UK Patent Application 2024P00471 GB. (Pending)

Montana, F., Jacobs, W., Ameri, N., Naylor, P., & Mills, A. (2024b, December). *Next generation diagnostic network - autoencoder.* UK Patent Application 2024P00260 GB. (Pending)

Montana, F., Jacobs, W., Ameri, N., Naylor, P., & Mills, A. (2024c, December). *Next generation diagnostic network - fault determination.* UK Patent Application 2024P00258 GB. (Pending)

Munir, M., Siddiqui, S. A., Dengel, A., & Ahmed, S. (2019). Deepant: A deep learning approach for unsupervised anomaly detection in time series. *IEEE Access*, *7*, 1991-2005. doi: 10.1109/ACCESS.2018.2886457

Ruff, L., Vandermeulen, R., Goernitz, N., Deecke, L., Siddiqui, S. A., Binder, A., ... Kloft, M. (2018). Deep one-class classification. In *International conference on machine learning* (pp. 4393–4402).

Vela, D., Sharp, A., Zhang, R., Nguyen, T., Hoang, A., & Pianykh, O. S. (2022, Jul). Temporal quality degradation in ai models. *Scientific Reports*, *12*(1). doi: https://doi.org/10.1038/s41598-022-15245-z

Verhagen, W. J. C., Santos, B. F., Freeman, F., van Kessel, P., Zarouchas, D., Loutas, T., ... Heiets, I. (2023,

Aug). Condition-based maintenance in aviation: Challenges and opportunities. *Aerospace*, *10*(9), 762. doi: https://doi.org/10.3390/aerospace10090762