FMDB *Transactions on Sustainable Computing Systems*

A Real-Time Explainable Hybrid AI Framework for Predictive Brake Fault Diagnosis with Multilingual Dashboard Alerts in Heavy Transport Vehicles

N. T. Sanjaai Raajan[1], S. Vaishnavii[2,*], Trilok Srinivasan[3], S. V. Siddarth Reddy[4]

[1,2,3,4]Department of Computer Science and Engineering, SRM Institute of Science and Technology, Ramapuram, Chennai, Tamil Nadu, India.
sn6592@srmist.edu.in[1], vaishn219@gmail.com[2], ts5111@srmist.edu.in[3], ss7275@srmist.edu.in[4]

Abstract: Heavy transport mode vehicles, such as trucks and buses, use very large amounts of air braking systems to ensure operational safety. The braking mechanism - any failure can cause serious accidents, loss of life and economic damage. Conventional methods for brake maintenance are mostly reactive, based on manual inspection and scheduled servicing, and may not detect faults early. With the development of Artificial Intelligence (AI) and Machine Learning (ML), Predictive Maintenance (PdM) techniques can analyse large volumes of sensor data generated by vehicle systems to identify hidden patterns and early signs of brake deterioration. However, many AI-based systems operate as black-box models, with predictions that are hard to interpret, which can cause distrust in safety-critical environments, including transportation. To address this issue, this paper presents a real-time, explainable hybrid artificial intelligence system for detecting brake defects in heavy transport vehicles. The proposed system uses a Random Forest model for current-sample scoring and an LSTM model to analyse the latest 5 sequential samples, with the Random Forest output applied early in warmup before switching to RFLSTM late fusion. The framework further converts the fault probability into a vehicle health score and incorporates SHAP-based explanation for unbiased decision support. Multilingual Streamlit dashboard with separate Driver View and Technical View modes, recommendation messages, trend and timeline visualisation, and cooldown-controlled WhatsApp and SMS alerts, integrated to facilitate drivers and maintenance personnel during replay-based monitoring.

Keywords: Predictive Maintenance; Machine Learning; Explainable Artificial Intelligence; Brake Fault Prediction; Air Brake System; Heavy Transport Vehicles; Long Short-Term Memory.

Received on: 03/05/2025, **Revised on:** 28/06/2025, **Accepted on:** 01/10/2025, **Published on:** 12/06/2026

Journal Homepage: https://www.fmdbpub.com/user/journals/details/FTSCS

DOI: https://doi.org/10.69888/FTSCS.2026.000687

Cite as: N. T. S. Raajan, S. Vaishnavii, T. Srinivasan, and S. V. S. Reddy, "A Real-Time Explainable Hybrid AI Framework for Predictive Brake Fault Diagnosis with Multilingual Dashboard Alerts in Heavy Transport Vehicles," *FMDB Transactions on Sustainable Computing Systems*, vol. 4, no. 2, pp. 135–148, 2026.

1. Introduction

High reliability of commercial vehicles working at high load requirements due to faults in critical subsystems being able to cause safety incidents and service interruption (and expensive repair) cycles. Of these subsystems, the air pressure system (APS) is important operationally because it facilitates braking and other Pneumatic Functions in Heavy Trucks [6]; [16]. A

*Corresponding author.

missed APS- related failure can therefore escalate to a brake safety risk and a high-cost maintenance event. This cost imbalance is also reflected in the data set, where missing a true APS fault has a much higher penalty than doing an unnecessary inspection [1]. Conventional approach to maintenance depending on or relying on periodic inspection of or human interpretation of noisy operational signals [5]; [21]. Such methods can be slow, subjective, and reactive. Data-Driven predictive maintenance addresses this gap by transforming historical operating data into early-warning signals that prompt preventive maintenance before a serious breakdown. Machine learning is particularly appropriate in this setting given the APS dataset, which contains high-dimensional, anonymised sensor attributes, missing values, and severe class imbalance [12]; [13]. Rather than presenting a classifier in a single department, this work develops a full pipeline for a prototype. The paper includes raw data preprocessing, model training, hybrid probability computation, generation of health scores, explainability, dashboard visualisation, and alert routing [15].

1.1. Problem Statement

The technical objective of the paper is to predict whether an incoming vehicle's operating state corresponds to an APS-related brake fault or a non-fault condition. Formally, the system is fed a structured feature vector based on APS operational data and must estimate the probability of fault occurrence and convert that estimate into an operational binary decision. The challenge is not a trivial one, for four reasons [26]. First, the raw APS dataset is highly imbalanced, with many more negative times than positive failures. Second, the dataset includes missing sentences of the ideas (denoted by symbolic placeholders), requiring explicit preprocessing before the model is trained [27]. Third, predictive outputs on their own are not sufficient in a maintenance setting; the system shall also provide interpretable evidence, an operational health indicator and a working monitoring interface. Fourth, the paper is intended to reflect a real-time decision support environment, so that it must address streaming, visualization, and alerting as opposed to focusing only on model accuracy in its present form, the paper is best interpreted as a binary brake fault monitoring prototype: the models are prepared offline and played back in an online style setting, and presented in a dashboard of combination fault probability, explanation and alerts [28].

1.2. Novelty of the Study

This study does not call for a novel breakthrough learning algorithm; its contribution lies in integrating several practically significant components into a unified brake-fault synthetic surveillance system. The implemented system combines the Random Forest and LSTM models via late-fusion probability averaging [7]; [13] of the current sample. Anomaly detection and Short-term temporal behaviour both play a role in arriving at a final diagnosis estimate. The prediction layer is further extended to include vehicle health scores, risk-based recommendation messages, and SHAP-based explanations for Random Forest branches [3]; [10]. If so, what can this mean for improving the interpretability of a maintenance setting? The monitoring layer is implemented in a Multilingual Streamlit separable Dashboard with Driver View and Technical View modes. Driver View = focus on what can be done. In contrast, Technical View provides model probabilities, warmup behaviour, trend plots, an AI fault timeline, a runtime fault table, a Level-Data SHAP sample preview, and appropriate Optional interpretation. The dashboard is similarly integrated with cooldown-controlled WhatsApp and SMS alerts. Collectively, these things extend the system beyond segregated model inference and toward an integrated predictive seeing maintenance prototype.

2. Related Work

Khan et al. [4] put forward a secure, explainable artificial intelligence system to forecast the occurrence of brake faults for heavy transport vehicles. The research concerns the development of a machine learning-based predictive maintenance system using the Random Forest algorithm to analyse sensor data from heavy vehicle brake systems. The authors employ explainable AI methods, including SHAP (Shapley Additive Explanations), to provide explanations. Model-Agnostic Explanations) to interpret the model predictions and determine the most important features that cause brake failures. The approach makes things more transparent by enabling engineers to understand the knowledge of different sensor signals that contribute to fault prediction. Although the framework offers high predictive ability and interpretability, the work primarily focuses on the prediction model itself. It gives little focus on real-time deployment and integration in an operational monitoring system [8]. Bagherpour et al. [6] and Hou et al. [16] studied intelligent fault diagnostics for air brake systems on heavy commercial vehicles using machine learning techniques. In this study, wheel speed sensor data, brake system parameters, and data for training classification models to identify abnormal brake system behaviour are used. The research compares multiple machine learning algorithms, including Decision Trees and Random Forest classifiers, and evaluates their performance in identifying brake faults. Explanations of experimental results show that ensemble learning methods, such as Random Forests, achieve higher classification accuracy than traditional machine learning models [17].

This work highlights the potential of data-driven methods to improve fault detection; however, the system lacks explainability techniques to interpret model predictions and does not support real-time monitoring or integration with predictive maintenance platforms [9]. Hafeez et al. [7] presented a sequential multivariate fault prediction method for behavioural predictive

maintenance. The authors underline the importance of analysing multivariate time-series sensor data to detect early signs of degradation in vehicle components. The proposed methodology is a sequential machine learning technique that captures relationships among multiple sensor variables over time. By learning patterns from historical operational data, the system can detect anomalies and predict possible failures before they happen. This research emphasises the key manifestations of proactive maintenance strategies and shows that time-dependent sensor data plays an important role in accurate fault prediction. However, the study primarily focuses on prediction accuracy and does not incorporate mechanisms for explainability that would allow engineers to interpret the model's decision-making process [18]. Garcia-Mendez et al. [2] developed an explainable machine learning tool designed for predictive maintenance of transport systems [20]. The study integrates SHAP-based feature interpretation techniques into machine learning models to enhance transparency in AI-driven fault-detection systems. The framework enables maintenance engineers to analyse feature importance and understand how Individual variables affect a system's predictions. The approach increases interpretability, thereby enhancing trust in machine learning models used in safety-critical applications, such as infrastructure, transportation, and vehicle systems [19].

3. Proposed Methodology

3.1. System Overview

The proposed makeshift system follows a pipeline of stages that starts with the ingestion of APS data and ends with the presentation of maintenance-oriented outputs. Raw APS records are preprocessed and machine-preseparated to a model-ready dataset; offline models are trained and calibrated; and the processed sample is then played back in a Dashboard environment that performs hybrid probability estimation, health scoring, SHAP explain, and triggers alerts.

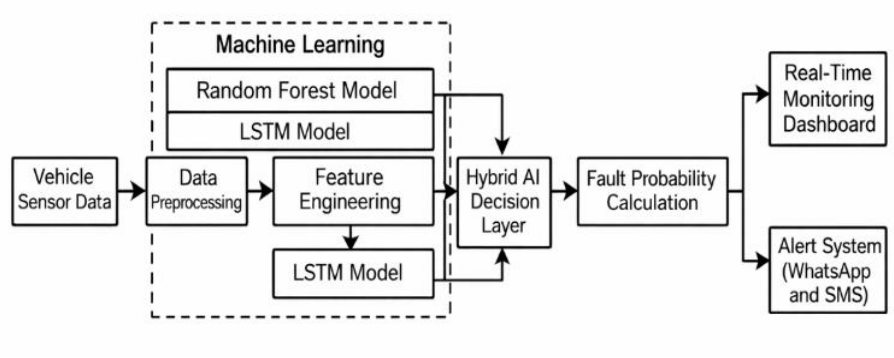

Figure 1: Architecture diagram

As shown in Figure 1, the complete architecture is organised as follows: raw APS data, preprocessing, model training, hybrid inference, monitoring dashboard, explainability module, and alert services.

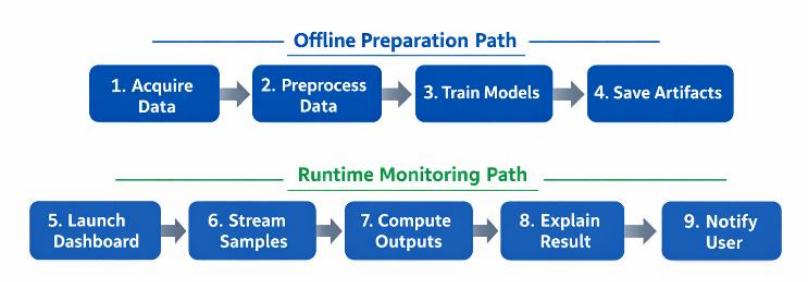

Figure 2: End-to-end workflow

The end-to-end operational workflow is illustrated in Figure 2. It should clearly distinguish between the offline preparation path and the runtime sample-replay path.

3.2. Dataset Description

In the project, the APS Failure at Scania Trucks data are utilised [1]. According to the dataset description, the training split has 60,000 instances: 59,000 are in the negative class, and 1,000 are in the positive class. The test set contains 16,000. It indicates

171 total attributes and says that the missing values are coded as [na]. The processed dataset available for modelling in the current implementation has 118,000 rows and 171 columns. Of these, 170 columns are considered numerical input features, and only one column is used as a binary class label. APS-related failures are positive, and failures not related to APS are negative (Table 1).

Table 1: Raw and processed dataset

Item	Value
Raw training instances	60,000
Raw test instances	16,000
Raw negative class count	59,000
Raw positive class count	1,000
Raw attribute count	171
Missing value token	Na
Processed Dataset shape	118,000 x 171
Model-ready feature count	170
Label representation in processed data	Binary class column

3.3. Data Preprocessing Pipeline

The preprocessing step converts raw APS records to model-ready data with a series of well-defined steps. First, missing values will be replaced with null values. Second, all feature columns are converted to numeric values. Third, missing values for features are imputed at a rate using the KNN Imputer with 5 neighbours. Fourth, feature scaling is performed using the Standard Scaler. Fifth, the class imbalance problem is addressed using SMOTE. The preprocessing stages have been summarised in Table 2.

Table 2: Preprocessing pipelines

Step	Operation	Purpose
1	Load raw APS CSV Files	Read the source operational data
2	Replace Na with null values	Normalize missing values
3	Convert features to numeric	Make attributes model-ready
4	KNN Imputation	Estimate missing sensor values
5	Standard scaling	Normalize feature magnitudes
6	SMOTE balancing	Increase minority-class representation
7	Save processed CSV	Persist a reusable model-ready dataset

3.4. Random Forest Branch

The Random Forest branch is trained on the processed dataset after train-test splitting. In the training notebook, the Split uses test_size=0.2, random_state=42, and class stratification. SMOTE is then applied only to the training split before model fitting. The implemented classifier uses 200 trees, a fixed random seed, and parallel execution through jobs=-1. This branch provides the most directly validated quantitative evidence in the study, including accuracy, class-wise precision, recall, F1-score, and a confusion matrix on a hold-out test set.

3.5. LSTM Branch

The LSTM branch is included in this training pipeline to learn temporal patterns from sliding-window inputs [13]. The length-five sequences are constructed from the processed dataset by grouping consecutive observations, and the class at the end of the window is used as the training label. The resulting network comprises an input layer, an LSTM layer with 64 units, a dense hidden layer with 32 ReLU units, and a sigmoid output layer. Training is performed using the Adam optimiser, binary cross-entropy loss, and early stopping based on the validation loss. In runtime, the latest five samples are stored in a rolling buffer. If fewer than five samples are available, the hybrid output is a Random Forest probability, allowing monitoring to start immediately. Once the buffer is full, the buffered sequence is reshaped into a 5-step tensor, the LSTM probability is computed, and late fusion with the Random Forest output is applied to obtain the final hybrid probability. This runtime design thus aligns with the sequence-based LSTM approach and no single-sample reshaping.

3.6. Hybrid Inference and Health Scoring

At runtime, the Random Forest branch checks the score of each incoming sample. During the first stage of warmup, when there are fewer than 5 consecutive samples, the fault probability is set to the Random Forest's output, allowing monitoring to start immediately. Once the rolling buffer has five samples, the last hybrid fault probability is calculated as an average of the Random Forest probability:

$$P\_fault = (PRF + PLSTM) / 2 \quad (1)$$

where Pfault is the final hybrid fault probability, PRF is the positive class probability generated using the Random Forest branch, and PLSTM is the sigmoid output of the LSTM branch. During the warmup, the system follows Pfault = PRF until the complete 5-sample sequence is available. The binary decision is then generated by thresholding the hybrid probability:

$$y\_hat = 1, \text{if } P\_fault > 0.5; \text{ otherwise}, y\_hat = 0 \quad (2)$$

where y_hat is the predicted one for brake-fault, the dashboard converts the probability further into a vehicle health score:

$$Health = (1 - P\_fault) * 100 \quad (3)$$

where Health is a percentage-style indicator, with higher values indicating a healthier vehicle condition.

3.7. Explainability Module

The explainability layer is implemented using a SHAP Tree explainer trained on a random forest, with SHAP values generated over the processed feature matrix [23]; [24]. The standalone for summary, bar, and single-instance explainability modules support explanations. dashboard, where the current sample can be explained by ranking the top ten features according to absolute SHAP impact and drawing them in a horizontal bar plot. This explainability design enhances transparency but does not yet address the full hybrid fusion; it only addresses the tree-based branch [10]; [11]; [25].

3.8. Monitoring and Alerting Layer

This layer is implemented in Streamlit and is known as runtime monitoring. Once the language is chosen, the processed dataset and the LSTM model are trained, and the Random Forest is loaded into the dashboard. The interface has two operational modes, i.e. Driver View and Technical View. Driver View shows the hybrid fault chance, car health score, health risk band, and a recommendation banner, and simplifies explanatory messages. Also shown on Technical View is the Random Forest probability, the LSTM-Long Short-Term Memory - probability, hybrid warmup status or ready status, animation notes, an active prototype monitor, the probability trend plots, an artificial intelligence fault timeline, a runtime fault table, as well as a SHAP explanation panel (non-compulsory) of the active sample (Table 3).

Table 3: Model-decision configuration

Component	Implemented Configuration
Random Forest	200 estimators, random state=42, n_jobs = -1
LSTM input design	Sequence length = 5 during training
LSTM architecture	Input -> LSTM (64) -> Dense (32, ReLU) -> Dense (1, sigmoid)
LSTM training control	Adam optimiser, binary cross-entropy, early stopping
Hybrid probability	Random Forest probability during warmup; arithmetic mean of RF and LSTM probabilities after 5 samples
Binary decision threshold	0.5
Health score	(1 - P_fault) * 100
Alert cooldown	5 samples

The replay of processed samples is sequenced to simulate real-time monitoring. Each time the Random Probability of forest is calculated, five sample buffers are rolled, which yields the final hybrid probability. A binary brake malfunction label and the vehicle's health score are obtained, after which the output is clustered into low-risk, warning, and critical bands based on 0.3 and 0.6 thresholds. WhatsApp and SMS notifications are also included, along with a cooldown of 5 samples to prevent a fault-positive prediction from leading to a flood of messages—banners containing system status and brake status, technical notes,

and LSTM readiness. Messages are also included in the dashboard to make interpretation easier for both operational People who use the program/developers, as well as technical reviewers (Table 4).

Table 4: System-module view of framework

Module	Implemented Role	Current Scope
Prediction engine	Hybrid fault probability, rolling sequence buffering	Backend helper with RF warmup and RF-LSTM late fusion
Dashboard monitor	Real-time style	Streamlit UI with Driver view, technical view, and multilingual support
Explainability layer	SHAP-based feature	Random-Forest-interpretation
Alerting service	WhatsApp and SMS Alerts	Environment-configured runtime service
Streaming simulator	Sequential replay	CSV-based demo stream

4. Experimental Setup

4.1. Experimental Design Overview

The study is based on the processed APS failure dataset, obtained by transforming the raw Scania records for a model-ready representation. The final modelling matrix has 118,000 samples and 171 columns, of which 170 are numerical input features, and 1 is the binary class label. The overall evaluation protocol is split into two stages. The first stage,s an offline training and testing phase, is used to estimate predictive performance under fixed data partitions. The second stage is a replay-based monitoring phase that provides hybrid inference, health scoring, explainability, and alert behaviour in a dashboard environment.

4.2. Dataset Split and Preprocessing Configuration

This is done by cleaning the APS records before training and regularising them using a multi-player preprocessing process. The raw data contains missing values that are coded as null. ALL sensor attributes are transformed into numeric expressions and entries. Fragmentary values are predicted using k-nearest neighbours. imputation with k = 5. Feature magnitudes are then normalised, and the class imbalance is addressed using synthetic minority oversampling. For the tree-based experiment, the dataset will be split into training and test sets at 80:20, with a fixed Split. random seed=42, retaining proportions of classes in the hold-out partition. The Split is reported to have 94,400 training. samples and 23,600 testing samples. The resulting class distributions refer to a balanced training set equal to 47200. number of samples per class, as well as a balanced test set of hold-out with 11,800 samples. samples per class. The design is such that the model is evaluated. is made on a fixed test partition instead of the same one. data report used to fit parameters.

4.3. Random Forest Training Protocol

The Random Forest branch is considered the primary quantitative branch. In the current investigation, it was used as a benchmark because it is the only branch for which a complete decision tree was used for classifier training, and a fixed seed was employed. Parallel execution allows for efficient fitting. After the model is tested using the hold-out test partition. with binary-classification measures, such as accuracy, precision, recall, F1-score and confusion matrix. analysis. Along with tessellated anticipations, the model gives out class-probability approximations to be utilised subsequently in the hybrid. inference stage. Because this branch has a clearly defined training protocol and maintains evaluation outputs, a property that it shares with formal training protocols. delivers the best quantitative data in the present. implementation.

4.4. LSTM Sequence Training Protocol

The sequential branch is anchored in a Long Short-Term. A memory network which captures the temporal behaviour. between individual sensor observation windows. To construct the processed feature matrix, it is converted into an input: repeating length-5 sequences and the class designation. To every sequence is counted in the last time step of the. window. The resulting sequence tensor is divided into size partitions with an 80:20 split, using the same fixed random seed for training and testing. The network architecture comprises. Variable: an input layer, an LSTM layer, which has 64 hidden units. A 32-neuron-thick hidden layer with ReLU activation, and a binary probability estimation is done by a sigmoid output layer. The Adam optimiser and binary cross-entropy are used to perform training. Loss, tracking accuracy, and a batch size of 128. and a maximum of 10 epochs. With a forbearance untimely fashioned, stopping. The underfitting is minimised with three epochs. best validation-loss state.

4.5. Runtime Monitoring and Demonstration Configuration

The real-time element of the system is verified by replaying the processed samples in a live vehicle telemetry manner. The tracking will occur during the publicity. Scan displays are saved in a linear format. At runtime, the monitoring interface continuously delivers samples from the processed dataset. It calculates the probability of the Random Forest, the probability of the LSTM when the five-sample sequence buffer is prepared, the probability of the hybrid fault, the ultimately discrete choice, and the resulting vehicle health score for each observation. The replay window may be set to 10-150 samples, and the inter-sample delay may be set to 0-2.0 s to make it appear as though it is running at various speeds. The sidebar also provides options for the dashboard view, turning SHAP on or off in Technical View, setting WhatsApp and SMS alerts, changing the language, and starting the streams. The interface is also updated with recommendation messages and explanatory notes, status indicators, probability trends, timeline outputs, and a sample-level preview panel in addition to probability output. The relative interface of the runtime configurations is displayed in Figure 3.

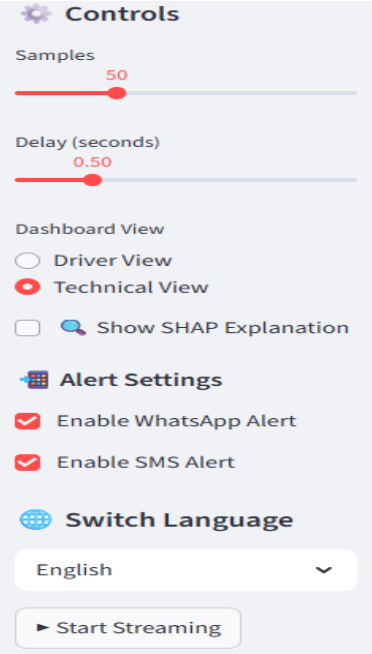

Figure 3: Runtime control panel

As shown in Figure 3, the runtime interface exposes the main demonstration controls directly to the user. This architecture also affirms repeatable purchase-based evaluation. with streaming delay, dashboard, by enabling the sample window. choice of view, SHAP visibility, alert activation, and language. option and streaming to be changed without modification. The conceptual hypothesis process.

4.6. Evaluation Protocol and Reporting Scope

The current research is planned to be analysed with the purposeful separation of quantitative evidence and qualitative system. behavior. The degree of quantitative analysis is now limited to the. Random Forest holds out well, with high classification accuracy. There are several confusion matrices. Instead, the one that is used to assess the. The principal coherence of the general structure, which entails similar probability changes, similar behaviour in health scores, explainable outputs, and alert criteria, is the runtime dashboard. In line with the above, the dashboard is not intended to replace the system, as it is a system-level demonstration environment and a full benchmark study. This is one important difference, in particular, because the hybrid approach to inference has not yet been maintained. A passionate comparative analysis based on replay. processed data rather than real-time on-board acquisition underpins the stream's throughput.

5. Results and Performance Analysis

5.1. Quantitative Random Forest Results

The branch Random Forest scored well on the hold-out test set, achieving 99.57% accuracy across 23,600 test samples. The corresponding confusion-matrix counts are TN = 11703, FP = 97, FN = 4, and TP = 11796, as shown in Figure 4. The class-

wise precision, recall, and F1-score are all nearly unity, indicating high discrimination between fault and non-fault categories under the adopted evaluation setup (Table 5).

Table 5: Hold-out Evaluation Metrics

Metric	Value
Testing samples	23,600
Accuracy	99.57%
Class 0 precision	1.00
Class 0 recall	0.99
Class 0 F1-score	1.00
Class 1 precision	0.99
Class 1 recall	1.00
Class 1 F1-score	1.00
Macro average F1-score	1.00
Weighted average F1-score	1.00
Confusion matrix	[[11703, 97], [4, 11796]]

Figure 4 of the confusion matrix shows that the Random Forest branch is highly discriminating on the held-out test set. partition, having a few false positives and false negatives in relation to the total size of the test.

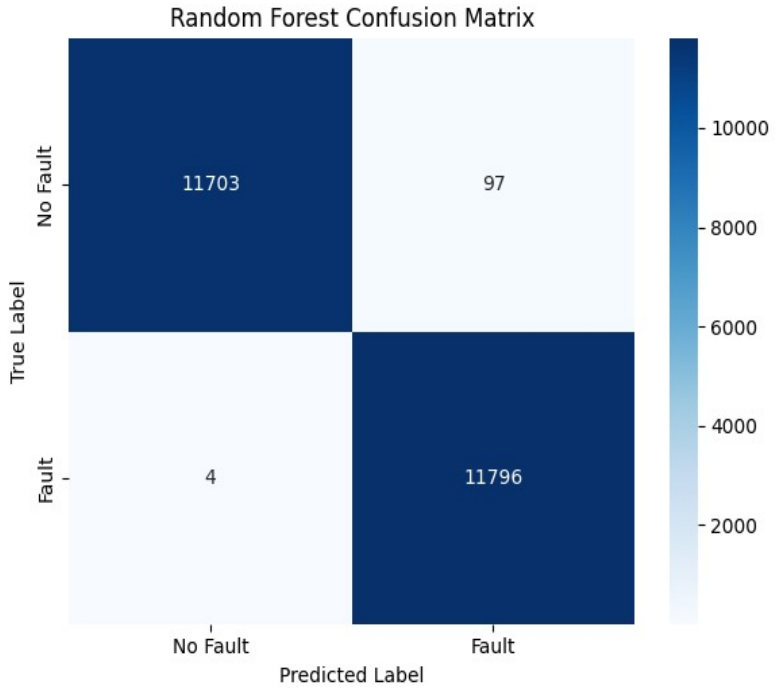

Figure 4: Random forest confusion matrix

5.2. System-Level Monitoring Results

The system-level capabilities, aside from classifier output, are few and displayed on the dashboard. The system is displayed on the main Technical View screen. Statuses of random forest, brake, hybrid fault, status. filter, wireless, vehicle health score, commentaries on recommendations, technical notes, LSTM preparations, status and the presence of an indicator of the active sample sensor. The probability trend plot of faults provides the capabilities to compare the hybrid, Random Forest and Learning behaviour against the replayed samples. One of the tables used to supplement the plots is the runtime fault timeline table, which provides model probabilities, sample index, vehicle health score, and a status for each sequence of replayed samples. Besides that, a dashboard-supported vehicle-health trend panel is available with an optional SHAP account for an active sample. Technical View is enabled. All of these outputs improve the level of operational and technical interpretability.

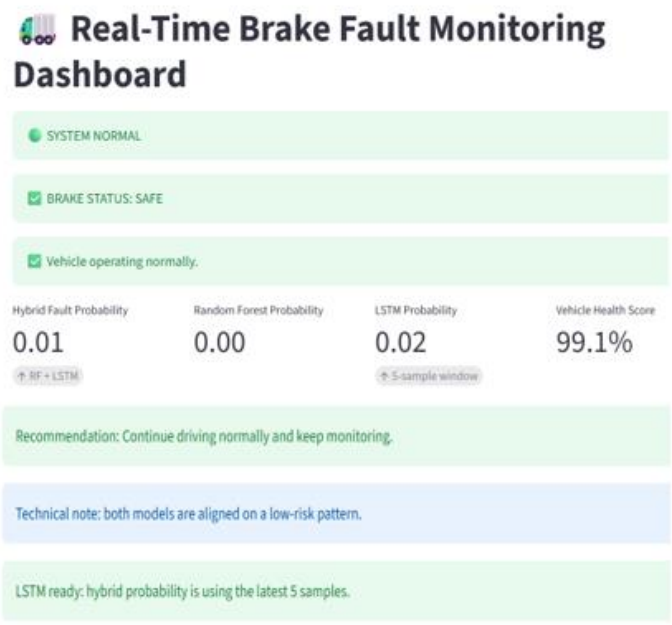

Figure 5: Dashboard overview

The primary dashboard state, depicted in Figure 5, shows the system in operation, with predictive output and operational interpretation illustrated simultaneously. By combining hybrid probability, vehicle health rating, braking, and a single view, the interface lets the system owner see the overall system status. A more easily interpreted diagnostic outcome in a maintenance-oriented context.

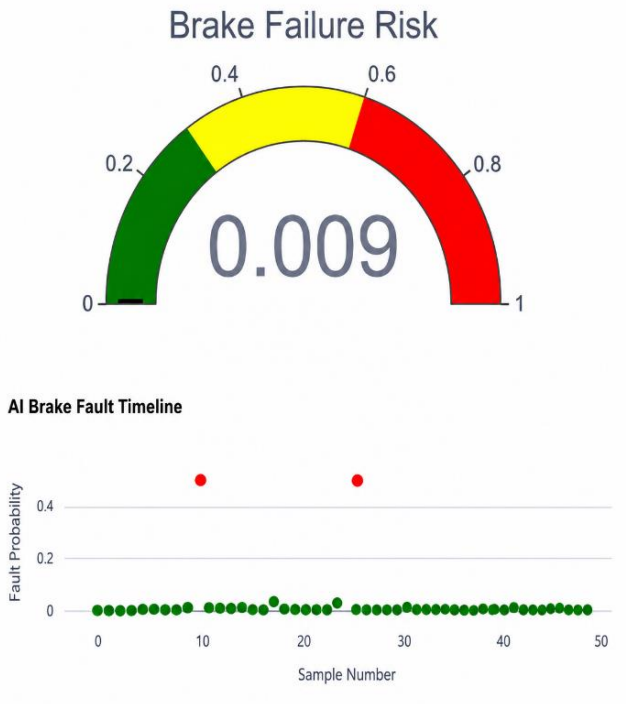

Figure 6: Brake-failure risk meter and AI fault timeline

The brake-failure risk meter, as illustrated in Figure 6, represents the current level of severity, and the AI fault timeline, respectively, illustrates the changing status at the sample level throughout the replay sequence. This combined view helps the operator see not only the current risk state but also the frequency and probability of flawed events throughout history.

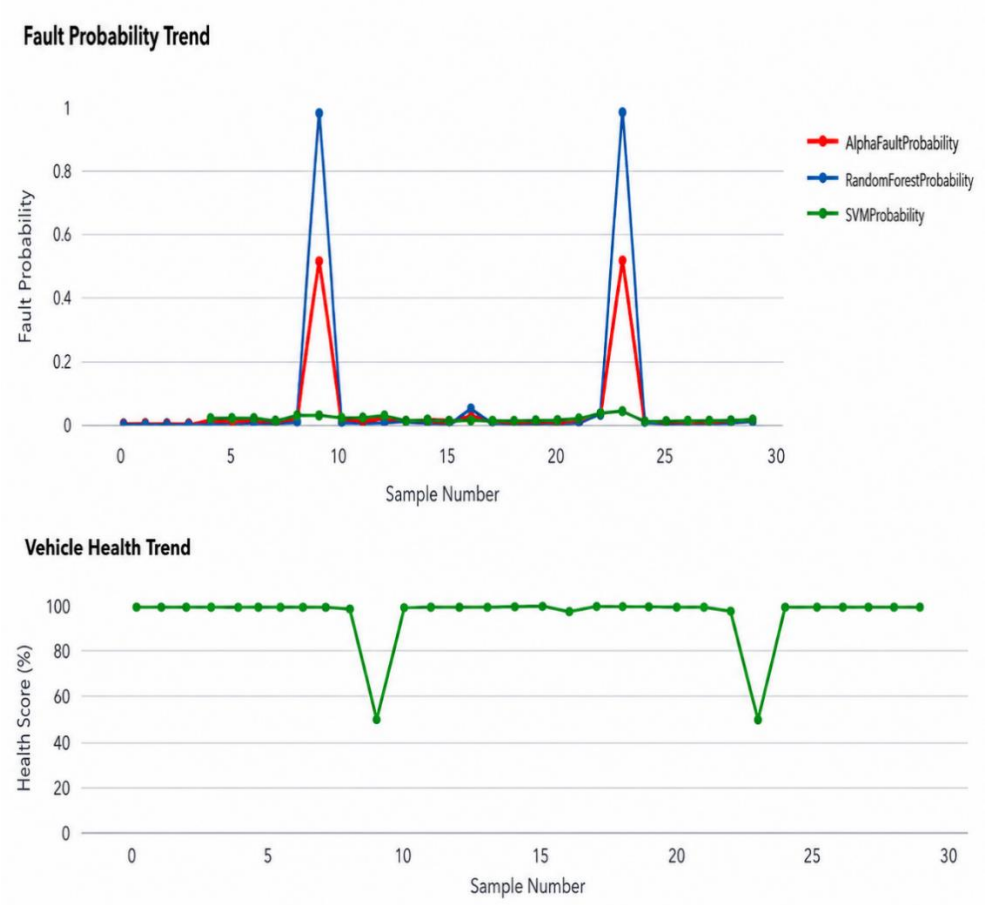

Figure 7: Trend fault and vehicle health

Across replayed samples, a longitudinal perspective on runtime behaviour is provided in Figure 7, plotting both the hybrid fault probability and the derived vehicle health score. Peaks in fault probability are followed by parallel declines in health score, as anticipated, since the health indicator is calculated as the inverse of the probability of a hybrid fault (Table 6).

Table 6: Runtime fault summary table

Fault Timeline						
Sample		**Random Forest Probability**	**LSTM Probability**	**Hybrid Fault Probability**	**Vehicle Health Score**	**Status**
5	5	0	0.017	0.009	99.1	Safe
6	6	0	0.022	0.006	99.4	Safe
7	7	0	0.011	0.005	99.5	Safe
8	8	0.005	0.027	0.005	98.4	Safe
9	9	1	0.025	0.533	48.7	Critical
10	10	0	0.026	0.023	98.7	Safe

The graphical is supplemented by a tabular summary in Figure 8 and dashboard views that list the precise sample index, the estimated probability of faults, the health rating, and the safety status. The repression enhances auditability since it permits. The replay playout should be inspected sample-by-sample instead. but not by anything but visual fashions. The local SHAP explanation obtained in the dashboard is shown in Figure 8. Weak and strong feature contributions provide insight into which sensor variables are driving the Random Forest branch towards fault or non-fault behaviour, and enhance understanding of the monitoring workflow [3]; [23]; [24]. From a monitoring perspective, the system provides a coherent user-facing workflow: it estimates hybrid probability, translates that estimate into a health score, assigns a safety state, and links model output to a concise explanation.

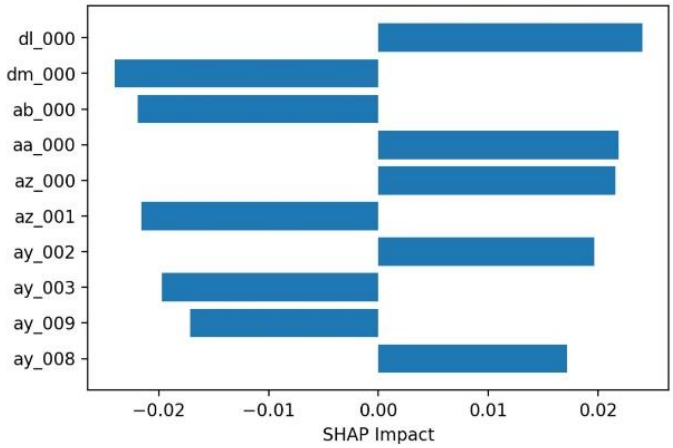

Figure 8: SHAP-based local explanation view

5.3. Alerting and Runtime Behaviour

The dashboard incorporates WhatsApp and SMS notifications and sends five sample cooldowns, then retransmits actions following a positive prediction of a fault. Each alert channel can be turned on and off in the sidebar, making the system appropriate for both controlled protests and surveillance-style operations. The frequency notification workflow is an enhancement to Visual Supervisor, transforming recurring high-risk products into controlled, user-facing abstract news rather than clogging messages. In this way, the dashboard not only illustrates brake-fault risk but also provides actionable communication in the event a fault condition is detected.

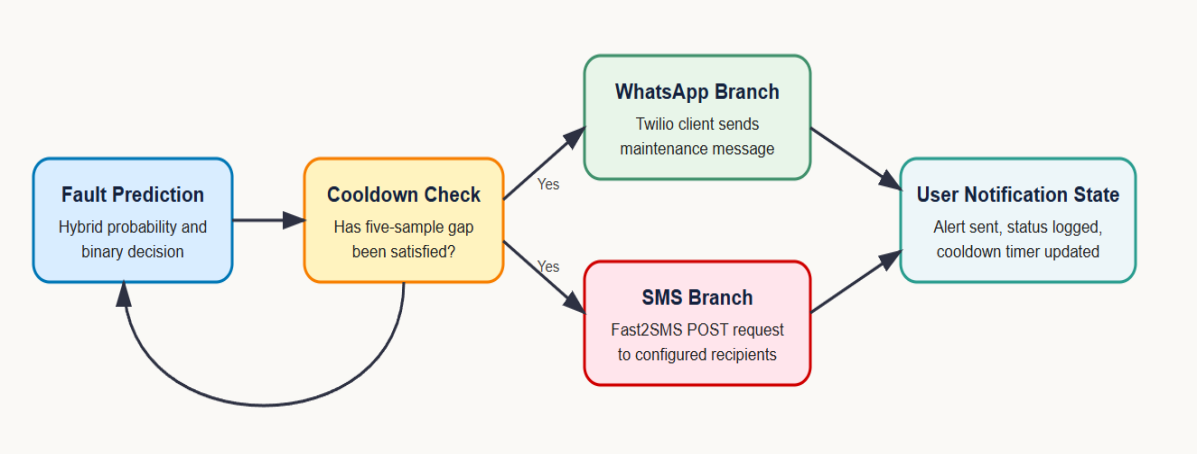

Figure 9: Cooldown-controlled alert-routing workflow

The alert-routing process, as illustrated in Figure 9, explains how a high-risk state is converted into an outbound notification behaviour. The inclusion of a cooldown control is particularly important, as it minimises repeated bursts of messages that occur when faults persist across multiple replayed samples.

Figure 10: WhatsApp alert generated during the brake-faults detection

Figure 10 provides direct qualitative observations that the monitoring pipeline can generate a user-facing WhatsApp notification when a brake fault is detected.

5.4. Evidence Boundaries

At this point, the paper does not entail a separate hybrid model, such as the one available in the notebook for the Random Forest. It also does not include ROC curves, AUC values or ablation results of the hybrid. setup. That is why this paper does not claim that the hybrid model has already been proven to be better.

5.5. Discussion

The paper demonstrates several practical strengths. First, it combines data preprocessing, model training and runtime, monitoring, Explainability, and alerting into a single, consistent workflow [5]; [21]. Second, it was the exploration of probability rather than simple hard labelling that would facilitate downstream interpretation using risk banks and health scoring. Third, the interface is oriented toward non-specialist evaluators, which is convenient for both academic presentations and fieldwork. Simultaneously, the current implementation has a few drawbacks; in particular, the monitoring flow is still performed only via CSV replay rather than live sensor ingestion. SHAP-based interpretation can now explain the Random Forest branch and, therefore, is not a full description of our fused RF-LSTM [11]; [14]; [25]. The screened set of streaming scripts is, yet again, Random Forest-only. Still, the runtime combination streamlining is currently also paired with a rolling five-sample sequence buffer and a separate benchmark report for the deployed hybrid model, such as the Random Forest holdout metrics. Such a restriction positions the current system as a powerful prototype rather than a 100-per-cent-validated, deployment-ready solution [22].

6. Conclusion

In this paper, a real-time explainable hybrid artificial intelligence framework for predictive brake issue diagnostics in large transport trucks was proposed. The suggested system integrates Random Forest and Long Short-Term Memory (LSTM) models to efficiently fuse instantaneous fault evaluation with temporal pattern analysis, enabling precise and prompt identification of a braking system defect. The Random Forest model yields reliable predictions during the first warmup phase, while the late-fusion RF-LSTM architecture improves diagnostic performance when sufficient consecutive sensor data are available. A major feature of the system is its focus on explainability through SHAP-based interpretation, enabling drivers, maintenance engineers, and fleet operators to understand the reasoning behind each forecast. This transparency builds trust in AI-augmented decision-making, especially in safety-critical transportation contexts where accountability and dependability are paramount.

Fault probabilities are converted into an intuitive vehicle health score that further enables condition assessment and maintenance planning. The created multilingual Streamlit dashboard enhances accessibility by providing separate Driver View and Technical View interfaces, enabling users with varying levels of competence to successfully monitor vehicle conditions. Other capabilities, such as trend visualisation, event timelines, maintenance recommendations, and WhatsApp and SMS warnings based on cooldowns, help enable timely communication and proactive intervention before severe failures occur. Overall, the suggested framework shows the potential of explainable hybrid AI to enhance vehicle safety, reduce unexpected breakdowns, reduce maintenance costs, and enable data-driven fleet management. Future work might investigate integrating additional vehicle subsystems, deploying the framework in large-scale fleet environments, and applying federated learning and edge computing technologies further to enhance scalability, privacy, and real-time operational performance.

Acknowledgment: The authors sincerely acknowledge SRM Institute of Science and Technology at Ramapuram for providing the academic support and resources necessary to carry out this research work.

Data Availability Statement: The findings and supporting data associated with this study are maintained by the authors and may be shared with qualified researchers upon justified request.

Funding Statement: No external sponsorship, grant, or institutional funding was utilised in conducting this research. All work was completed using the authors' own resources.

Conflicts of Interest Statement: The authors affirm that there are no financial, professional, or personal relationships that could have influenced the outcomes or interpretation of this research.

Ethics and Consent Statement: The research process was conducted in accordance with ethical research standards. Participation was voluntary, and all contributors provided their consent before inclusion in the study.

References

1. Scania CV AB, "APS Failure at Scania Trucks," Dataset, *UCI Machine Learning Repository*, 2016. [Accessed by 12/03/2025].
2. S. Garcia-Mendez, F. de Arriba-P erez, F. Leal, B. Veloso, B. Malheiro, and J. C. Burguillo-Rial, "An explainable machine learning framework for railway predictive maintenance using data streams from the metro operator of Portugal," *Scientific Reports*, vol. 15, no. 7, p. 27495, 2025.
3. V. Parate, R. K. Punithavel, S. Agrawal, and A. Jaiswal, "Explainable AI in fleet predictive maintenance: Using SHAP and LIME for transparency and regulatory alignment," *International Journal of Applied Mathematics*, vol. 38, no. 6s, pp. 246–264, 2025.
4. M. A. Khan, M. Khan, S. Khan, A. Farooq, W. Li, Z. Ahmad, and F. Ahmad, "Unveiling Brake Faults in Heavy Vehicles with Explainable AI," in Proceedings of Data Analytics and Management, Lecture Notes in Networks and Systems, *Springer*, Singapore, 2025.
5. Y. Mahale, S. Kolhar, and A. S. More, "A comprehensive review on artificial intelligence driven predictive maintenance in vehicles: technologies, challenges and future research directions," *Discover Applied Sciences*, vol. 7, no. 3, p. 243, 2025.
6. S. Bagherpour, M. Rezaeian Akbarzadeh, and S. Mouloodi, "Sensitivity Analysis of Heavy Vehicle Air Brake System to Air Leakage," *SAE International Journal of Commercial Vehicles*, vol. 14, no. 1, pp. 69–83, 2021.
7. A. B. Hafeez, E. Alonso, and A. Ter-Sarkisov, "Towards Sequential Multivariate Fault Prediction for Vehicular Predictive Maintenance," in Proc. *20th IEEE International Conference on Machine Learning and Applications (ICMLA)*, Pasadena, California, United States of America, 2021.
8. B. Tormos, B. Pla, R. Sanchez-Marquez, and J. L. Carballo, "Explainable AI Using On-Board Diagnostics Data for Urban Buses Maintenance Management: A Study Case," *Information*, vol. 16, no. 2, p. 74, 2025.
9. Y. F. Li, H. Wang, and M. Sun, "ChatGPT-like large-scale foundation models for prognostics and health management: A survey and roadmaps," *Reliability Engineering and System Safety*, vol. 243, no. 3, p. 109850, 2024.
10. B. Steurtewagen and D. Van den Poel, "Adding interpretability to predictive maintenance by machine learning on sensor data," *Computers and Chemical Engineering*, vol. 152, no. 9, p. 107381, 2021.
11. J. Tang, Y. Liu, X. Huang, Z. Jiang, and F. Wu, "Explainable AI for post-hoc and pseudo-post-hoc predictive maintenance of governor valve actuators," *Scientific Reports*, vol. 15, no. 11, p. 39504, 2025.
12. P. Singh and L. Behera, "Prediction of Failure in Scania Truck Due to Air Pressure System Failure," in Distributed Computing and Intelligent Technology, Lecture Notes in Computer Science, *Springer*, Cham, Switzerland, 2024.
13. M. E. Mumcuoglu, S. M. Farea, M. Unel, S. Mise, S. Unsal, E. Cevik, M. Yilmaz, and K. Koprubasi, "Detecting APS failures using LSTM-AE and anomaly transformer enhanced with human expert analysis," *Engineering Failure Analysis*, vol. 165, no. 11, p. 108811, 2024.
14. S. M. Farea, M. E. Mumcuoglu, and M. Unel, "An Explainable AI approach for detecting failures in air pressure systems," *Engineering Failure Analysis*, vol. 173, no. 5, p. 109441, 2025.
15. M. S. Rahman and V. Sumathy, "Forecasting failure-prone air pressure systems (FFAPS) in vehicles using machine learning," *Automatika*, vol. 65, no. 1, pp. 1–13, 2024.
16. Z. Hou, C. K. M. Lee, Y. Lv, and K. L. Keung, "Fault detection and diagnosis of air brake system: A systematic review," *Journal of Manufacturing Systems*, vol. 71, no. 12, pp. 34 –58, 2023.
17. Y. Fan, S. Nowaczyk, and T. Rognvaldsson, "Evaluation of Self-Organized Approach for Predicting Compressor Faults in a City Bus Fleet," *Procedia Computer Science*, vol. 53, no. 8, pp. 447–456, 2015.
18. C. Panda and T. R. Singh, "ML-based vehicle downtime reduction: A case of air compressor failure detection," *Engineering Applications of Artificial Intelligence*, vol. 122, no. 6, p. 106031, 2023.
19. A. Theissler, J. Perez-Velazquez, M. Kettelgerdes, and G. Elger, "Predictive maintenance enabled by machine learning: Use cases and challenges in the automotive industry," *Reliability Engineering and System Safety*, vol. 215, no. 11, p. 107864, 2021.
20. Y. Fan, S. Nowaczyk, and T. Rögnvaldsson, "Incorporating Expert Knowledge into a Self-Organized Approach for Predicting Compressor Faults in a City Bus Fleet," in *Proc. 13th Scandinavian Conference on Artificial Intelligence (SCAI 2015)*, Halmstad, Sweden, 2015.
21. P. P. Anand, G. Jayanth, K. S. Rao, P. Deepika, M. Faisal, and M. Mokdad, "Utilising hybrid machine learning to identify anomalous multivariate time-series in geotechnical engineering," *AVE Trends in Intelligent Computing Systems*, vol. 1, no. 1, pp. 32–41, 2024.
22. M. N. Hussain, M. M. Rahman, and D. Ramasamy, "Artificial Intelligence-Driven Vehicle Fault Diagnosis to Revolutionize Automotive Maintenance: A Review," *Computer Modeling in Engineering and Sciences*, vol. 141, no. 2, pp. 951–996, 2024.
23. B. Licaj and D. A. Karras, "Integral smart vehicle automation framework using IoT, modern software, and infrastructure," *AVE Trends in Intelligent Computing Systems*, vol. 1, no. 3, pp. 128–141, 2024.

24. M. T. Ribeiro, S. Singh, and C. Guestrin, ""Why Should I Trust You?": Explaining the Predictions of Any Classifier," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco, California, United States of America, 2016.
25. A. Singaravelan, S. Sujitha, S. Gopikha, T. Karthikeyan, and R. Panakkal, "Smart transportation in VANET using a hybrid reptile search algorithm-based infrastructure model with VANET," *AVE Trends in Intelligent Computer Letters*, vol. 1, no. 3, pp. 104–118, 2025.
26. Y. Fan, S. Nowaczyk, T. Rognvaldsson, and E. A. Antonelo, "Predicting Air Compressor Failures with Echo State Networks," *PHM Society European Conference*, vol. 3, no. 1, pp. 1–11, 2016.
27. S. Gawde, S. Patil, S. Kumar, P. Kamat, K. Kotecha, and S. Alfarhood, "Explainable Predictive Maintenance of Rotating Machines Using LIME, SHAP, PDP, ICE," *IEEE Access*, vol. 12, no. 2, pp. 29345–29361, 2024.
28. A. Ciobotaru, C. Corches, D. Gota, and L. Miclea, "An Explainable Deep Learning-Based Predictive Maintenance Solution for Air Compressor Condition Monitoring," *Sensors*, vol. 25, no. 18, p. 5797, 2025.