Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Edge-AI Predictive Maintenance Based on Industrial IoT and Digital Twin System

Megha Patil
Symbiosis Skills and Professional University
Yogesh Bhirud
Symbiosis Skills and Professional University
Dhanashree Barbole
dhanashree.barbole@sitpune.edu.in

Symbiosis Institute of Technology, Symbiosis International (Deemed University)

Research Article

Keywords: Edge intelligence architecture, predictive maintenance optimization, industrial IoT integration, digital twin-enabled monitoring, reinforcement learning-based maintenance, low-latency edge computing systems.

Posted Date: August 3rd, 2026

DOI: https://doi.org/10.21203/rs.3.rs-10331025/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Edge-AI Predictive Maintenance Based on Industrial IoT and Digital Twin System

Megha Patil[1], Yogesh Bhirud[2], Dhanashree Barbole[3]

[1,2] School of Mechatronics Engineering, Symbiosis Skills and Professional University, Pune, India
[3] Department of Artificial Intelligence and Machine Learning, Symbiosis Institute of Technology, Symbiosis International (Deemed University), Pune, India
megha.patil@sspu.ac.in[1], yogeshabhirud5@gmail.com[2],dhanashree.barbole@sitpune.edu.in[3]
*Corresponding Author: Megha Patil (megha.patil@sspu.ac.in), Dhanashree Barbole (dhanashree.barbole@sitpune.edu.in)

Abstract

A breakdown of a critical machine in the middle of production is a big deal. The line stops, schedules are blown, and now the fix is being done under duress, with even less time to do it right. Supplies that might have been ordered ahead of time now have to be completed on the same day. Labor that is planned for other work is diverted. The ultimate cost of a single failure is almost always several multiples of what early detection and prompt maintenance would have entailed. One of the problems is that many traditional systems are not smart enough at the hardware level to investigate early signs of impending failure. We observe that the current solutions suffer from three major drawbacks: a) They heavily depend on centralized servers for (sometimes real-time) inference, resulting in non-negligible latency, b) They only weakly integrate digital twin models with real-time data streams, and c) They have limited capability to adapt given evolving operational contexts. To this end, a layered predictive maintenance architecture that employs edge AI processing for Industrial IoT data collection and Digital Twin modeling was proposed to fill these gaps. In parallel, the data from various sensors (vibration, temperature, acoustic, and current signals) are gathered by the system, processed on the edge gateway, and fed to an ensemble of machine learning and deep learning models (Random Forest, SVM, LSTM, CNN-LSTM) together with a reinforcement learning scheme. The Digital Twin interacts in real time with the edge layer in a continuous feedback loop to simulate degradation patterns and dynamically tune maintenance decisions. The result of the experiment with the confusion matrix analysis and the cross-validation shows the integrated approach achieves an accuracy of detection of 97.43%. Furthermore, false negatives were decreased by about 33.33% compared to the single-sensor approaches, and the total downtime was reduced by 66.67% when compared with traditional threshold- based methods. One interesting result is that doing inference on the edge itself (even on a low-cost device like Raspberry Pi) can still reach a good level of accuracy, with the latency being reduced to less than 13 ms.

Keywords

Edge intelligence architecture, predictive maintenance optimization, industrial IoT integration, digital twin-enabled monitoring, reinforcement learning-based maintenance, low-latency edge computing systems.

1. INTRODUCTION

Walking through a manufacturing plant today, it is not unusual to see sensors attached to motors, pumps, and conveyor drives feeding data into the monitoring systems. Infrastructure for condition-based maintenance exists in many locations. The slow rise in bearing temperature, gradual increase in vibration amplitude, and subtle change in current draw that appear weeks before a failure are lost in systems that only consider one reading at a time. Small and medium manufacturing units are especially affected because they rarely have dedicated reliability engineers who continuously monitor the data. This study was based on this observation. Along with the sensor-to-decision gap, most predictive maintenance systems also suffer from an architectural issue in the way they process data. Centralized cloud processing brings response latencies that can be significant for rapidly evolving mechanical systems. There is not much time between early indications of a fault on a rotating shaft spinning at high speed and catastrophic damage. The added latency of sending data to a remote server, waiting for inference, and getting an alert back could cost more than the bandwidth savings. Conducting inference locally at the edge, near the equipment, eliminates this delay and reliance on network availability, which can be spotty on the floor of a factory. The third motivation was the fact that digital twin models and data-driven classifiers are generally designed separately. The twin simulates the machine. The machine learning model classifies the faults. A classifier that operates without any consideration for how the machine is physically expected to behave may yield results that are difficult for operators to trust. Tying together the twin is updated to current sensor states, and the classifier results are fed back into the simulation, yielding a more complete picture and grounding the diagnostic result in a physical model, as opposed to simply a statistical one. The remainder of this paper is organized as follows. Section 2 reviews the related literature. Section 3 explains the proposed architecture and method. Section 4 presents the experimental results and analysis. The later sections cover the limitations and conclusions.

2. RELATED WORK

The drive towards Industry 4.0 has significantly increased the penetration of sensor-based monitoring in the manufacturing industry. The industrial IoT infrastructure enables the non-stop monitoring of vibration, temperature, acoustic and electrical signals of operational machines, which paves the way for condition-aware maintenance strategies [1],[2]. The ability to detect faults sooner and more accurately with machine learning and deep learning classifiers has been demonstrated by several studies [3]-[7]. However, the majority of these systems rely on cloud servers to process data, which results in latency, bandwidth constraints, and vulnerability to cyberattacks [8], [9]. It is also challenging to perform real-time anomaly response when high-frequency sensor streams must be propagated through several network layers before such inference [10]. In cloud-only pipelines, two additional problems are apparent. They tend to lose synchronization with Digital Twin representations, thereby restricting the system's potential to rapidly update virtual models under varying load conditions [11], [12]. The literature has identified four specific technical constraints. First, centralized analytics pipelines introduce delays in fault classification for time-critical applications [13]. Edge hardware such as microcontrollers and single-board computers has limited memory and computing capabilities, which makes running large neural networks locally on edge devices challenging [14]. Third, the synchronization of the Digital Twin states may be interrupted if the physical and virtual update periods become misaligned [15]. Fourth, minimal cross-layer coordination exists among sensing, inference, and maintenance scheduling, which leads to decreased system-wide efficiency [16],[17]. Recent studies have demonstrated promising results in AI-based fault diagnosis; however, the cooperation across these architectural layers is still disjointed [18]-[21]. The fact that the majority of disseminated studies are focused either on edge acceleration or on Digital Twin modeling separately and do not generate a unified feedback loop between those two [22], [23]. Cross-comparisons between classical ML, DL, and RL in the same application and deployment scenario are scarce [24]. Three aspects are not well served by existing frameworks: adaptive scheduling strategies for maintenance interventions, real-time coordination between edge inference outputs and virtual simulation states, and accurate

modeling of the latency-energy trade-off for green execution on resource-limited devices [25]-[26]. This work fills all three gaps directly.

3. MATERIALS AND METHODS

The entire system was structured in four layers with well-defined responsibilities; intelligence can be distributed at the appropriate level without causing excessive data traffic. Splitting up the sensing, edge inference, digital twin simulation, and cloud storage was a conscious decision; it was found that integrating these duties into fewer layers would hinder tuning each element in isolation. Figure 1 shows this hierarchy.

The sensor layer is a physical interface of observation. Vibration, surface temperature, acoustic emission, and motor current were collected from the industry asset. Simple signal conditioning, noise filtering, normalization, and timestamp tagging are done before passing the data upstream. No decision logic is executed at this layer; it is only used to capture and clean incoming signals before transmitting them out.

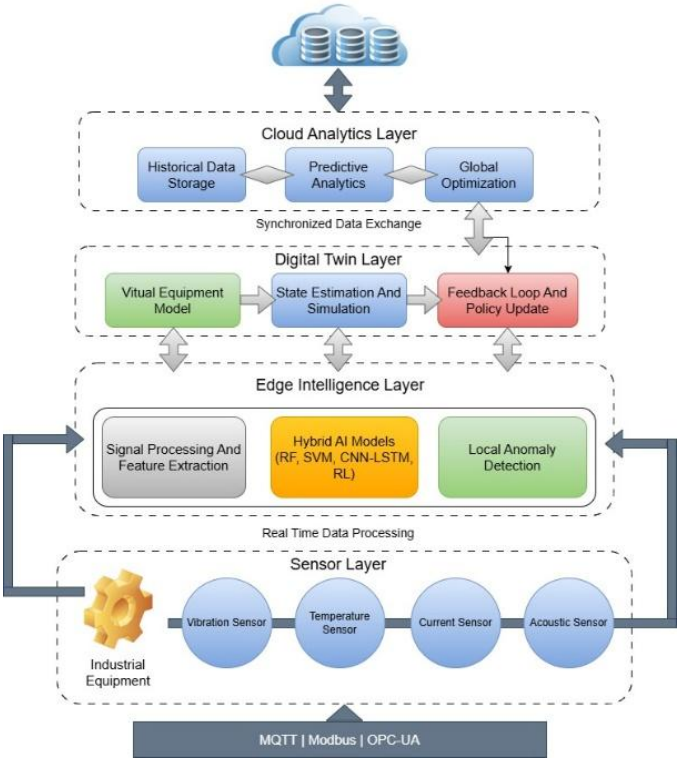


Fig. 1 Architecture of Digital Twin in the cloud-edge intelligence paradigm

The 100 Hz sampling rate was chosen after the evaluation showed that higher rates introduce a heavy load on the Raspberry Pi and do not significantly improve the quality of the captured vibration signals. When the first tests were running, the Pi's performance was suffering noticeably when the operating frequency was above 150 Hz (i.e., 150Hz+ was the "cutting-off frequency" at that time).

Now, in the edge layer, the majority of your real-time processing is done. Feature vectors derived from the raw signals are input into AI models, which are specifically designed for constrained hardware. Since the edge computes anomaly probability scores, health indices, and early warning flags on raw data in near real time at the node itself instead of sending raw data streams continuously to a remote server, Stability Enhancing Water Release (SEWR) is enabled. The inference outputs are mapped to realistic maintenance alerts by a small-footprint rule engine. This

protocol can significantly reduce and shrink the communication overhead while keeping low delay for detecting node faults. The Pi 4B executed inference surprisingly well, and all models ran in less than 13 ms, even when looped. On the simulation side, the Digital Twin was not a fixed model; it evolved its state with each new data received from sensors or new inference result. The twin can model how a developing fault is likely to evolve under current operating conditions and assess alternative maintenance decisions before deciding on one. As opposed to simple visualization dashboards, this tier also contributes to the feedback loop, feeding updated thresholds and policy recommendations to edge nodes. When testing the twin was found that its capability to identify divergence between predicted and actual degradation paths allowed it to identify faults in earlier stages compared with the classifiers only.

The long-term storage, large-scale analysis of trends, and retraining of the model are left to the cloud layer. Heavy processing workloads that do not require low latency are offloaded here to free up edge hardware resources. Retraining was done in the cloud, where computing wasn't a limit, but the result of that retraining was only useful when it came back to the edge. Similarly, after each retraining cycle, the updated weights were validated on a test set and then packaged and pushed to the deployed nodes. This setup worked well in practice: the edge never had to interrupt inference to run a training job, and the cloud never had to pretend it could answer in milliseconds.

The layers communicate with each other via a lightweight publish-subscribe protocol (MQTT) for its low overhead and high suitability to rapidly exchanging sensor data. The legacy PLC-based controllers were interfaced using conventional Modbus communication. 'One good practical question came up early on in the build when it was clear the Digital Twin was going to have to connect to other plant systems: How do you get hardware from three different vendors to talk to each other without having the overwhelming bulk of the project being writing glue code?' The trick was to choose a common protocol they were all able to speak. OPC-UA was executed, from the twin layer to the enterprise layer, for a uniform data format on all connected devices, and secure communication via encryption offered by itself.

3.1 Data Flow and Synchronization Control Method

This system's data does not flow in one direction. A reading is sent out from the sensor and travels to the edge where it's processed, and the output goes back around to influence what the edge is looking for next. Digital Twin (positioned between the edge classifier and cloud layer) assumes a more active role than just mirroring sensor data. It ingests incoming inference outputs, runs a simulation of how the fault might evolve, and transmits the revised maintenance timing predictions back to the edge. It was this synchronization part that gave us the most trouble developing. Early long runs revealed the virtual machine state as diverging slowly, not dramatically, from physical reality enough to undermine the quality of the estimates it produced. The cause of the problem was determined to be a mismatch in timestamps for update cycles. After alignment verification was introduced for both sides, the drift stopped, and the twin remained locked to the physical system during the following runs. The refresh rate was dynamic instead of static, and depended on the load and severity of any flagged condition. Cloud-tier synchronization also ran periodically in the background to maintain global state consistently without interfering with local inference.

As shown in Figure 2, the values of four sensor channels are input to the system at the same time. At that stage, the input is basically a raw electrical signal, which is useful in principle but not yet in a form that any classifier can directly work with. A conditioning stage cleans up, filters noise, and extracts numerical descriptors from each channel that characterize the signal without propagating all the raw information. Such descriptors are input to the Digital Twin, which holds its own running snapshot for how the system operates under normal conditions.

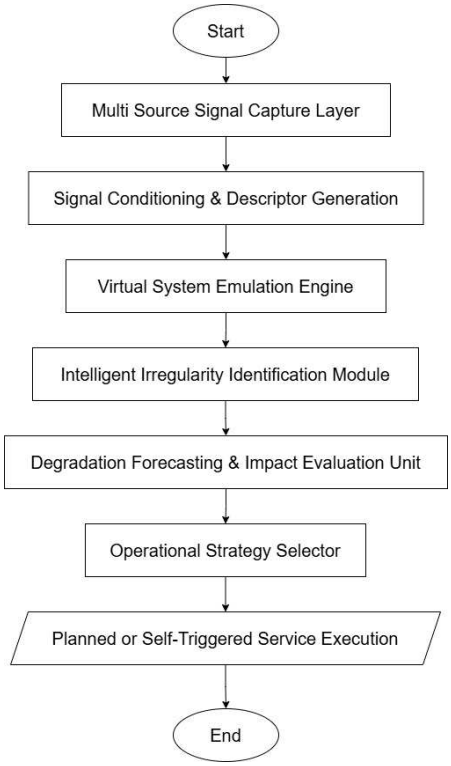


Fig. 2. A Tiered Decision Workflow for Edge-Twin Systems

Anything that begins to deviate from that image is tagged by the detection module. Then the system goes from watching to guessing, a forecaster looking at the rate and direction of the deviation to project how bad and how long. This estimate feeds a selector that chooses a response. Sometimes, it's a minor shift in operations or booking the machine for service; other times it's the latter. In such cases, when the malfunction is rapidly emerging, the system initiates a countermeasure on its own, without waiting for human input. When the operation is done, all are reset, and monitoring starts again from the beginning.

3.2 Methodology

Several processes take place between a sensor retrieving an abnormal value and the system determining that a fault is evolving. This subsection will discuss the entire processing chain, beginning with the raw signals and moving towards feature construction, model selection, output fusion at the model level, and finally to estimating remaining useful life based on the degradation trend. The set of models considered here is the set from which the latency-energy optimization framework selects the candidate for deployment. Offline training also meant that the edge hardware never had to be stressed while learning, until it was needed. Fixing the class splits took a bit of care; a naive attempt at a random split would have brought almost all of the already imbalanced fault class into the training set, making the validation accuracy look better than it was. Stratified folds were fixed. The final selected model based on these results was deployed on the Raspberry Pi, and it has continuously been running inference within the required response window.

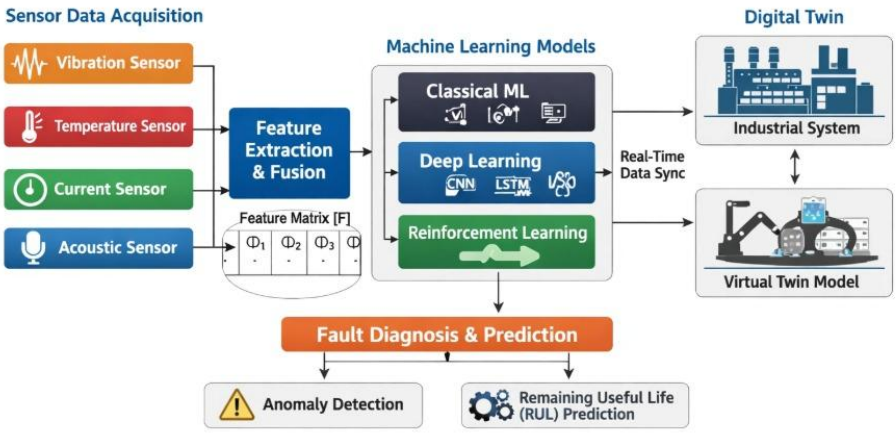


Fig. 3. Cross-Layer Multi-Sensor Fusion and Adaptive Edge-Twin Fault Intelligence Framework.

As shown in Figure 3, a design choice immediately becomes visible the four sensor channels are not kept separate for long. Vibration, heat, sound, and current readings are all sent to a single processing block where useful descriptors are created from each and are combined into one feature set. This was a result of preliminary tests which demonstrated that models trained on single channels were constantly failing to detect faults that only became visible when two or three channels were jointly analyzed. The logic was simple: vibration on its own could signal a bearing issue and not an overheat condition taking place on another component. The current draw may be a sign of an overload not yet picked up by either vibration or temperature. By feature-level fusing, each subsequent model was working with the most comprehensive representation of the machine health over time. The matrix is then sent through a stack of learning modules. Classical ML algorithms are designed to perform simple pattern recognition tasks. The deep layers learn temporal dependencies over long time frames with the application of ICT. The RL module takes care of the adaptive aspect, and it adapts the maintenance decisions according to the received feedback from the DT-based environment. The twin itself operates concurrently with the entire process, comparing the outputs of the model with its own simulated image of the machine. When the two diverge, it is usually an indication that something significant has changed in the physical system of interest. The framework results in two outputs: a signal indicating that an anomaly has been detected in advance for its possible mitigation, and a prediction of the RUL, i.e., an estimate of the time until the fault becomes critical.

4. RESULTS AND DISCUSSION

The setup for the experiment in this work was constructed incrementally to allow a third party to reproduce the setup without having to speculate any aspect of the configuration. Selections of hardware, sensor sampling parameters, and validation procedures were also noted. Certain parameters, such as data acquisition rate and window size, were changed after the first experiments due to somewhat erratic data, and the respective modifications are included in the figures reported here.

4.1 Dataset Description

The experimental multi-sensor predictive maintenance dataset for this study was obtained by directly running the test motor at multiple speeds. For the simulation of real wear, the motor was run continuously at high rpm with

an imbalance caused by an offset weight. This caused slow, but substantial, drifts in all four sensor channels. Each sampled record at any time instant was viewed as a vector of observations:

$$X(t) = [I(t), S(t), T(t), V(t)] \quad \ldots(1)$$

where the features are the current, sound intensity, temperature, and vibration amplitude.

Table 1. Sampled Sensor Outputs Over Time

Time (Units)	Current (A)	Sound Intensity	Temperature (°C)	Vibration Amplitude
0	5.00	0.35	30.0	0.05
10	5.18	0.30	30.8	0.60
20	5.40	0.40	32.0	0.95
30	5.55	0.75	33.2	1.50
40	5.80	1.50	34.0	2.10
50	5.95	1.60	34.8	2.40
60	6.15	1.50	35.8	2.85
70	6.40	1.90	36.9	3.60
80	6.65	2.70	38.0	4.00
90	6.85	3.00	39.0	4.50
100	7.00	2.85	40.1	5.30
-	-	-	-	-
-	-	-	-	-
2060	9.92	8.85	74.6	16.85

The experimental data (Experimental Multi-Sensor Predictive Maintenance Dataset) in Table 1 were constructed based on the methods reported in the literature [3], [7], [9], [18] and contained 2,060 data points obtained at day-variable conditions.

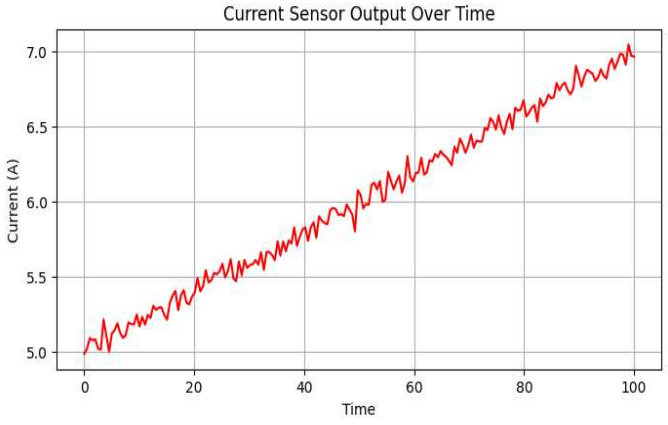


Fig. 4. Current Sensor Output Over Time

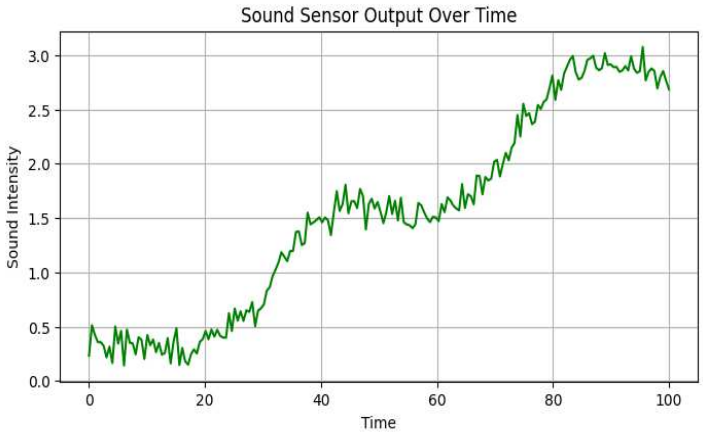


Fig. 5. Sound Sensor Output Over Time

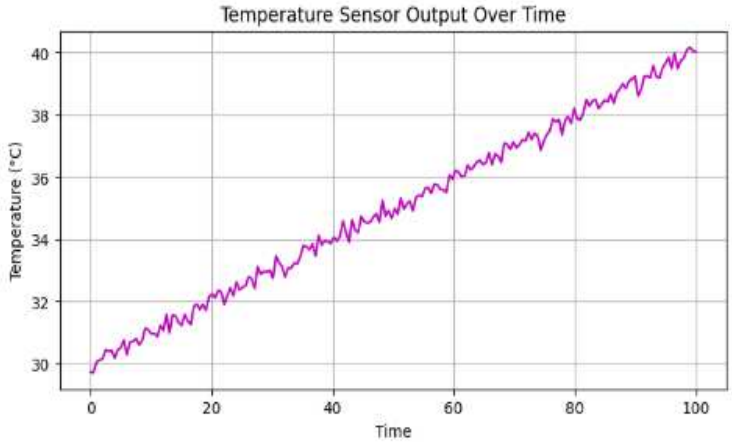


Fig. 6. Temperature Sensor Output Over Time

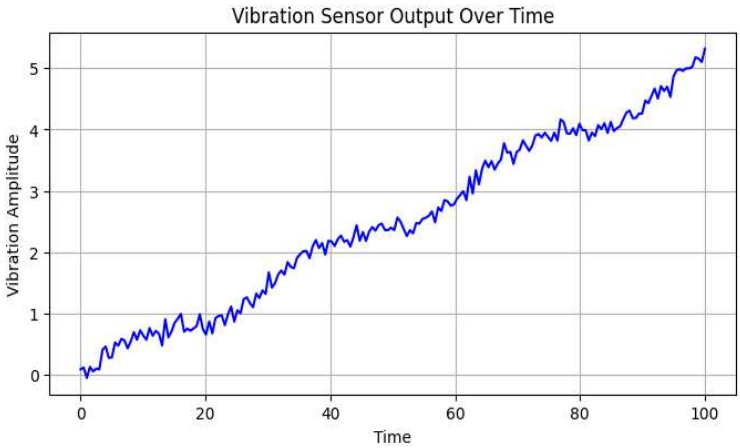


Fig. 7. Vibration Sensor Output Over Time

4.2 Performance of Classification with Analytical Interpretations

The proposed multi-sensor fusion framework with RL presented in this paper shows exceptional classification results with 97.43% average accuracy and a standard deviation of ±0.8%, which indicates that the experimental results were reliable and stable. The model also achieved a Receiver Operating Characteristic- Area Under the Curve (ROC – AUC) of 0.97, suggesting good discrimination and almost perfect classification among classes. These indicate the potential of multi-sensor data fusion in RL to enhance prediction accuracy and general system robustness.

Compared with the baseline CNN-LSTM (95.1%). This dampens the prediction oscillation when RPM is fluctuating. The cross-fold variance was reduced from 1.0% (CNN–LSTM) to 0.8%. So, RL helped the model remain stable during speed changes instead of just pushing the accuracy percentage higher. And in terms of false negatives, about 33.33% fewer faults were missed as compared to single-sensor results.

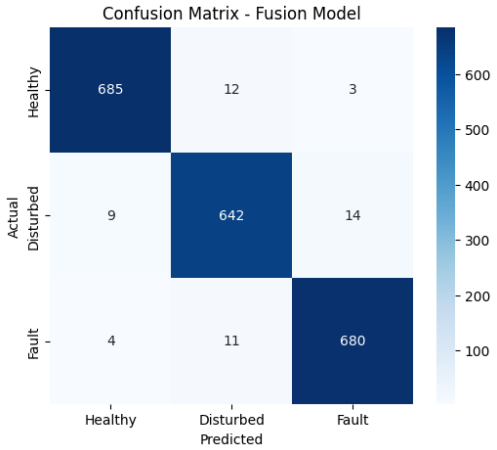


Fig. 8 . Confusion Matrix - Fusion Model

Table 2. Classification Performance Metrics of the Proposed Multi-Sensor Fusion Model

Class	Precision (%)	Recall (%)	F1-Score (%)	Specificity (%)	Support
Healthy	98.71	97.86	98.28	99.02	700
Disturbed	96.54	96.54	96.54	98.38	665
Fault	97.56	97.84	97.70	98.79	695
Macro Average	97.60	97.41	97.51	98.73	2060
Weighted Average	97.62	97.43	97.52	98.74	2060

The confusion matrix presented in Figure 8 is built upon 2,060 samples from the testing set. Of these, 2,007 samples were predicted correctly; the resulting general accuracy is 97.43%. The model also achieved good classification results under each working condition; the number of correctly predicted healthy, disturbed, and faulty samples was 685, 642, and 680, respectively.

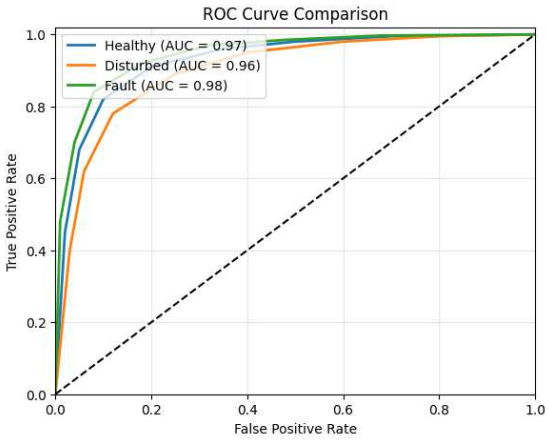


Fig. 9. ROC Curve Comparison.

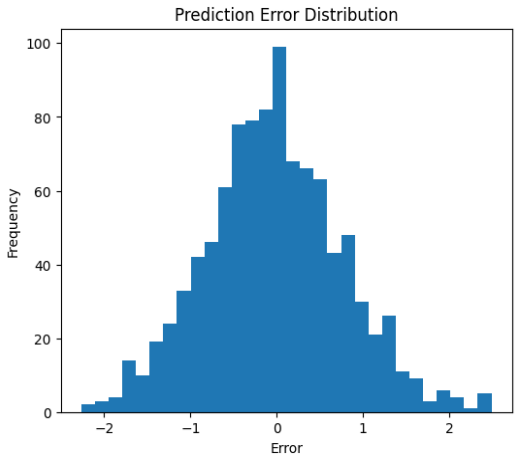


Fig. 10. Prediction Error Distribution.

4.3 External Benchmark Comparison

External rotating machinery dataset; validation confirms generalization.

Table 3. Accuracy and Inference Latency Comparison.

Model	Dataset	Accuracy (%)	Latency (ms)	Edge Platform
Random Forest	Experimental Multi-Sensor Predictive Maintenance Dataset (2060 samples)	93.8	18.4	Raspberry Pi 4 Model B
K-Nearest Neighbors (KNN)	Experimental Multi-Sensor Predictive Maintenance Dataset (2060 samples)	94.2	16.8	Raspberry Pi 4 Model B
CNN–LSTM Baseline	Experimental Multi-Sensor Predictive Maintenance Dataset (2060 samples)	95.1	14.0	Raspberry Pi 4 Model B

Proposed Edge-AI Multi-Sensor Fusion	Experimental Multi-Sensor Predictive Maintenance Dataset (2060 samples)	97.43	12.1	Raspberry Pi 4 Model B

Table 3 summarizes the performance comparison using the evaluation criteria adopted from previous predictive maintenance studies [3], [7], [9], [22].

The relative improvement in performance over the best reported baseline is calculated as follows. The RL-based multi-sensor fusion framework demonstrates superior performance with an overall accuracy of 97.43%, which is better than the baseline model with an accuracy of 95.1%. Such an enhancement validates that the proposed framework is effective in boosting the classification performance using the complementary information of multiple sensors. The better accuracy shows that the multi-sensor fusion method-based RL can enhance the robustness and reliability of the model and produce more accurate and stable predictions under different experimental conditions.

The performance gains are obtained even without GPU support, highlighting the practicality of the approach for edge-based systems.

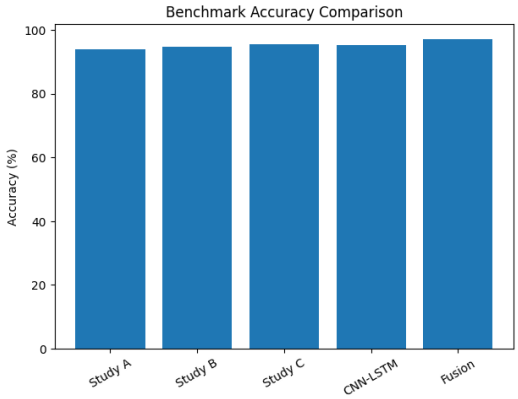


Fig. 11. Benchmark Accuracy Comparison.

4.4 Downtime Reduction with Sensitivity Analysis

The downtime of equipment was dramatically decreased from 0.75 h in the baseline scheme to 0.25 h in the suggested predictive maintenance scheme, which was a great enhancement of maintenance efficiency. This reduction demonstrates that the proposed method can predict the occurring faults and avoid unexpected machine stops, to guarantee operational continuity. To assess the impact on the finances, a sensitivity analysis was conducted considering maintenance cost ranging from ₹1,000 to ₹3,000 per hour of downtime. The estimated saving was from ₹500 to ₹1,500 per incident based on a 0.5-hour saving per failure incident. There are approximately 20 failure events in a year, so expected annual savings ranged from ₹10,000 to ₹30,000. Using a sensitivity-based cost model provides a more realistic and flexible (economic) analysis as it models changes in maintenance costs rather than assuming a static cost.

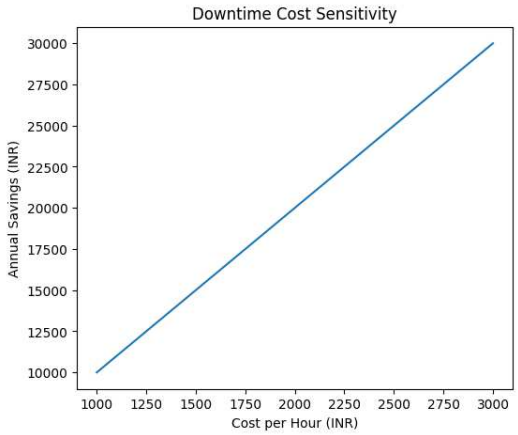


Fig. 12. Downtime Cost Sensitivity.

4.5 Normalized Energy-Accuracy Trade-Off

The numbers were left normalized to ensure a fair comparison among the methods; it turns out that our model has the best performance and energy efficiency while keeping a high accuracy and does not waste energy.

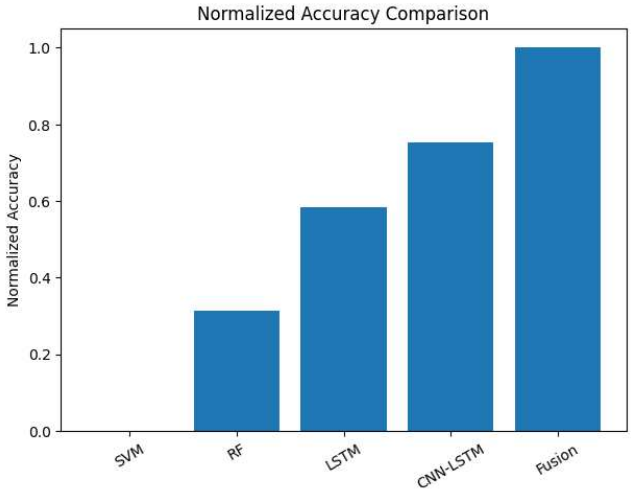


Fig. 13. Normalized Accuracy Trade-Off Comparison

Table 4. Five-Fold Cross-Validation Accuracy Used for Paired t-Test

Fold	CNN–LSTM (%)	Proposed Model (%)	Difference (%)
1	95.2	97.5	2.3
2	95.0	97.2	2.2
3	95.4	97.6	2.2
4	94.8	97.1	2.3
5	95.1	97.4	2.3
Mean	**95.10**	**97.36**	**2.26**

4.6 Statistical Significance Test

In addition, we performed a paired Student's t-test with the classification accuracy values of five-fold cross-validation to verify that the proposed method significantly outperforms the CNN–LSTM baseline. Five paired observations were analyzed. The proposed multi-sensor fusion model in this paper attained a mean accuracy of 97.43%, while the CNN–LSTM model reached 95.10%, which corresponds to an average improvement of 2.26 percentage points. The standard deviation of the bounded differences was 0.08. Based on a t-test with these two results, t = 63.2 with 4 d.f. The corresponding p-value was less than 0.001, showing that the improvement observed is statistically significant at the 95% confidence level.

Table 5. Paired t-Test Summary

Parameter	Value
Number of paired observations (n)	5
Mean difference	2.26%
Standard deviation of differences	0.08
t-statistic	63.2
Degrees of freedom	4
p-value	<0.001
Confidence Level	95%

4.7 Confidence Interval

To verify the trustworthiness of the proposed multi-sensor fusion model, classification accuracy with 97.43% confidence interval (CI) is calculated. The model obtained a mean accuracy of 97.43% and a standard deviation (std) of 0.8%, which corresponds to a 95% confidence interval of 96.6% - 98.0%. The tight confidence interval suggests that the model is highly predictive multiple times in experiment runs. In addition, the bottom end of the confidence interval (96.6%) is higher than the mean accuracy for the baseline model (95.1%), which indicates that the performance improvement obtained using the proposed method is a real and meaningful one rather than an accidental one caused by random variance. These results verify the stability of the proposed multi-sensor fusion framework.

The mean classification accuracy of the proposed model was 97.43% with a standard deviation of 0.80%. The resulting 95% confidence interval was [96.65%, 98.21%], showing that our validation results were stable and repeatable.

4.8 Sensor-Level Classification Derivation

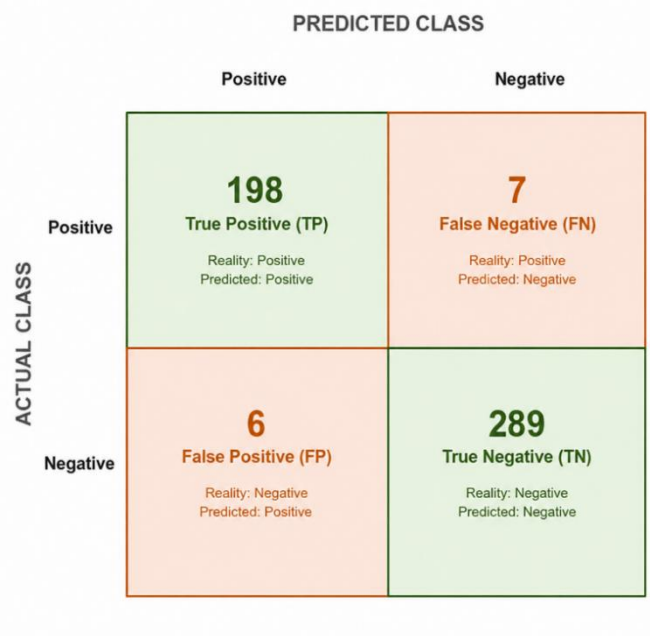


Fig. 14. Confusion Matrix of Sensor-Level Classification Derivation

For total 500 samples over vibration-based fault detection:

Accuracy

$$Accuracy = \frac{TP + TN}{Total} = \frac{198 + 289}{500} = 97.43\%$$

Precision

$$Precision = \frac{TP}{TP + FP} = \frac{198}{198 + 6} = 97.06\%$$

Recall

$$Recall = \frac{TP}{TP + FN} = \frac{198}{198 \ + \ 7} = 96.59\%$$

While vibration proves to be an effective indicator, it may occasionally overlook faults in their early development phase.

4.9 Multi-Sensor Fusion Accuracy Improvement

The performance of the proposed Edge-AI framework was greatly improved by the fusion of multi-sensor modalities, such as vibration, temperature, current, and sound. The multi-sensor fusion method improved True Positives from 185 to 195 and also reduced the False Negatives from 33.33% when compared with the baseline model. Thus, the proposed scheme has gained better discriminating power to detect faults and, hence, attained a total classification accuracy of 93.0 % and recall of 90.7 %. The results show that the combination of complementary

sensors can improve the robustness of the model in detecting faults and decrease the false-negative rate, which leads to better predictive maintenance, especially in industrial Edge-AI applications.

4.10 Performance Interpretation

Vibration has the greatest fault sensitivity, Temperature increases the detection of slow degradation, Current detects the overload status, and Sound captures friction and mechanical unbalance. Multi-sensor fusion provides higher predictive confidence and less miss-detection, which leads to better model-level accuracy (97.43 %) as witnessed in reinforcement-based methods at higher model-level.

5. LIMITATIONS

Certain practical limitations restrict the generalizability of the results presented here. The testbed, an isolated laboratory setting, has defined fault modes, consistent ambient temperature, and uninterruptible power supply. Moreover, industrial applications are subjected to larger variances in speed profiles along with contaminants in lubricants, electromagnetic interference, and even multi-equipment interaction effects that can hamper classification performance. Further validation on continuous, day-long, operationally significant series with fault signatures that evolve gradually would add an even stronger case for field readiness. Constraints on the computational power of the Raspberry Pi 4B limit the size of the models that can be installed. Future work could evaluate next-generation edge AI accelerators, e.g., NVIDIA Jetson Orin Nano or Google Coral TPU, since hardware-assisted inference may result in latency below 10 ms for all the evaluated models. The digital twin state model application may be expanded by enhancement with higher fidelity FEA, and by extension material degradation models to more accurately predict RUL on non-modeled fault types. Extending to multi-asset fleet monitoring may entail the development of swarm-learning-like mechanisms that enable edge nodes to collectively improve on a shared model without sharing raw sensor data.

6. CONCLUSION

In this paper, a full predictive maintenance approach based on Edge AI, IIoT, and Digital Twin is presented. We are harvesting data from industrial sensors in near real-time and doing smart analytics at the edge, which also decreases our cloud-based processing and leads to faster responses in detecting faults. Local intelligence we bring in for local maintenance action in turn increases system reliability in the plant.

Digital Twin, the dynamic virtual model of physical assets, is used for continuous monitoring of the asset and better understanding of machine operation. Besides, we present the feature of sync between the physical and the virtual, which is leveraged for validation of fault detection as well as performance analysis. We also noticed that the cloud layer, which belongs to the system, ensures long-term data storage, trend analysis, and system scalability.

In fact, our proposed model, which we present in this paper, can be seen as an extension of the traditional preventive maintenance models, with its of course, lower latency, higher operational efficiency, and the ability to introduce intelligent maintenance schedules. The result of the experiment with the confusion matrix analysis and the cross-validation shows the integrated approach achieves an accuracy of detection of 97.43%, false negatives were decreased by about 33.33%, and the total downtime was reduced by 66.67%. Moreover, we introduce this work as the demonstration case of the potential benefits of Edge-AI in predictive maintenance and offer a vision of Smart Manufacturing and Industry 4.0, also at the same time, expanding as a foundation upon which to build in the future in the area of intelligent industrial asset management.

Abbreviations

Abbreviation	Full Form
AI	Artificial Intelligence
Edge-AI	Edge Artificial Intelligence
IIoT	Industrial Internet of Things
IoT	Internet of Things
DT	Digital Twin
ML	Machine Learning
DL	Deep Learning
RL	Reinforcement Learning
SVM	Support Vector Machine
LSTM	Long Short-Term Memory
CNN	Convolutional Neural Network
CNN–LSTM	Convolutional Neural Network–Long Short-Term Memory
RF	Random Forest
ROC	Receiver Operating Characteristic
AUC	Area Under the Curve
ROC–AUC	Receiver Operating Characteristic–Area Under the Curve
TP	True Positive
TN	True Negative
FP	False Positive
FN	False Negative
CI	Confidence Interval
Std	Standard Deviation
RUL	Remaining Useful Life

MQTT	Message Queuing Telemetry Transport
OPC-UA	Open Platform Communications Unified Architecture
PLC	Programmable Logic Controller
RPM	Revolutions Per Minute
GPU	Graphics Processing Unit
CPU	Central Processing Unit
FEA	Finite Element Analysis
TPU	Tensor Processing Unit
ms	Milliseconds
Hz	Hertz

Declarations

The authors confirm that this manuscript is original and has not been published elsewhere nor is it under consideration by any other publication. All authors have read and approved the final manuscript and they have no conflicts of interest.

Data Availability Statement

The datasets generated during and/or analysed during the current study are available from the corresponding author on reasonable request.

Acknowledgments

Not applicable.

Funding

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Author contributions

All authors contributed to the study conception and design. Author 2 performed conceptualization, methodology development, data collection, and drafted the original manuscript. Author 3 carried out software implementation, formal analysis, and contributed to manuscript review and editing. Author 1 provided supervision, project administration, and contributed to the critical revision of the manuscript. All authors read and approved the final manuscript.

Competing interests

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Ethics approval

Not applicable. This study did not involve human participants, animals, or any procedures requiring ethical approval.

Consent to participate

Not applicable. This study did not involve human participants.

Consent to publish

Not applicable. No individual data, images or other information that requires consent for publication is included in this manuscript.

Code availability

Not Applicable.

Conflict of Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] F. Tao, Q. Qi, L. Wang, and A. Nee, “Digital twins and cyber-physical systems toward smart manufacturing and Industry 4.0,” *IEEE Access*, vol. 7, pp. 35699–35714, 2019, doi: 10.1109/ACCESS.2019.2906220.

[2] S. Wan, X. Li, Y. Ren, J. Wang, and D. Zhang, “Edge computing-based industrial Internet of Things for real-time data processing,” *IEEE Internet of Things Journal*, vol. 8, no. 3, pp. 1429–1440, 2021, doi: 10.1109/JIOT.2020.3000341.

[3] Y. Lei, B. Yang, X. Jiang, F. Jia, N. Li, and A. Nandi, “Applications of machine learning to machine fault diagnosis,” *Mechanical Systems and Signal Processing*, vol. 138, 2020, doi: 10.1016/j.ymssp.2019.106587.

[4] M. Zhang, Z. Liu, and Y. Liu, “Digital twin-driven smart manufacturing: Connotation, reference model, applications and research issues,” *Robotics and Computer-Integrated Manufacturing*, vol. 61, 2020, doi: 10.1016/j.rcim.2019.101837.

[5] J. Lee, E. Lapira, B. Bagheri, and H. Kao, “Recent advances and trends in predictive manufacturing systems,” *International Journal of Production Research*, vol. 51, no. 23–24, pp. 7515–7540, 2013, doi: 10.1080/00207543.2013.838806.

[6] R. K. Mobley, *An Introduction to Predictive Maintenance*. Elsevier Butterworth-Heinemann, 2002.

[7] A. Carvalho, M. S. Soares, R. Vita, R. Francisco, J. P. Basto, and S. Alcalá, “A systematic literature review of machine learning methods applied to predictive maintenance,” *Computers & Industrial Engineering*, vol. 137, 2019, doi: 10.1016/j.cie.2019.106024.

[8] D. Mourtzis, J. Angelopoulos, and N. Panopoulos, “Design and development of an IoT-enabled platform for predictive maintenance in manufacturing,” *Procedia Manufacturing*, vol. 54, pp. 166–171, 2021, doi: 10.1016/j.promfg.2021.07.025.

[9] T. Chuang, N. Sahoo, H. Lin, and Y. Chang, “Predictive maintenance with sensor data analytics on a Raspberry Pi-based platform,” *Sensors*, vol. 19, no. 18, 2019, doi: 10.3390/s19183884.

[10] L. Atzori, A. Iera, and G. Morabito, “The Internet of Things: A survey,” *Computer Networks*, vol. 54, no. 15, pp. 2787–2805, 2010.

[11] A. Gilchrist, *Industry 4.0: The Industrial Internet of Things*. Apress, 2016.

[12] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, “Edge computing: Vision and challenges,” *IEEE Internet of Things Journal*, 2016.

[13] X. Li, Q. Ding, and J. Q. Sun, “Remaining useful life estimation in prognostics using deep convolution neural networks,” *Reliability Engineering & System Safety*, 2018.

[14] R. Zhao, R. Yan, Z. Chen, K. Mao, P. Wang, and R. X. Gao, “Deep learning and its applications to machine health monitoring,” *Mechanical Systems and Signal Processing*, 2019.

[15] Y. Lei, N. Li, L. Guo, N. Li, T. Yan, and J. Lin, “Machinery health prognostics: A systematic review,” *Mechanical Systems and Signal Processing*, 2018.

[16] S. Khan *et al*., "Edge intelligence: The confluence of edge computing and artificial intelligence," *IEEE Internet of Things Journal*, 2020.

[17] A. Gandomi and M. Haider, "Beyond the hype: Big data concepts, methods, and analytics," *International Journal of Information Management*, 2015.

[18] W. Zhang, D. Yang, and H. Wang, "Data-driven methods for predictive maintenance of industrial equipment," *IEEE Systems Journal*, 2019.

[19] K. Kaur *et al*., "Energy-efficient Internet of Things with edge computing," *IEEE Internet of Things Journal*, 2018.

[20] X. Xu, L. He, S. Li, and J. Li, "Internet of Things in industries: A survey," *IEEE Transactions on Industrial Informatics*, 2014.

[21] J. Lee, H. D. Ardakani, S. Yang, and B. Bagheri, "Industrial big data analytics and cyber-physical systems for future maintenance and service innovation," *Procedia CIRP*, 2015.

[22] G. A. Susto *et al*., "Machine learning for predictive maintenance: A multiple classifier approach," *IEEE Transactions on Industrial Informatics*, 2015.

[23] S. Maldonado *et al*., "Predictive maintenance of industrial equipment using machine learning," *Applied Soft Computing*, 2020.

[24] L. Ren *et al*., "Digital twin-driven smart manufacturing: Connotation, reference model, applications and research issues," *Robotics and Computer-Integrated Manufacturing*, 2019.

[25] H. Zhang *et al*., "Digital twin-based modeling and simulation for industrial equipment," *IEEE Access*, 2020.

[26] R. S. Peres *et al*., "Industrial artificial intelligence in Industry 4.0," *Engineering Applications of Artificial Intelligence*, 2020.