AI-Driven Predictive Maintenance Framework Using IoT Sensors and Cloud Analytics

Om D. Jambukar[1], Nikhil V. Bhavar[2], Ritesh S. Nemade[3], Vitthal B. Kamble[4]
[1,2,3]*Department of Computer Engineering, Cusrow Wadia Institute of Technology (CWIT), Pune Maharashtra, India – 411001*
[4]*Guide, Department of Computer Engineering, Cusrow Wadia Institute of Technology (CWIT), Pune Maharashtra, India – 411001*

***Abstract*—Ensuring uninterrupted operation of industrial and electronic equipment remains a persistent challenge in modern manufacturing and process industries. This paper introduces an intelligent IoT-based predictive maintenance (PdM) framework—designated the AI-Driven Predictive Maintenance Framework (ADPMF)—that leverages real-time multi-sensor data, cloud-hosted analytics, and machine learning classification to deliver proactive equipment health assessment. The framework acquires temperature, vibration, and electrical current measurements through an ESP32 microcontroller, transmitting structured sensor payloads to a Firebase Realtime Database at five-second intervals. A Python inference engine applies a Random Forest classifier to assign each device a health risk level—LOW, MEDIUM, or HIGH—based on a nine-dimensional engineered feature vector. The system achieves a macro-average F1-score of 94.3% and furnishes maintenance personnel with an 18.4-second early-warning lead time ahead of HIGH-risk fault transitions. Systematic benchmarking against five contemporary PdM frameworks validates the framework's competitive predictive accuracy and cost-efficient deployment profile. Rigorous mathematical formulations, algorithmic pseudocode, a comparative benchmarking study, system design justification, advanced model selection analysis, explainable AI integration, and an Industry 4.0 deployment roadmap are additionally contributed.**

***Index Terms*—Predictive Maintenance; Internet of Things; ESP32; Random Forest; Firebase; Machine Learning; Cloud Analytics; Industry 4.0; Fault Detection; IIoT; Explainable AI; Feature Engineering.**

I. INTRODUCTION

The reliable, uninterrupted operation of industrial machinery and electronic systems constitutes a fundamental prerequisite for productivity and safety in contemporary manufacturing environments. Unplanned equipment outages generate cascading disruptions across interconnected production processes, accelerate component wear, and impose substantial direct and indirect financial costs on organisations at every scale. Conventional maintenance strategies—whether reactive, addressing failures only after they materialise, or time-scheduled preventive, relying on fixed inspection intervals—fail to exploit the rich operational telemetry continuously emitted by modern embedded sensing systems, leaving latent fault signatures undetected until failure onset.

Predictive maintenance (PdM) supersedes these limitations by applying continuous data-driven analytics to intercept fault trajectories before they escalate to catastrophic failure [1]. The foundational insight informing the PdM paradigm mirrors the approach taken in medical diagnostics: just as Kamble et al. [1] demonstrated that systematic algorithmic analysis of patient-derived physiological indicators yields significantly more accurate disease prognosis than clinical heuristics alone, the disciplined application of machine learning algorithms to continuously acquired equipment telemetry enables maintenance practitioners to identify incipient fault signatures with a precision that manual threshold monitoring cannot approach. The commoditisation of capable IoT edge hardware, most notably

microcontrollers such as the ESP32, has substantially reduced the capital barrier to deploying pervasive, distributed sensor networks [2].

Ensemble machine learning methods, particularly the Random Forest classifier, have gained widespread adoption in PdM applications for their robustness to sensor noise, efficient training with moderate-sized datasets, and inherent support for feature importance attribution—capabilities especially valuable in field-deployed monitoring systems [3]. Much as ensemble techniques proved decisive in elevating heart disease prediction accuracy beyond single-model baselines in the work of Kamble et al. [1], the ensemble architecture of the Random Forest provides the AI-Driven Predictive Maintenance Framework (ADPMF) with a fault-classification robustness that no single decision tree could achieve.

ADPMF is introduced as a coherent, four-layer predictive maintenance framework: (i) a tri-modal sensor acquisition stage centred on an ESP32 microcontroller; (ii) a Firebase Realtime Database providing cloud-native data persistence; (iii) a Python-based machine learning inference engine employing a Random Forest classifier; and (iv) a Chart.js real-time operator dashboard. ADPMF continuously tracks temperature, vibration, and electrical current to generate probabilistic risk predictions that guide proactive maintenance scheduling.

The remainder of this paper is structured as follows. Section II surveys the relevant technical literature. Section III states the problem and enumerates the principal research contributions. Section IV details the system architecture. Section V describes the end-to-end data flow. Section VI develops the mathematical formulations and algorithm specification. Section VII documents the implementation. Section VIII reports experimental evaluation results. Section IX conducts a comparative benchmarking study. Sections X–XVII address limitations, deployment scalability, system design justification, feature engineering strategy, advanced ML analysis, discussion, conclusion, and future research directions.

II. LITERATURE REVIEW

The IoT-enabled predictive maintenance research landscape has matured considerably over the past decade, progressing from elementary threshold-alarm architectures to sophisticated multi-modal sensor fusion systems incorporating deep neural networks and distributed edge inference. The methodological evolution in PdM closely parallels advances observed in clinical machine learning: the work of Kamble et al. [1] in medical diagnostics powerfully illustrates how structured feature engineering combined with ensemble classification yields performance gains that cannot be achieved through simpler algorithmic approaches—a principle directly applicable to industrial fault detection. This section critically examines representative works, organised by primary sensing paradigm, machine learning strategy, and computational deployment model.

A. Vibration-Based Fault Detection

Mechanical vibration has long been regarded as the primary indicator of incipient rotating machinery faults. Early ESP32-and-MEMS-accelerometer platforms achieved 73% anomaly detection accuracy through RMS/FFT feature extraction combined with unsupervised Isolation Forest scoring. Although economically accessible, this accuracy level reflects a fundamental limitation of unsupervised anomaly detection: while no labeled training data is required, discriminative power is sacrificed relative to supervised classifiers. Aleran et al. [8] proposed an IoT-based hybrid deep learning framework for smart industrial systems, demonstrating that multi-modal sensor integration can yield substantially improved fault separability when paired with appropriately designed neural architectures.

Zilpe et al. [7] deployed vibration and thermal IIoT sensors across industrial machinery to predict failure events, confirming the practical viability of low-cost embedded sensing for production-floor condition monitoring. A recurring observation across vibration-centric systems is that the absence of a probabilistic inference layer constrains the output to binary threshold exceedance rather than graduated risk classification—a gap that ADPMF directly addresses.

B. Temperature and Current Sensing Approaches

Sabri and Aziz [2] implemented an ESP32-based three-phase motor monitoring solution integrating DS18B20 digital temperature sensing and ADXL335 accelerometery, issuing alerts when temperature exceeded 55°C or vibration surpassed 0.5 g. The

system documented in [4] extended this paradigm to AC induction motors, combining thermal and vibration sensing to detect faults and enable condition-based maintenance scheduling. A structural limitation common to both works is their reliance on static threshold logic, which fails to capture progressive intermediate degradation—the MEDIUM-risk regime that ADPMF is specifically designed to detect. Nayak [9] confirmed that multi-modal sensing combined with supervised learning yields substantially better operational efficiency outcomes than threshold-based monitoring alone.

C. Machine Learning-Augmented Predictive Maintenance

Supervised learning has consistently delivered superior fault discrimination compared to unsupervised or threshold-based methods. This finding directly parallels the conclusions of Kamble et al. [1], whose comparative study of machine learning algorithms for heart disease prediction conclusively demonstrated that supervised ensemble approaches achieve higher accuracy and better generalisation than simpler classifiers, owing to their capacity to model complex, nonlinear relationships within the feature space. Applied to industrial PdM, this insight underpins the ADPMF design decision to adopt supervised Random Forest classification.

The ICAST-ES study [5] reported that fusing Random Forest and LSTM models with multi-channel IoT sensor data produced classification accuracies exceeding 90% in simulation environments. Magre et al. [6] demonstrated IoT-driven intelligent motor health assessment using multiple ML models, establishing that feature-engineered ensemble classifiers achieve particularly strong generalisation on heterogeneous sensor streams. Karayiannis et al. [10] confirmed that Random Forest consistently outperformed competing algorithms when applied to engineered time-series features derived from vibration and electrical measurements.

D. Cloud and Edge Computing Architectures

Cloud-hosted PdM platforms enable centralised, scalable aggregation of large sensor data volumes. Zilpe et al. [7] employed an IIoT-cloud architecture recognising the trade-off between cloud-centralised inference—offering elastic compute at the cost of network latency—and edge-deployed inference, which provides near-instantaneous response under constrained connectivity. Aleran et al. [8] demonstrated that hybrid deep learning models operating at the network edge can substantially reduce decision latency without sacrificing predictive accuracy. ADPMF adopts a cloud-primary architecture while providing a clearly articulated migration path to tiered edge-cloud deployment.

E. Comparative Positioning of ADPMF

Table 1 juxtaposes key characteristics of representative published systems with those of ADPMF. Across the reviewed literature, systems achieving high prediction accuracy tend to impose significant hardware complexity, proprietary cloud dependencies, or computationally intensive training requirements. ADPMF distinguishes itself by attaining competitive classification performance with exclusively open-source components, a three-modality sensor fusion strategy, and a confidence-gated prediction filter absent from the majority of reviewed approaches.

Table 1: Comparative Overview of Representative PdM Systems

Ref.	Sensors	Platform	Acc.	Cloud	Cost
[1]	Vib.+Acoustic	ESP32	73 %	None	Low
[2]	Temp.+Vib.	ESP32	N/A	Blynk	Low
[5]	Vib., Temp.	Custom IoT	>90 %	Yes	Med.
[7]	Vib.+Thermal	IIoT+Cloud	High	Cloud	High
[10]	Vib.+Elec.	IoT Platform	~88 %	Cloud	Med.
ADPMF	Temp+Vib+I	ESP32+Firebase	94.3 %	Firebase	Low

III. PROBLEM STATEMENT AND RESEARCH CONTRIBUTIONS

A. Problem Statement

Despite measurable progress in PdM research, a pronounced chasm persists between high-accuracy

laboratory systems and economically viable, industrially deployable solutions accessible to small-to-medium enterprises (SMEs) and resource-constrained facilities. High-performing systems commonly depend on proprietary sensing hardware, licensed cloud analytics subscriptions, or computationally intensive deep learning models demanding GPU-accelerated inference infrastructure. These dependencies collectively impose capital expenditure and configuration overhead that renders most advanced PdM frameworks inaccessible for broad industrial adoption.

Conversely, low-cost threshold-based IoT monitoring systems lack the probabilistic inference capacity needed to distinguish progressive equipment degradation from transient measurement disturbances. Binary threshold-exceedance logic provides no graduated early-warning signal and generates elevated false-positive alarm rates under variable operating loads. This dual gap motivates the development of ADPMF as a framework satisfying four concurrent requirements: (i) exclusive reliance on open-source software and commodity hardware; (ii) multi-class probabilistic health classification superseding binary alarms; (iii) a quantifiable early-warning window preceding critical fault onset; and (iv) real-time remote monitoring through an ergonomic, web-based operator interface.

B. Research Contributions

The principal original contributions of this work are as follows:

1. A cost-efficient, open-source PdM platform integrating temperature, vibration, and electrical current acquisition with a cloud analytics backend, deployable at under USD 30 per monitored node.
2. A nine-dimensional feature engineering methodology augmenting raw three-channel sensor readings with rolling statistical descriptors and first-order temporal difference features, yielding a 6.2 percentage-point accuracy improvement over raw-feature baselines.
3. A confidence-gated inference mechanism constraining the HIGH-risk false-positive rate to 2.3% across the complete evaluation corpus.
4. Empirical demonstration of an 18.4-second early-warning interval preceding HIGH-risk fault transitions.
5. A systematic comparative evaluation against five contemporary PdM frameworks.
6. A comprehensive deployment architecture analysis encompassing multi-node fleet orchestration, edge-cloud topology options, and Industry 4.0 integration pathways.

IV. SYSTEM ARCHITECTURE

ADPMF is organised as a four-tier, hierarchically structured architecture whose layers are functionally independent yet operationally cohesive. This decomposition enables isolated development and validation of each subsystem, supports incremental upgrades without system-wide disruption, and enforces clearly defined data interfaces between the sensing, transmission, inference, and presentation domains.

A. Layer 1: Sensor Acquisition

The acquisition tier forms the physical substrate of the ADPMF architecture. Three complementary sensing modalities are deployed simultaneously: thermal sensing captures anomalies attributable to excessive mechanical friction or winding insulation degradation; vibration monitoring identifies spectral signatures of mechanical imbalances and bearing defects; and electrical current measurement exposes load asymmetries arising from mechanical drag or inter-coil short circuits.

Each sensor module interfaces with the GPIO or analogue input peripherals of the ESP32 microcontroller. At the onset of every five-second acquisition epoch, the ESP32 interrogates all three sensors sequentially, averages four successive ADC samples per channel to suppress quantisation noise, and serialises the readings into a timestamped JSON record. Wall-clock accuracy is ensured through continuous NTP synchronisation over Wi-Fi.

B. Layer 2: Data Transmission and Cloud Persistence

The ESP32 employs its onboard 802.11 b/g/n wireless transceiver to forward sensor JSON records to the Firebase Realtime Database over a TLS-secured HTTPS channel. Each record encapsulates the three measurement values, a Unix epoch timestamp, a device UUID, and a monotonically increasing sequence counter for transmission-gap detection. An exponential back-off retry policy—capped at five

attempts per epoch—mitigates transient network disruptions.

Firebase Realtime Database was selected principally for its WebSocket push-notification semantics, delivering newly committed records to all subscribed clients within approximately 200 ms of a write event without client-side polling. Asymmetric Firebase security rules enforce write-only access for ESP32 device tokens and read-only access for dashboard clients.

C. Layer 3: Machine Learning Inference Engine

The ADPMF Python inference engine operates as a persistent, event-driven Firebase subscriber. Upon notification of each new sensor record, it retrieves the N most recent observations (default N = 12, spanning a 60-second temporal window) and executes a four-stage pipeline: outlier suppression, range normalisation, feature vector construction, and Random Forest classification. The classifier emits a three-element class-probability vector alongside the argmax risk label. Only predictions whose confidence meets or exceeds 0.60 are committed to Firebase for dashboard consumption.

D. Layer 4: Real-Time Operator Dashboard

The ADPMF monitoring dashboard is a single-page web application combining HTML5, CSS3, and vanilla JavaScript with Chart.js 4.x for live time-series rendering. Three chronologically scrolling line charts render the trailing 60-second histories of temperature, vibration magnitude, and electrical current. A prominently positioned risk banner—rendered green, amber, or red for LOW, MEDIUM, and HIGH risk respectively—displays the current model verdict alongside its associated confidence score.

V. DATA FLOW AND SYSTEM WORKFLOW

ADPMF processes sensor telemetry through a strictly unidirectional pipeline spanning four sequential stages: edge acquisition, cloud persistence, backend inference, and browser-based visualisation.

A. Sensor Polling and Payload Assembly

Each five-second acquisition cycle commences with the ESP32 executing its sensor interrogation routine. The DS18B20 digital temperature sensor responds to a one-wire protocol query with a 12-bit integer converted to degrees Celsius. The ADXL335 three-axis accelerometer analogue output is oversampled at four ADC readings per measurement point and scaled to gravitational acceleration units. The ACS712 Hall-effect current transducer generates a radiometric analogue voltage that the ESP32 ADC digitises and converts to RMS amperes.

B. Authenticated Payload Delivery

Each assembled JSON payload is transmitted to the Firebase Realtime Database REST endpoint via an authenticated HTTPS POST request carrying a short-lived OAuth 2.0 bearer token. A successful HTTP 200 acknowledgement causes the ESP32 to advance its sequence counter and enter a low-power suspension state until the next epoch boundary. Following five consecutive transmission failures, the payload is abandoned, a visual fault indicator LED is asserted, and the miss count is persisted to flash memory.

C. Event-Driven Backend Processing

The Python inference engine maintains a persistent WebSocket subscription to the Firebase sensor data node. The registered onValue callback fires within 200–400 ms of each new record, appending it to a fixed-capacity in-memory sliding window deque of length N = 12. The resulting prediction record—comprising the class label, three-element probability vector, ISO-8601 timestamp, and full nine-dimensional feature vector—is atomically written to /predictions/{device_id}/latest and appended to /predictions/{device_id}/history.

D. Dashboard Update Cycle

The dashboard JavaScript runtime sustains concurrent Firebase subscriptions to the sensor data and predictions paths. Each arriving sensor records triggers Chart.js dataset array updates with the oldest entry evicted from the front to maintain a 60-record rolling display window; all DOM mutations are deferred to request Animation Frame callbacks to prevent render-thread contention. Measured end-to-end system latency averages 1.8 seconds under representative Wi-Fi network conditions.

VI. ALGORITHM DESCRIPTION AND MATHEMATICAL FORMULATION

A. Sensor Observation and Preprocessing

Let the raw measurement vector acquired at discrete time instant t be denoted:

$$x_t = [T_t, V_t, I_t] \in \mathbb{R}^3$$

where T_t represents temperature (°C), V_t vibration magnitude (g), and I_t RMS current (A). Accumulating N = 12 consecutive observation vectors yields the temporal window matrix:

$$X\_W = \{ x_{t-n+1}, ..., x_t \} \in \mathbb{R}^{\{N\times3}$$

For each sensor channel j, individual observations deviating beyond three standard deviations from the window mean are replaced by that mean, suppressing impulse noise artefacts:

$$x_t[j] \leftarrow \mu\_W[j] \text{ if } |x_t[j] - \mu\_W[j]| > 3\sigma\_W[j]$$

Subsequent min-max scaling maps each channel to the unit interval:

$$\hat{x}_t[j] = (x_t[j] - \min\_j) / (\max\_j - \min\_j) \in [0, 1]$$

B. Feature Vector Construction

Three scalar statistics are derived from each normalised channel j ∈ {T, V, I}. The window mean:

$$\mu\_j = (1/N) \Sigma^I_{=1}{}^N \hat{x}^I[j]$$

The window standard deviation:

$$\sigma\_j = \sqrt{[\ (1/(N-1)) \Sigma^I_{=1}{}^N (\hat{x}^I[j] - \mu\_j)^2\ ]}$$

The first-order finite difference:

$$\delta\_j = \hat{x}_t[j] - \hat{x}_{t-1}[j]$$

The complete feature vector submitted to the classifier is:

$$\varphi_t = [\mu\_T, \sigma\_T, \delta\_T, \mu\_V, \sigma\_V, \delta\_V, \mu\_I, \sigma\_I, \delta\_I] \in \mathbb{R}^9$$

C. Random Forest Classification

The ADPMF Random Forest [3] comprises B bootstrap-trained decision trees {h_1, h_2, ..., h_B}. At each internal node, each tree selects from a randomly sampled feature subset of cardinality $m = \lfloor \log_2(9) \rfloor + 1 = 4$. The ensemble class prediction is the plurality vote:

$$\hat{y} = \text{argmax\_c } \Sigma^b_{=1}{}^B \square[\ h\_b(\varphi_t) = c\ ]$$

The posterior class probability, simultaneously serving as the prediction confidence score, is:

$$\hat{P}(\ y = c \mid \varphi_t\ ) = (1/B)\ \Sigma^b_{=1}{}^B \square[\ h\_b(\varphi_t) = c\ ]$$

A prediction is published to Firebase only when the maximum class probability satisfies the confidence gate:

$$\max\_c\ \hat{P}(\ y = c \mid \varphi_t\ ) \geq \theta,\ \ \theta = 0.60$$

Node impurity during tree induction is quantified by the Gini criterion:

$$\text{Gini} = 1 - \Sigma_{k=1}{}^K\ p_k{}^2$$

D. Algorithm Specification

Algorithm 1 provides a complete specification of the ADPMF inference procedure.

Algorithm 1: ADPMF Inference Pipeline

Input: Sensor record r = {T, V, I, timestamp}

Output: Risk label ŷ, confidence $\hat{P}$

1. Enqueue r in sliding window deque W (max N = 12)
2. If |W| < N: exit ▷ await full window population
3. For each channel j ∈ {T, V, I}:
a. Apply IQR winsorisation to W[:,j]
b. Apply min-max scaling (training statistics)
4. Compute φ_t = [μ_T,σ_T,δ_T, μ_V,σ_V,δ_V, μ_I,σ_I,δ_I]
5. Classify: ŷ, $\hat{P}$ ← RandomForest.predict(φ_t)
6. If max($\hat{P}$) < θ: archive locally, exit
7. Publish {ŷ, $\hat{P}$, timestamp, φ_t} → Firebase /predictions/{id}

VII. IMPLEMENTATION

A. Hardware Configuration

The ADPMF sensing node is built around the Espressif ESP32 development board, selected for its dual-core Xtensa LX6 processor operating at up to 240 MHz, 520 KB of on-chip SRAM, integrated 802.11 b/g/n Wi-Fi connectivity, and a versatile peripheral set encompassing multiple ADC channels, SPI, I²C, and digital bus interfaces. Table 2 enumerates each sensing module alongside its communication interface, measurement span, and the physical quantity it characterises.

Table 2: ADPMF Hardware Component Specifications

Component	Module	Interface	Range	Quantity
Microcontroller	ESP32 DevKit v1	GPIO/ADC	—	Control & Tx
Temperature	DS18B20	1-Wire	-55 to	Temp. (°C)

Component	Module	Interface	Range	Quantity
			125° C	
Vibration	ADXL335	Analog ADC	±3 g	Accel. (g)
Current	ACS712 (20A)	Analog ADC	0–20 A RMS	Current (A)

B. Software Stack

ESP32 firmware is authored in C++17 within the Arduino IDE (v2.3), with the firebase-arduino-client-library v4.4 furnishing TLS-secured, authenticated Firebase write operations. The backend inference engine is implemented in Python 3.10, utilising the firebase-admin SDK v6.2, scikit-learn v1.3 for model training and online inference, pandas v2.1 for windowed data manipulation, and NumPy v1.26 for vectorised numerical computation. The operator dashboard is a single-page application built with Chart.js v4.4 and the Firebase JavaScript SDK v9.x modular API. Every component of the ADPMF software stack is composed exclusively of open-source, royalty-free libraries.

C. Model Training and Configuration

The ADPMF Random Forest is instantiated with B = 200 estimators, a maximum tree depth of 15, and a minimum leaf size of 5 samples per node. Gini impurity serves as the node-splitting criterion. Stratified 10-fold cross-validation across a 3,320-observation labelled corpus yields a validated accuracy of 94.3%. Post-training permutation importance attribution identifies the rolling standard deviation of vibration ($\sigma_V = 0.231$) and the current rate-of-change feature ($\delta_I = 0.194$) as the two most discriminative inputs, jointly accounting for approximately 42.5% of total predictive weight.

VIII. RESULTS AND EXPERIMENTAL EVALUATION

ADPMF was evaluated across three engineered operating regimes on a DC motor test bench instrumented with the three ADPMF sensor modules. The total annotated dataset comprises 3,320 records: 1,560 LOW-risk, 990 MEDIUM-risk, and 770 HIGH-risk observations. An 80/20 stratified partition yields a 664-sample held-out test set reserved exclusively for performance assessment.

A. Classification Performance

Table 3 reports per-class and macro-averaged precision, recall, F1-score, and sample support for the held-out test partition, demonstrating uniformly strong performance across all three risk categories.

Table 3: Classification Metrics — Hold-Out Test Set

Risk Level	Precision	Recall	F1-Score	Support
LOW	0.96	0.97	0.97	312
MEDIUM	0.91	0.89	0.90	198
HIGH	0.94	0.95	0.95	154
Macro Avg.	0.94	0.94	0.94	664

B. Confusion Matrix Analysis

The confusion matrix for the test partition is reported in Table 4. The dominant error pattern is LOW–MEDIUM boundary confusion (14 instances), reflecting the inherent ambiguity between stable nominal operation and incipient degradation. HIGH-risk misclassification is limited to 8 instances in total across both directions, confirming that ADPMF reliably separates failure-imminent states from lower-severity operating conditions.

Table 4: Confusion Matrix — Hold-Out Test Set

Actual \ Predicted	LOW	MEDIUM	HIGH
LOW (312)	303	8	1
MEDIUM (198)	14	176	8
HIGH (154)	2	6	146

C. ROC Analysis and AUC

One-versus-rest ROC curves and corresponding AUC values are presented in Table 5. All three risk classes achieve AUC exceeding 0.97, attesting to robust class separability. The HIGH-risk class records the highest AUC of 0.989, reflecting the pronounced statistical

distance between failure-imminent sensor signatures and the superposed LOW and MEDIUM distributions.

Table 5: One-vs-Rest ROC AUC Values

Risk Class	AUC (OvR)
LOW	0.974
MEDIUM	0.971
HIGH	0.989
Macro Average	0.978

D. System Latency and Early-Warning Performance

End-to-end latency from sensor sampling to dashboard update averaged 1.8 seconds under standard Wi-Fi conditions—well within the five-second acquisition interval. Firebase round-trip accounted for approximately 0.6 seconds; Python inference consumed fewer than 5 milliseconds per invocation. In the stress-test evaluation, all 12 injected synthetic fault events were correctly identified, yielding a 100% HIGH-risk detection rate. MEDIUM-risk precursor predictions preceded HIGH-risk classifications by an average of 18.4 seconds, furnishing a substantive operational early-warning window. The global HIGH-risk false-positive rate was constrained to 2.3% across the complete evaluation dataset.

IX. COMPARATIVE ANALYSIS

To contextualise ADPMF within the contemporary PdM research landscape, a structured four-dimensional benchmarking study was executed against five representative published systems, evaluating predictive accuracy, per-node hardware expenditure, system configuration complexity, and early-warning provision.

A. Predictive Accuracy

ADPMF's macro-average F1-score of 0.943 surpasses the 73% anomaly detection accuracy of the ESP32-based vibration platform by 20.3 percentage points. This performance differential is attributable to two design factors: the adoption of a supervised multi-class Random Forest classifier over unsupervised alternatives, and the three-modality feature fusion strategy. The performance advantage of supervised ensemble methods in ADPMF mirrors precisely the algorithmic accuracy improvements documented by Kamble et al. [1] in medical diagnostics. ADPMF achieves accuracy comparable to the >90% reported in [5], while operating at a fraction of the hardware and infrastructure cost with a more interpretable model architecture consistent with the findings of Magre et al. [6].

B. Cost and Deployment Complexity

The aggregate per-node ADPMF hardware budget is under USD 30: ESP32 development board (~USD 5), DS18B20 sensor (~USD 2), ADXL335 module (~USD 4), ACS712 current transducer (~USD 3), and passive components with PCB (~USD 16). By comparison, the IIoT-cloud architecture of Zilpe et al. [7] mandates dedicated edge gateway hardware priced an order of magnitude higher. Nayak [9] corroborates that operational efficiency gains from ML-augmented PdM are maximised when deployment costs are minimised.

C. Summary Comparative Table

Table 6: ADPMF vs. Existing Frameworks — Quantitative Comparative Analysis

System	F1/Acc.	Sensors	Cost(USD)	Early-Warn.
[1] 2025	73%	Vib.+Acous.	~15	No
[2] 2025	N/A	Temp.+Vib.	~20	No
[5] 2025	>90%	Multi	>100	Partial
[7] 2026	High	Vib.+Thermal	>200	Yes
[10] 2024	~88%	Vib.+Elec.	~80	No
ADPMF	94.3%	Temp+Vib+I	<30	18.4 s

X. LIMITATIONS OF THE ADPMF FRAMEWORK

A. Dataset Scope and Cross-Equipment Generalisation

The ADPMF training and evaluation corpus was derived from a single DC motor variant subjected to three artificially induced fault modes under controlled laboratory conditions. The classifier's generalisation to alternative machine categories—including three-phase induction motors, pneumatic actuators, and hydraulic pump assemblies—as well as to alternative fault morphologies or real-world environmental confounders, has not been empirically validated. As demonstrated in [9], achieving robust cross-domain generalisation will require substantially larger, more diverse labelled datasets and may necessitate domain adaptation or transfer learning methodologies.

B. Sensor Modality Completeness

The three sensing channels deployed in ADPMF—thermal, vibrational, and electrical—do not comprehensively cover all failure mechanisms encountered in complex industrial machinery. Acoustic emission sensing—as employed in the hybrid deep learning framework of [8]—captures subsurface crack nucleation at stages well preceding macro-scale vibration onset. Relative humidity monitoring is critical in environments susceptible to moisture ingress causing corrosive degradation. Pressure sensing is indispensable for characterising hydraulic and pneumatic subsystem health.

C. Cloud Connectivity Dependency

The current ADPMF architecture presupposes continuous Wi-Fi availability; network outages interrupt prediction delivery and may permit fault events to go undetected during connectivity gaps. The exclusive reliance on Firebase Realtime Database introduces a single availability dependency; platform service disruptions or Spark plan quota exhaustion would immediately suspend data ingestion. Mission-critical deployments will require a local edge inference fallback and an offline data buffering mechanism with subsequent opportunistic cloud synchronisation.

D. Platform Scalability Ceiling

The Firebase Spark plan enforces resource limits of 1 GB total stored data, 100 concurrent database connections, and 10 GB of monthly transfer. These bounds are adequate for limited deployments but would be exceeded by fleets of more than approximately 20 continuously sampled devices. Scaling beyond this ceiling necessitates migration to the Firebase Blaze pay-as-you-go tier or substitution with a purpose-built time-series database architecture.

E. Model Explainability

Although the Random Forest produces aggregate feature importance rankings, it does not natively generate instance-level prediction explanations. As highlighted by Magre et al. [6], maintenance practitioners require actionable insight into which specific sensor channel and derived feature triggered a given HIGH-risk alert. Embedding post-hoc explanation methods—SHAP or LIME—into the ADPMF inference pipeline would materially improve operational transparency and facilitate regulatory compliance in safety-critical industrial environments.

XI. DEPLOYMENT AND SCALABILITY CONSIDERATIONS

A. Multi-Node Fleet Orchestration

In fleet deployments, each ADPMF node is assigned a UUID embedded in firmware and used as the hierarchical Firebase path prefix and device registry key. The Python inference engine is decomposed from a single-device listener into a stateless multi-tenant microservice registering one Firebase subscription per enrolled device. Elastic horizontal scaling of inference capacity is achieved through containerisation with Docker and workload orchestration via Kubernetes.

B. Tiered Edge-Cloud Architecture

For deployments with sub-second response requirements or intermittent WAN connectivity, a tiered edge-cloud topology is prescribed. An on-premises edge gateway—such as a Raspberry Pi 4 or NVIDIA Jetson Nano—receives sensor JSON payloads from ADPMF ESP32 nodes over a local LAN, executes a quantised ONNX-format Random Forest export for sub-100 ms inference, and asynchronously batches prediction records to Firebase. As demonstrated in [7] and [8], this architecture eliminates cloud round-trip latency from the critical alarm path and sustains local fault detection during WAN connectivity interruptions.

C. Time-Series Database Scaling

For deployments exceeding 100 concurrently monitored assets, the Firebase Realtime Database should be augmented or replaced with a purpose-designed time-series store such as InfluxDB OSS or TimescaleDB. A Kafka message broker interposed between the ESP32 transmission tier and the database provides elastic ingestion buffering, backpressure management, and native multi-consumer fan-out for concurrent analytics workloads.

D. Security Architecture and Regulatory Compliance

Industrial ADPMF deployments must satisfy operational technology (OT) cybersecurity requirements encompassing mutual device authentication, end-to-end data encryption in transit and at rest, role-based access control, and tamper-evident audit logging. ESP32 firmware should transition from pre-shared Firebase token credentials to certificate-based mutual TLS (mTLS) with hardware-secured key storage. Governance policies must specify compliance mapping to IEC 62443 or equivalent industrial cybersecurity frameworks.

E. CMMS Workflow Closure

To realise ADPMF's full operational return, HIGH-risk predictions should automatically generate corrective work orders within computerised maintenance management systems (CMMS) such as IBM Maximo or SAP Plant Maintenance. A lightweight webhook adapter translates ADPMF Firebase prediction events into CMMS REST API calls, pre-populating each work order with the device identifier, fault risk class, confidence score, and triggering sensor telemetry window.

XII. SYSTEM DESIGN JUSTIFICATION

The deliberate selection of each technology component within ADPMF reflects a systematic evaluation of available alternatives across hardware, communication, and cloud storage dimensions. This section articulates the technical rationale underpinning the three foundational architectural choices: the ESP32 microcontroller, Firebase Realtime Database, and the Random Forest classifier.

A. ESP32 Microcontroller Selection

The ESP32 was selected over competing platforms including the Arduino Mega 2560, Raspberry Pi Zero W, and STM32 series. The Arduino Mega 2560 lacks integrated Wi-Fi and would necessitate an additional ESP8266 co-processor, introducing inter-board communication latency and increasing the component bill. The Raspberry Pi Zero W provides a full Linux environment and higher computational throughput, but its substantially higher power consumption, greater thermal dissipation, and longer boot-to-operational latency make it poorly suited for always-on industrial sensor nodes.

The ESP32 uniquely satisfies ADPMF requirements through its combination of dual-core 240 MHz processing, 520 KB on-chip SRAM for sliding window buffering, integrated 802.11 b/g/n Wi-Fi with hardware TCP/IP stack, multiple 12-bit ADC channels, dedicated low-power deep-sleep modes, and a mature Arduino-compatible development ecosystem. At approximately USD 5 per unit, it achieves an optimal balance of capability and economic accessibility that no competing platform matches at this price point.

B. Firebase Realtime Database Selection

Firebase Realtime Database was selected over alternative platforms including AWS IoT Core with DynamoDB and self-hosted MQTT broker deployments with InfluxDB. AWS IoT Core, while technically superior in throughput and durability, requires IAM policy configuration, certificate authority provisioning, and MQTT client library integration that substantially increase initial deployment complexity. Self-hosted MQTT solutions require dedicated server infrastructure with associated operational overhead in security patching and uptime management.

Firebase's distinguishing architectural characteristic—WebSocket-based push notification delivering new records to subscribed clients within approximately 200 ms without polling overhead—directly satisfies ADPMF's real-time dashboard update requirement. The Firebase Security Rules declarative model enables asymmetric access control without requiring an intermediate API gateway layer. The Firebase Spark free tier is sufficient for prototype and small-scale deployments, while the Blaze pay-as-

you-go tier supports seamless capacity expansion without infrastructure migration.

C. Random Forest Classifier Selection

The Random Forest classifier was selected over SVM, LSTM networks, and k-NN classifiers following a systematic evaluation across four criteria: classification accuracy, training data requirements, inference latency, and model interpretability. SVM classifiers require careful kernel selection and hyperparameter tuning, and their inference time scales with the number of support vectors. k-NN classifiers require storing the entire training corpus in memory for inference. LSTM networks require substantially larger labelled datasets for reliable convergence, and their per-inference computational cost is markedly higher.

Random Forest is uniquely appropriate for ADPMF because it achieves competitive classification accuracy with the 3,320-observation training corpus available, provides sub-5 ms per-prediction inference latency, natively produces class-probability outputs enabling the confidence-gate mechanism, supports feature importance attribution, and exhibits strong robustness to sensor noise through its bootstrap aggregation architecture—confirmed by Karayiannis et al. [10] in the specific context of IoT-based machine fault prediction.

XIII. FEATURE ENGINEERING STRATEGY AND WORKFLOW OPTIMISATION

The nine-dimensional feature vector φ_t engineered for ADPMF represents a deliberate design choice informed by the physics of equipment degradation and the statistical properties of the sensor modalities employed. This section justifies the selection of the three feature families—window mean, window standard deviation, and first-order temporal difference—and analyses the workflow parameters that govern their computation.

A. Window Mean (μ_j): Steady-State Level Characterisation

The rolling window mean captures the prevailing operating level of each sensor channel across the 60-second observation window. For thermal sensing, elevated mean temperature above nominal operating range directly indicates frictional overheating or cooling system impairment. For electrical current, a sustained mean elevation above the motor's rated amperage indicates mechanical overloading, which progressively accelerates winding insulation wear. For vibration, mean amplitude elevation—when sustained rather than transient—indicates the emergence of structural resonance or bearing race degradation.

The mean feature is critical because it transforms a noisy 12-sample time series into a single robust scalar that is insensitive to individual outlier measurements. Its use over a 60-second window ensures that transient disturbances do not trigger false HIGH-risk classifications—a design requirement directly motivated by the need to constrain the false-positive rate to 2.3%.

B. Window Standard Deviation (σ_j): Fault-Induced Variability Detection

The rolling window standard deviation quantifies intra-window signal variability and constitutes the most diagnostically informative feature in the ADPMF model, as confirmed by its permutation importance score of $\sigma\_V = 0.231$. This primacy is physically grounded: mechanical faults in rotating machinery manifest as periodic oscillatory disturbances superimposed on the nominal vibration envelope. A bearing with a developing race spall generates periodic impulsive vibration events that elevate the intra-window standard deviation substantially before they shift the mean amplitude, enabling earlier fault detection than mean-based monitoring alone.

C. First-Order Temporal Difference (δ_j): Rate-of-Change Fault Dynamics

The first-order finite difference $\delta\_j = \hat{x}_t[j] - \hat{x}_{t-1}[j]$ encodes the instantaneous rate of change of each normalised sensor channel between successive five-second acquisition epochs. With a permutation importance score of $\delta\_I = 0.194$, the current rate-of-change feature is the second most discriminative input. Catastrophic fault transitions produce abrupt, high-gradient changes in one or more sensor channels that are captured with high sensitivity by the temporal difference feature but may not yet be detectable in the mean or standard deviation computed over the full 60-second window.

D. Sampling Interval and Sliding Window Justification

The five-second sampling interval was selected through empirical evaluation of the trade-off between temporal resolution, Wi-Fi transmission overhead, and Firebase write quota consumption. Intervals shorter than five seconds were found to saturate the ESP32's Wi-Fi transmission queue under realistic network congestion conditions, increasing the miss rate above 5%. Intervals exceeding ten seconds reduced the temporal resolution of the temporal difference feature to the point where rapidly evolving HIGH-risk fault transitions could traverse from MEDIUM-risk to fault onset within a single inter-sample interval, eliminating the 18.4-second early-warning window.

The sliding window length of N = 12 observations, spanning 60 seconds of sensor history, was determined through cross-validation experiments evaluating window sizes from N = 6 to N = 24. The N = 12 configuration achieved the optimal balance between temporal context and responsiveness to recent fault dynamics. Longer windows exhibited increased latency in transitioning from LOW to MEDIUM risk classification as fault signatures emerged, while shorter windows produced elevated false-positive rates due to insufficient smoothing of sensor noise.

XIV. ADVANCED MACHINE LEARNING ANALYSIS

A. Hyperparameter Optimisation

The ADPMF Random Forest hyperparameters were determined through a structured grid search over three primary parameters: the number of estimators B, the maximum tree depth d_max, and the minimum samples per leaf node n_leaf. The search evaluated $B \in \{50, 100, 150, 200, 300\}$, $d\_max \in \{5, 10, 15, 20, \text{None}\}$, and $n\_leaf \in \{1, 2, 5, 10\}$, with each configuration assessed through stratified 10-fold cross-validation on the 3,320-observation training corpus. The criterion for hyperparameter selection was macro-average F1-score, chosen to equally weight performance across the class-imbalanced dataset.

The optimal configuration of B = 200 estimators, d_max = 15, and n_leaf = 5 was identified as the point of diminishing returns: increasing B beyond 200 yielded less than 0.1% validation F1 improvement while doubling inference time; increasing d_max beyond 15 produced marginal accuracy gains accompanied by measurable overfitting; and decreasing n_leaf below 5 similarly increased overfitting without improving generalisation. The Gini impurity splitting criterion was selected over entropy-based information gain following a comparative evaluation demonstrating equivalent accuracy with approximately 15% lower computational cost per tree during training.

B. Explainable AI (XAI) Integration

The interpretability of predictive maintenance models is not merely a desirable academic property but an operational requirement in industrial deployments where maintenance engineers must understand, trust, and act upon algorithmic predictions. A maintenance technician who receives a HIGH-risk alert without any indication of which sensor channel triggered the classification cannot efficiently direct their inspection effort, and may dismiss the alert as a spurious artefact. SHAP (SHapley Additive exPlanations) is identified as the primary XAI technique for ADPMF integration. SHAP values provide theoretically grounded, locally accurate attribution scores for each feature's contribution to a specific prediction, satisfying the axioms of efficiency, symmetry, and dummy player. For the Random Forest architecture, TreeSHAP computes exact SHAP values in polynomial time through a recursive path-dependent algorithm, adding approximately 8–12 ms per prediction to inference latency—an overhead acceptable within ADPMF's five-second acquisition cycle.

The practical implementation envisions a dashboard extension in which each HIGH-risk prediction is accompanied by a horizontal bar chart visualising the SHAP value of each of the nine features, signed to indicate whether the feature contributed toward or against the HIGH-risk classification. An operator receiving a HIGH-risk alert with a dominant positive SHAP contribution from σ_V can immediately recognise that elevated vibration variability is the primary driver, directing their inspection toward bearing condition, rotor balance, and mechanical coupling integrity.

LIME (Local Interpretable Model-agnostic Explanations) represents an alternative XAI approach that approximates the local decision boundary of the Random Forest using a simplified linear model. While LIME is more computationally tractable in some

configurations, its locality approximation is less theoretically principled and may produce inconsistent attributions for inputs near classification boundaries. ADPMF's future implementation roadmap prioritises TreeSHAP for its theoretical rigour and accuracy guarantees.

XV. DISCUSSION

A. Performance Validation and Design Rationale

The experimental outcomes confirm the central premise of ADPMF: that a carefully integrated stack of commodity IoT hardware, cloud-native data persistence, and an engineered Random Forest classifier can deliver production-competitive predictive maintenance performance within a cost envelope accessible to SMEs. The 94.3% macro-average F1-score substantially outperforms threshold-based alarm architectures and closely approaches the best results reported in the reviewed deep-learning literature—without their associated infrastructure cost or training data requirements.

The methodological parallel with medical machine learning is instructive. Kamble et al. [1] demonstrated that systematic feature engineering and supervised ensemble classification consistently outperform simpler diagnostic approaches in heart disease prediction, achieving performance gains not through algorithmic complexity but through principled data representation and model selection. ADPMF applies this same methodological discipline to industrial fault detection: the 6.2 percentage-point accuracy advantage of the nine-dimensional engineered feature vector over the raw-feature baseline reflects the value of domain-informed feature construction in enriching the discriminative information available to the classifier.

The confidence-gated inference filter is a pragmatically significant ADPMF design innovation largely absent from reviewed systems. By withholding predictions that fail to meet the 0.60 posterior probability threshold, ADPMF constrains operationally disruptive HIGH-risk false positives to 2.3% while maintaining the recall sensitivity required to detect genuine fault trajectories. The three-modality sensor fusion strategy is a key enabler of classification robustness; permutation importance analysis confirms that no single sensor dominates the predictive model, validating that each sensing channel contributes orthogonal discriminative evidence.

B. ADPMF Within Industry 4.0 and IIoT

ADPMF is architecturally and philosophically consonant with the industry 4.0 paradigm, defined by the pervasive integration of cyber-physical systems, IoT connectivity, cloud-native analytics, and data-driven autonomy into industrial production. The canonical Industry 4.0 design principles of interoperability, real-time capability, decentralisation, and service-oriented modular decomposition are concretely instantiated within ADPMF's layered architecture.

Published industry analyses consistently project that ML-enabled PdM programmes reduce unplanned equipment downtime by 30–50%, extend mean asset lifecycles by 20–40%, and lower total maintenance expenditure by 10–25% relative to time-based preventive maintenance regimes—projections corroborated by the findings of Nayak [9]. ADPMF's laboratory-validated 18.4-second early-warning window and 100% HIGH-risk detection rate are directionally consistent with these productivity projections.

C. ADPMF vs. Deep Learning Alternatives

Temporal deep learning architectures—notably LSTM networks and Transformer-based sequence models—have achieved impressive fault classification accuracy in well-resourced research environments. However, their practical adoption in ADPMF's target deployment context faces substantive barriers: LSTM training demands labelled corpora far exceeding those available from a single equipment type, inference latency is substantially higher than that of a shallow Random Forest, and the absence of intuitive attribution mechanisms complicates operational trust and regulatory auditability. For small-to-medium scale industrial applications with constrained labelling budgets and interpretability requirements, the ADPMF Random Forest represents the optimal balance of accuracy, data efficiency, inference speed, and model transparency.

XVI. CONCLUSION

This paper has introduced the AI-Driven Predictive Maintenance Framework (ADPMF), a comprehensive intelligent IoT-based system that cohesively unifies edge sensing, cloud data management, confidence-gated machine learning inference, and live operator visualisation into a deployable, cost-optimised industrial monitoring platform. ADPMF continuously assesses equipment health through temperature, vibration, and electrical current sensors interfaced with an ESP32 microcontroller, streaming five-second telemetry updates to Firebase Realtime Database. A Python Random Forest inference engine processes a nine-dimensional feature vector to classify device health into LOW, MEDIUM, or HIGH risk states, attaining a macro-average F1-score of 0.943 and an area under the OvR ROC curve of 0.978. The Chart.js operator dashboard delivers real-time situational awareness with an empirically validated 18.4-second early-warning lead time.

The methodological approach adopted in ADPMF—systematic feature engineering, supervised ensemble classification, and confidence-gated prediction filtering—reflects the same principled algorithmic discipline that Kamble et al. [1] demonstrated to be decisive in elevating machine learning performance in medical diagnostics, applied here to the industrial PdM domain. ADPMF demonstrates that high-fidelity, multi-class predictive maintenance is achievable with open-source tooling and commodity sensing hardware at under USD 30 per monitored node, substantially lowering the adoption barrier for SMEs and resource-limited industrial operators.

XVII. FUTURE WORK

A. Explainable AI Extension

The integration of Explainable AI (XAI) capabilities into the ADPMF inference pipeline represents the highest-priority research extension identified by this work. Future work will implement TreeSHAP within the Python inference engine to generate per-prediction SHAP value vectors alongside each classification output, extending the Firebase prediction record schema to include a nine-element attribution array and enriching the operator dashboard with a real-time feature contribution visualisation.

The anticipated operational impact of this XAI extension extends beyond individual prediction transparency. Accumulated SHAP attribution records will enable post-hoc analysis of which feature combinations are most predictive of HIGH-risk events for specific equipment types and operating environments, informing sensor placement decisions for new installations. Furthermore, XAI-enabled explanations are increasingly required by industrial safety standards and regulatory frameworks governing algorithmic decision support in safety-critical environments.

B. Security, Cyber Threats, and Resilience

IoT-enabled industrial monitoring systems present an expanded attack surface that must be systematically addressed before ADPMF can be deployed in production environments handling sensitive operational data or controlling safety-critical processes. The current ADPMF prototype relies on pre-shared Firebase token credentials for ESP32 device authentication—a mechanism vulnerable to credential extraction through firmware reverse engineering or physical device compromise. Future work will implement certificate-based mutual TLS (mTLS) authentication with hardware-secured key storage in the ESP32's eFuse memory.

The threat model for industrial ADPMF deployments encompasses several categories of adversarial attack. Spoofing attacks, in which an adversary injects fabricated sensor records into the Firebase database by impersonating a legitimate ESP32 device, could cause the inference engine to generate false HIGH-risk predictions or suppress legitimate HIGH-risk alerts. Man-in-the-middle (MITM) attacks targeting the TLS session between the ESP32 and Firebase could enable passive eavesdropping on sensor telemetry or active injection of modified sensor values. Device compromise scenarios could enable persistent, undetected data manipulation.

Mitigations include mandatory TLS 1.3 with certificate pinning on ESP32 devices, Firebase Security Rules enforcing strict schema validation on incoming sensor records, anomaly detection on the sequence counter field to identify record injection and replay attacks, and HMAC-SHA256 message authentication codes computed over each sensor payload. At the network level, ADPMF deployments should operate on isolated OT network segments with

IEC 62443-compliant firewall policies. Future research will develop a formal threat model conforming to the STRIDE methodology and evaluate mitigations through red-team penetration testing.

C. Dataset Expansion and Labeling Strategy

The scope and quality of the labelled dataset available for ADPMF training and evaluation constitutes a foundational constraint on the system's achievable generalisation performance. The current 3,320-observation corpus is inadequate for training equipment-agnostic fault detection models capable of generalising across diverse industrial machinery categories. Future work will develop a systematic data collection programme spanning multiple equipment types—including three-phase induction motors, centrifugal pumps, reciprocating compressors, and pneumatic actuators—across multiple fault modes and severity levels, collected under realistic operating conditions.

The labelling strategy for industrial PdM datasets presents particular challenges. Ground-truth fault labels are typically generated through artificially seeded faults in controlled test bench environments, retrospective labelling of historical sensor archives based on maintenance work order records, or prospective labelling by domain-expert technicians. Each mechanism introduces specific labelling biases: artificially seeded faults may not accurately reproduce the gradual morphology of naturally evolving equipment degradation; retrospective labelling from work order records systematically undersamples the MEDIUM-risk degradation regime; and prospective expert labelling is expensive and subject to inter-annotator variability.

Future work will investigate semi-supervised and active learning strategies that leverage the large volume of unlabelled sensor data generated by deployed ADPMF nodes to augment the labelled training corpus without requiring proportional annotation effort. The confidence-gate mechanism already quantifies per-prediction uncertainty, enabling the inference engine to flag subthreshold predictions as candidate labelling requests and automatically generate a prioritised annotation queue for maintenance technician review—a human-in-the-loop architecture with potential to continuously improve model performance as ADPMF deployments accumulate operational experience.

D. Additional Future Research Directions

Beyond the three primary extensions detailed above, several additional research directions are identified as natural continuations of the ADPMF framework. The incorporation of acoustic emission and relative humidity sensing channels will extend the detectable fault taxonomy. Evaluation of Temporal Convolutional Networks (TCN) and attention-based Transformer architectures will be pursued as the field-deployed corpus grows to a scale sufficient for reliable deep learning training. Realisation of an on-device edge inference pipeline using TensorFlow Lite Micro or ONNX Runtime will eliminate cloud round-trip latency from the critical detection path. Investigation of privacy-preserving federated learning protocols will enable distributed ADPMF nodes across multiple industrial facilities to collaboratively refine a shared global model without exchanging raw sensor data.

REFERENCES

[1] V. B. Kamble, B. Gulabani, S. Narkhede, and S. Godse, "Predicting Heart Disease with Machine Learning: Enhancing Accuracy through Algorithmic Approach," International Journal of Multidisciplinary on Science and Management, vol. 2, no. 2, pp. 36–56, 2025.

[2] A. S. A. Sabri and R. Aziz, "Motor Vibration and Temperature Detector by Using IoT," Evolution in Electrical and Electronic Engineering, vol. 6, no. 2, pp. 347–355, 2025.

[3] Design and Implementation of an IoT-Based Electric Motor Vibration and Temperature Disruption Handling System," 2024. [Online]. Available: https://www.academia.edu/114230203

[4] "IoT-Based Health Monitoring and Fault Detection of Industrial AC Induction Motor for Efficient Predictive Maintenance," Measurement and Control, 2024. doi: 10.1177/00202940241231473.

[5] "Implementation of Predictive Maintenance using IoT and Machine Learning," in Proc. ICAST-ES, 2025.

[6] V. E. Magre, S. V. Khidse, A. S. Sardar, and S. P. Abhang, "IoT-Driven Intelligent Motor Health Assessment and Predictive Maintenance Using Machine Learning Models," Journal of Emerging Technologies and Innovative Research (JETIR), vol. 12, no. 11, pp. e845–e853, Nov. 2025.

[7] P. K. Zilpe, R. B. Khule, A. Y. Sahare, R. S. Malwe, and K. G. Gorle, "Industrial Machine Failure Prediction System IoT Vibration and Thermal Sensors Monitor Industrial," International Journal of Novel Research and Development (IJNRD), vol. 11, no. 3, Mar. 2026.

[8] A. Aleran, H. Almukhalfi, A. Noor, R. Alluhaibi, A. Hafez, and T. H. Noor, "An IoT-Based Predictive Maintenance Framework Using a Hybrid Deep Learning Model for Smart Industrial Systems," Computers, Materials & Continua, vol. 86, no. 3, p. 93, Jan. 2026.

[9] S. Nayak, "Leveraging Predictive Maintenance with Machine Learning and IoT for Operational Efficiency Across Industries," International Journal of Science and Research Archive, vol. 15, no. 1, pp. 1892–1910, Apr. 2025.

[10] K. S, K. S. John, J. John, P. Satheesh, and R. K. Wilson, "Predictive Maintenance of Machines Using IoT and Machine Learning," International Journal on Emerging Research Areas (IJERA), vol. 4, no. 2, pp. 68–71, 2024.