

Edge-Cloud Intelligent Monitoring and Maintenance for Smart Production Lines

Xinwen Shan, Jiaxin Lu*, Jiajia Zhu, Zihan Liu, Junjie Zha

State Grid Jiangsu Electric Power Co., Ltd. Information and telecommunication branch, Jiangsu, Nanjing, 210024, China

Abstract

INTRODUCTION: Smart manufacturing production lines integrate Computer Numerical Control (CNC) machines, industrial robots, motor drives, conveying units, sensors, and Programmable Logic Controller (PLC) workstations. Their operation and maintenance (O&M) requires low-latency recognition of coupled vibration, temperature, current, cycle-time, and alarm changes under production takt and downtime constraints.

OBJECTIVES: To improve real-time condition monitoring and maintenance decision-making, an edge computing-enabled collaborative intelligent O&M method is developed for discrete manufacturing production lines. The objective is to connect equipment state perception with health assessment, resource-constrained maintenance priority, and edge-cloud model updating.

METHODS: Multi-source manufacturing data are segmented into sliding status windows at edge nodes, where a lightweight Convolutional Neural Network–Long Short-Term Memory (CNN-LSTM) model extracts local impact features and temporal degradation patterns. The resulting anomaly scores and health levels are jointly modeled with workstation importance, production load, spare-part availability, labor-hour constraints, and downtime cost, while reinforcement learning updates maintenance actions according to execution feedback.

RESULTS: Experiments on CNC machining units, industrial robots, conveying units, motor drives, and PLC workstations show that the proposed model achieves an F1-score of 0.946, an Area under the Curve (AUC) of 0.973, and an average detection latency of 126 milliseconds (ms). The maintenance task matching rate reaches 94.1%, and the high-risk task priority resolution rate reaches 96.3%.

CONCLUSION: The method links edge-side condition recognition with manufacturing-oriented maintenance execution and cloud-side parameter calibration, providing a deployable closed-loop framework for continuous O&M in smart production lines.

Keywords: Edge computing; Smart production line; CNN-LSTM; Real-time condition monitoring; Health assessment

Received on 09 July 2026, accepted on 10 August 2026, published on 12 October 2026

Copyright © 2026 Xinwen Shan *et al.*, licensed to EAI. This is an open access article distributed under the terms of the [CC BY-NC-SA 4.0](#), which permits copying, redistributing, remixing, transformation, and building upon the material in any medium so long as the original work is properly cited.

doi: 10.4108/eetsis.13957

*Corresponding author. Email: Ljx1330403995@163.com

1. Introduction

Smart production lines operate through tightly coupled machining, transfer, assembly, and inspection processes, so a local equipment anomaly can rapidly propagate to downstream workstations and interrupt production continuity. Conventional cloud-centered maintenance

systems are often constrained by communication latency, bandwidth consumption, and delayed response, while fixed thresholds and periodic maintenance cannot adequately distinguish transient disturbances from progressive degradation. The practical challenge is therefore not limited to detecting abnormal equipment states. Monitoring results

must also be converted into maintenance actions that account for production takt, workstation criticality, downtime windows, spare-part availability, labor resources, and the cost of production interruption. Recent studies have established important foundations for intelligent maintenance, but the remaining limitations can be examined at the data, model, and decision layers. At the data layer, Bala et al. [1] confirmed that artificial intelligence and edge computing can reduce data-transmission pressure in equipment maintenance, although their study mainly classified available technologies and application patterns. Somu and Dasappa [2] developed an edge-cloud Industrial Internet of Things framework for predictive maintenance, demonstrating the value of distributed processing for manufacturing data. Sathupadi et al. [3] further showed that edge-cloud collaboration can improve the real-time processing of sensor streams. Abdullahi et al. [4] enhanced equipment-state synchronization through a distributed digital-twin framework. Nevertheless, vibration, temperature, current, cycle-time, alarm-event, and maintenance-record data differ substantially in sampling frequency, temporal granularity, reliability, and semantic form. Existing studies mainly focus on local processing, data transmission, or state synchronization, while the transformation of heterogeneous manufacturing data into temporally aligned and stable equipment-state representations remains insufficiently developed.

At the model layer, Mennilli et al. [5] verified the feasibility of deploying machine-learning models on resource-constrained Programmable Logic Controllers, but the representation of coupled and long-term degradation was limited. Artiushenko et al. [6] developed a resource-efficient edge solution for predictive maintenance, highlighting the importance of controlling computational and storage overhead. Shoorkand et al. [7] applied a hybrid Convolutional Neural Network-Long Short-Term Memory model to production and maintenance optimization, showing that convolutional feature extraction and temporal modeling can be combined within a unified architecture. However, existing approaches rarely achieve both lightweight edge deployment and sufficient sensitivity to local impacts, cross-cycle dependence, and progressive degradation. This limitation creates a gap between computational efficiency and reliable early-warning performance under production-line operating conditions.

At the decision layer, Varalakshmi and Kumar [8] introduced deep reinforcement learning for maintenance decision-making based on streaming Industrial Internet of Things data. Although such methods improve maintenance optimization, most existing studies terminate at anomaly detection, health prediction, or isolated action selection. Monitoring outputs are rarely coupled with production takt, workstation criticality, downtime windows, spare-part availability, labor resources, and production interruption costs in a unified decision process. Consequently, a complete closed-loop operations and maintenance mechanism is still lacking for converting heterogeneous manufacturing data into stable state representations, mapping edge-side recognition results to comparable health risks and

dynamically selecting executable maintenance actions under changing site constraints. Accordingly, the study constructs an edge-cloud collaborative closed-loop operations and maintenance chain through multi-source data integration, lightweight CNN-LSTM condition monitoring, anomaly scoring and health assessment, manufacturing-constraint-based maintenance prioritization, reinforcement learning-based action updating, and cloud-assisted parameter calibration. Accordingly, the study constructs an edge-cloud collaborative O&M chain comprising multi-source data integration, lightweight CNN-LSTM condition monitoring, anomaly scoring, health assessment, constraint-based maintenance prioritization, reinforcement learning-based action updating, and cloud-assisted parameter calibration. Its contribution is reflected in three linked elements: a lightweight temporal monitoring model designed for edge deployment; a quantitative mapping from anomaly recognition to health level and maintenance priority; and a feedback mechanism that coordinates edge-side response with cloud-side model updating. This structure connects equipment-state recognition with production continuity and supports continuous maintenance decision-making for smart production lines.

2. Edge-Cloud Maintenance Foundations

Edge-cloud collaborative maintenance distributes computing tasks according to latency sensitivity, data volume, and update scope. Edge nodes are placed close to Computer Numerical Control (CNC) machines, industrial robots, motor drives, conveying mechanisms, and PLC-controlled workstations. They receive vibration, temperature, current, rotational speed, air pressure, cycle time, alarm-event, and maintenance-record data generated under production-takt constraints. These sources differ in sampling frequency, format, reliability, and temporal granularity. High-frequency signals describe local impacts and rapid fluctuations, whereas low-frequency logs provide operational semantics related to tool wear, motor overheating, bearing looseness, robot execution deviation, transfer blockage, process switching disturbance, and unplanned downtime.

To obtain unified inputs, edge preprocessing performs timestamp alignment, noise filtering, missing-value imputation, status coding, standardization, and sliding-window segmentation [9]. Continuous sensor channels are aligned with event-driven PLC and alarm records in the same status window. A Convolutional Neural Network (CNN) extracts short-term impact patterns, while a Long Short-Term Memory (LSTM) network models cross-cycle dependence and progressive degradation. The resulting representation is combined with deviations from normal baselines, prediction residuals, and alarm severity to generate comparable anomaly scores.

Anomaly scores are mapped to equipment health levels by incorporating equipment load, alarm information, and maintenance history, thereby supplying quantitative inputs for maintenance task generation. Maintenance decisions must also satisfy production load, workstation importance,

downtime, spare-part inventory, labor-hour, and interruption-cost constraints. Edge nodes execute preprocessing, inference, health mapping, and urgent action triggering because these functions require short response times. The cloud aggregates cross-device and cross-shift records, retrains the monitoring model, calibrates health thresholds, updates maintenance-policy parameters, and distributes revised parameters to the edge. This division provides the technical basis for the closed-loop methodology developed in Section 3.

3. Edge-Cloud Collaborative O&M Method

3.1. Overall Architecture

To address delayed equipment-state perception, high latency of centralized cloud analysis, and insufficient maintenance adaptability in smart manufacturing production lines, an overall architecture for edge computing-enabled collaborative intelligent O&M is designed. The architecture covers manufacturing equipment sensing, edge preprocessing, real-time condition monitoring, health assessment, maintenance decision-making, cloud model calibration, and feedback control [10]. As shown in Figure 1, the closed-loop chain is organized as “Production Equipment — Sensor Acquisition — Edge Preprocessing — State Monitoring — Health Assessment — Maintenance Decision — Cloud Update — Feedback Control.” The production equipment layer corresponds to practical manufacturing resources, including CNC machining units, industrial robots,

transfer mechanisms, motor drive modules, PLC workstations, and alarm terminals. These devices are not independent monitoring objects. Their operating states are coupled by production takt, workstation sequence, material transfer rhythm, and quality inspection requirements, which means that maintenance decisions must consider both equipment risk and production continuity. The device sensor layer integrates multiple sources of data — vibration, temperature, electric current, rotational speed, cycle time, alarm log, and maintenance log — to cover a variety of risk sources, including mechanical wear, electric fluctuations, cyclic time anomalies, and maintenance behavior. Based on the device control cycle and the communication delay restriction, the edge processing period is set in the range of 200 ms to 1 s, and the cache length is configured to store a minimum of 30 seconds of continuous high frequency sampling data to ensure that the state sequence is not interrupted in the short term fluctuation of the network; the state recognition layer uses the sliding-window sample as input and uses the CNN-LSTM model to extract the local impact characteristics and the temporal evolution characteristics, thus solving the difficulty of capturing early degradation trends with single point thresholds [11]; in the Health Assessment Layer, the anomaly scores, the equipment load, the historical maintenance records are mapped to the health level, providing a uniform comparison scale for the risk status of the various devices; the maintenance decision layer generates the maintenance priority by integrating the health rating, the production line load, the spare parts availability, the downtime cost, and the urgency of the task, and calls the reinforcement learning strategy to update the maintenance action.

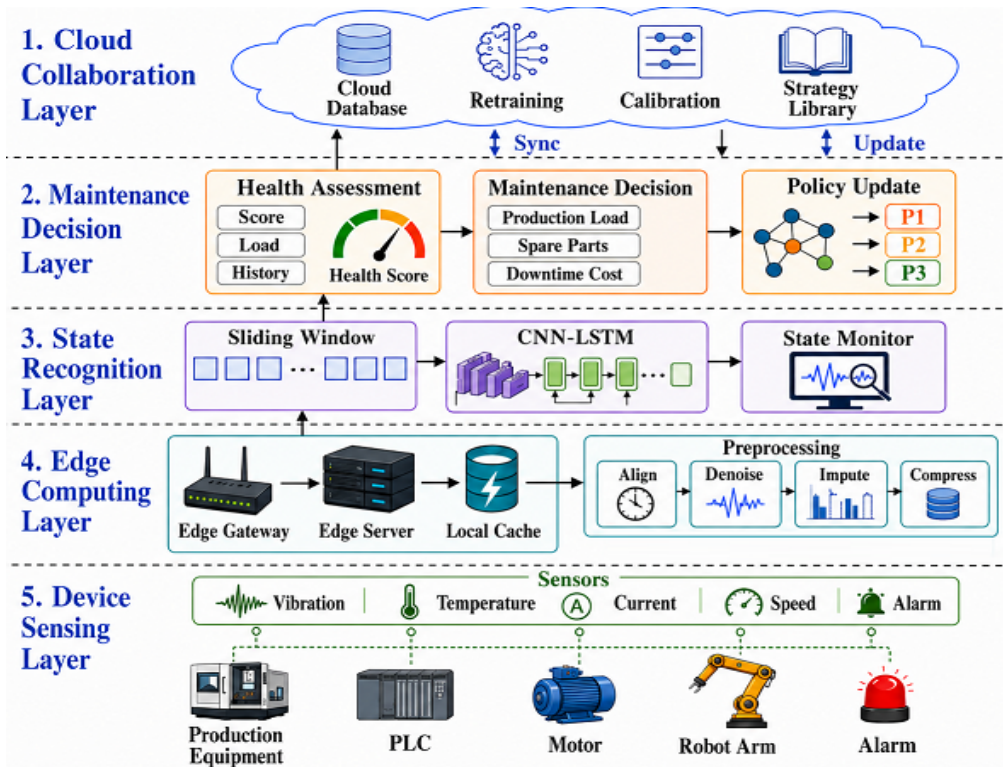


Figure 1. Overall architecture of edge computing-enabled collaborative intelligent operations and maintenance

3.2. Multi-Source Data Integration and Edge Preprocessing

Edge nodes are connected through OPC UA, Modbus TCP, MQTT, and industrial gateway interfaces to CNC devices, sensors, PLCs, visual inspection terminals, and maintenance protocols. They uniformly map vibration, temperature, current, rotation speed, air pressure, cycle time, alarm code, and maintenance record into data segments marked with device ID, station position, sampling time, and state [12]. Table 1 shows data collection objectives, preprocessing methods, and risk indicators for various data sources, so as to ensure that subsequent status monitoring models receive structural consistency and semantic clarity inputs. At the edge, a uniform time stamp is used to align high frequency sensor data and low frequency log data. The window length is set to 10 – 30 s based on the device's state change cycle and edge cache capacity, with a sliding step size of 1 – 5 s to balance the capture of short-term anomalies with inference

load control; for burst-type pulse interference, median filtering is used to attenuate isolated noise points, and for short term loss, the missing data is filled by interpolation based on neighboring sample points and trends of similar channels. Standardization is performed as follows:

$$x'_{i,t} = \frac{x_{i,t} - \mu_i}{\sigma_i + \varepsilon} \quad (1)$$

where $x_{i,t}$ represents the raw sample value of the state variable of class i at time t , μ^i and σ^i represent the mean and standard deviation of this variable within the current edge buffer window, respectively, ε is a small constant to prevent a zero denominator, and $x'_{i,t}$ is the normalized model input.

Table 1. Characteristics and preprocessing methods of multi-source operational status data

Data Source	Measured Variables	Data Format	Acquisition Mode	Edge Preprocessing	Associated Risk
Vibration sensor	Amplitude, spectral energy, impact peak	Continuous time series	High-frequency sampling	Median filtering, window segmentation, normalization	Bearing wear and mechanical looseness
Temperature sensor	Motor, bearing, and ambient temperature	Continuous time series	Periodic sampling	Outlier removal, missing-value imputation, normalization	Overheating and abnormal lubrication
Electrical acquisition module	Current, voltage, and power fluctuation	Continuous time series	Periodic sampling	Timestamp alignment, fluctuation smoothing, feature compression	Abnormal load and electrical surge
PLC control system	Rotational speed, air pressure, cycle time, start/stop state	Discrete and continuous data	Event-triggered and periodic acquisition	Unified status coding and timestamp alignment	Cycle disruption and execution deviation
Alarm log	Alarm code, severity, and duration	Discrete events	Event-triggered recording	Semantic encoding and event merging	Workstation failure and control anomaly
Maintenance record	Repair type, downtime, and spare-part replacement	Structured and textual records	Post-maintenance update	Field structuring and label mapping	Cumulative degradation and recurring fault

3.3. Edge-Based Condition Monitoring

Based on the integration of multi-source operating state data and edge preprocessing, the vibration, temperature, current, rotation speed, cycle time and alarm events of intelligent

production line equipment are all transformed into continuous state matrix. However, this matrix needs to be further organized into time-series samples suitable for real time reasoning at the edge, and the real time state monitoring model is based on a "window construction-feature extraction — light-weight reasoning"[13]. As shown in Figure 2, the CNN-LSTM state feature extraction model captures both local fluctuation characteristics and cross-cycle degradation trends, enabling early anomalies to be identified in complex operating fluctuations; an edge-based lightweight reasoning

mechanism solves local computational constraints and response latency constraints by compressing the model structure, parameter scale, and reasoning workflow, so that state monitoring results can be quickly output within the edge node. Consequently, the real time condition monitoring model uses the results of the front-end data preprocessing, but also provides a stable anomaly score as an input for subsequent health evaluation and maintenance decisions [14].

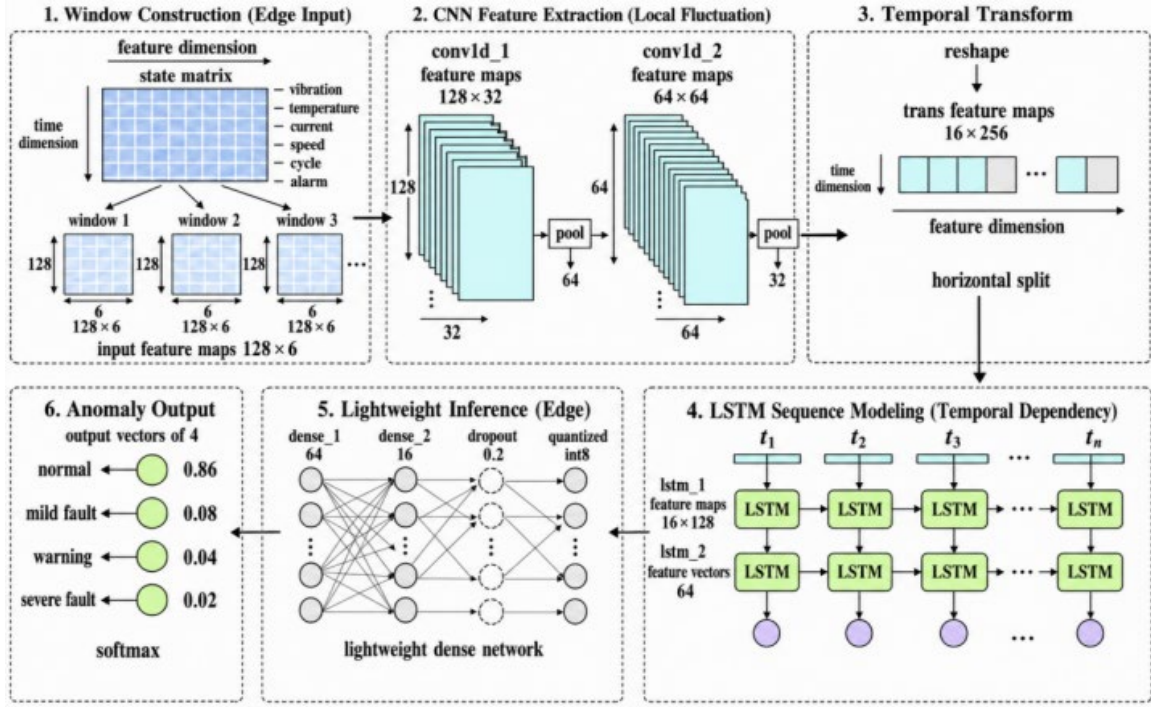


Figure 2. Architecture of the edge-based CNN-LSTM real-time condition monitoring model

3.3.1 Construction of equipment condition sliding windows

A sliding-window mechanism is used to transform the continuous multi-source state stream stored in the edge cache into model samples with a fixed temporal context. This organization reduces the influence of isolated fluctuations and allows short-duration impacts and progressive degradation patterns to be represented within the same input sequence [15]. Let the temporally aligned equipment-state

sequence be $X = \{x_t\}_{t=1}^T$, where $x_t \in R^d$ denotes the d-dimensional state feature vector of the tth aligned frame. The feature vector includes vibration descriptors, temperature, current, rotational speed, air pressure, cycle time, and timestamp-aligned alarm information. A sliding window of length L and step size SSS is used to construct the kth state sample:

$$X_k = [x_{(k-1)SSS+1}, x_{(k-1)SSS+2}, \dots, x_{(k-1)SSS+L}] \in R^{L \times d} \quad (2)$$

Here, L denotes the number of aligned state frames within one window rather than the number of raw samples from an individual sensor, and SSS denotes the number of aligned frames between two adjacent windows. Let T_w , T_s be the temporal window length, T_s be the temporal sliding interval, and Δt be the duration of one aligned state frame. Their conversion relationship is expressed as:

$$L = \frac{T_w}{\Delta t}, \quad S = \frac{T_s}{\Delta t}, \quad N_m = T_w f_m \quad (3)$$

where N_m and f_m denote the number of raw samples and the sampling frequency of data source mmm, respectively. In the experimental implementation, Δt is set to 1 s, while T_w and T_s are set to 20 s and 2 s. Accordingly, each CNN-LSTM input contains $L=20$ aligned state frames,

and adjacent windows are separated by $S=2$ frames. For the vibration channel sampled at 1 kHz, each temporal window initially contains 20,000 raw samples and moves by 2,000 raw samples. The vibration samples within each 1 s frame are transformed into amplitude, spectral-energy, and impact-peak descriptors [16]. Temperature, current, and rotational-speed channels sampled at 1 s intervals provide one observation for each aligned frame, whereas PLC states and alarm events are assigned to the corresponding frames according to their timestamps. This conversion preserves high-frequency impact information while ensuring that heterogeneous variables share a consistent temporal dimension before entering the CNN-LSTM model.

3.3.2 CNN-LSTM state feature extraction model

The CNN-LSTM state feature extraction model takes window samples $W_k \in R^{L \times m}$ as input to extract local fluctuation features and cross-time-period degradation features from multi-source state sequences. Anomalies in intelligent production line devices are usually presented as a combination of vibration shocks, current disturbances, temperature rises, and cycle time fluctuations, and state values at one point of time struggle to reflect the latent degradation processes. As a result, a one-dimension convolutional layer is used to scan the window samples along the time axis. The kernel width is set to 3 or 5 depending on the range of short cycle fluctuations in the device, and the number of convolution channels is 32 – 64, according to edge node memory and inference latency constraints, in order to avoid increased edge-side load due to excessive feature size [17]. The convolutional feature mapping can be expressed as:

$$F_k = \sigma(W_k * \theta_c + b_c)_{(4)}$$

where F_k represents the local state features extracted from the k th window, $*$ denotes the one-dimensional convolution operation, θ_c is the convolution kernel parameter, b_c is the bias term, and σ is the nonlinear activation function. The feature sequence output from CNN is fed into the LSTM layer, where the forget gate, input gate, and output gate maintain the time dependence of the operating state of the device, allowing the model to detect the abnormal trends that are formed by the progressive accumulation of small fluctuations [18]. Depending on the state variable dimension and the edge reasoning cycle, the LSTM hidden units can be set to 64 or 128, and the state representation vector is output via a fully connected layer, which provides a uniform feature input for the classification of the anomaly and health level.

3.3.3 Edge-based lightweight inference mechanism

The edge based lightweight reasoning mechanism solves the problem of the limited computing capacity on the spot, the unstable communication links, and the tight response time

constraints of the intelligent production line. The CNN-LSTM model is deployed to a production line gateway or a workstation-level edge node, which allows a window sample to perform state recognition and anomaly scoring [19]. Based on the following four methods: convolutional channel compression, LSTM hidden unit pruning, weight quantization, and batch scheduling. The number of convolution channels is controlled from 32 to 64 depending on the use of the edge node and the capability of the feature representation. The LSTM hidden unit can be set to 64 or 128 depending on the length of the window and the size of the state. In order to avoid the queue backlog caused by successive window reasoning, an edge node uses a sliding stride and a cache of overlapping segments shared by neighboring windows for reuse, which reduces the amount of redundant convolution calculations [20]. The inference process must satisfy on-site real-time constraints:

$$T_p = T_{pre} + T_{cnn} + T_{lstm} + T_{out} \leq T_s \quad (5)$$

where T_p represents the total inference time for a single window, T_{pre} , T_{cnn} , T_{lstm} , and T_{out} represent the time required for edge preprocessing, convolutional feature extraction, temporal modeling, and anomaly output, respectively, and T_s represents the task trigger cycle corresponding to the sliding window step size.

3.4. Device health status assessment based on anomaly scores

In section 3.3, the edge-side state representation and anomaly output are still characteristic results at the model level, making it difficult to directly support maintenance task prioritization and risk mitigation; they need to be transformed into a device health state assessment result with engineering significance [21]. Anomaly-Based device health evaluation follows a "feature fusion — score computation — classification," which integrates time series characteristics extracted by CNN-LSTM model, the degree of variance in sensor variables, the intensity of alarm events, the load level of the equipment, and the historical maintenance information into a unified evaluation space. Multidimensional state feature fusion eliminates the one-sided impact of individual sensor measures on health assessment, allowing correlation representation of vibration impacts, temperature rise changes, current fluctuations, and cycle anomalies. The anomaly scoring function measures the extent of deviation from normal operation modes and provides a uniform scale for comparing risks between facilities and workstations [22]; health classification and risk assessment maps continuous anomaly scores to actionable status levels, providing decision inputs for maintenance constraint modeling, maintenance priority calculation, and adaptive strategy updates. The evaluation process converts the real-time monitoring results from "model recognition outputs" into "operational decision bases", ensuring that the results of the

edge device sensing can be incorporated into the subsequent maintenance optimization workflow.

3.4.1 Multidimensional state feature fusion

Then, a new method is proposed in which the time-series representation is transformed into a uniform feature vector which is fit for health evaluation. This addresses the problems of distributed equipment anomalies on intelligent production lines, the sensitivity of individual sensor variables to operational disturbances, and the difficulty of directly correlating alarm logs with the semantics of continuous signals [23]. Let the state representation obtained from the CNN-LSTM model for the k th sliding window be h_k , the deviation vector of the sensor variable relative to the normal baseline be d_k , the alarm event code be a_k , the current equipment load level be l_k , and the historical maintenance features be r_k . The fused features can be expressed as:

$$z_k = \phi(W_h h_k + W_d d_k + W_a a_k + W_l l_k + W_r r_k + b) \quad (6)$$

where z_k represents the fusion state feature of the k th window, W_h , W_d , W_a , W_l , and W_r represent the mapping weights of different information sources, respectively, b is the bias term, and ϕ is the nonlinear activation function. The weight parameters are designed according to the sensitivity of the equipment and the reliability of data. The short-term impact and load disturbance are characterized by high-frequency signals, such as vibration and current, while the sustained operating deviation is reflected by temperature, air pressure, and cycle. Alarm events and maintenance records are used to supplement operational semantics which are hard to distinguish from continuous signals. In the process of fusion, edge nodes keep the time synchronization relation between the variables in the same window and carry out sparse coding on the low frequency event characteristics to avoid being overwhelmed by high frequency sensor data.

3.4.2 Design of the anomaly scoring function

The anomaly scoring function is used to convert the resulting fused state feature z_k into a continuous risk metric that is comparable and threshold-based, addressing the issues of inconsistent measurement units across different devices, significant variations in anomaly manifestations, and inconsistent maintenance trigger criteria in smart production lines. The normal operating samples form a relatively stable distribution center in the fusion feature space. The extent to which the current window status differs from the normal center reflects the severity of the device's abnormal operation. Therefore, Mahalanobis distance is used to characterize feature deviation, and a comprehensive anomaly

score is constructed by combining model prediction residuals and alarm event severity:

$$D_k = \sqrt{(z_k - c_0)^T \Sigma_0^{-1} (z_k - c_0)} \quad (7)$$

$$A_k = \text{Sigmoid}(\alpha D_k + \beta E_k + \gamma B_k) \quad (8)$$

where D_k denotes the Mahalanobis distance between the fused feature of the k th window and the normal-state distribution, E_k is the CNN-LSTM prediction residual, B_k is the encoded alarm-event intensity, and A_k is the normalized anomaly score. Before coefficient calibration, the three components are scaled to $[0,1]$ using statistics obtained from the training set. The coefficients are constrained by $\alpha, \beta, \gamma \geq 0$ and $\alpha + \beta + \gamma = 1$. They are calibrated exclusively on the validation set rather than manually assigned or adjusted on the test set. A coarse grid search with an interval of 0.05 is first conducted over the feasible coefficient space, followed by a local refinement with an interval of 0.01 around the best combination. The calibration objective gives greater importance to Macro-F1 because the four health-state classes are imbalanced, while AUC is retained to evaluate score discrimination:

$$J_A = 0.60 F1_{Macro} + 0.40 AUC \quad (9)$$

The coefficient combination that maximizes J_A is selected and fixed before testing. The final values are $\alpha = 0.45$, $\beta = 0.35$, and $\gamma = 0.20$. The relatively larger value of α reflects the primary role of multivariate state deviation, whereas the prediction residual provides complementary evidence of progressive degradation and the alarm term supplies lower-frequency event semantics.

This function integrates state deviation, prediction residuals, and event semantics into a unified evaluation measure, preventing transient disturbances from directly triggering high-risk classifications while increasing the sensitivity to persistent deviations and cumulative alarm evidence.

3.4.3 Health status classification and risk assessment

Health status classification and risk assessment are used to convert anomaly scores A_k into device status labels that can be directly utilized in maintenance decisions, thereby addressing the challenge of mapping continuous scores to specific corrective actions [24]. Device health status is represented by a reverse mapping of the anomaly score, calculated as follows:

$$H_k = 1 - A_k \quad (10)$$

where H_k represents the device health status corresponding to the k th window (k), and A_k represents the normalized anomaly score, with a range of $[0,1]$. A higher anomaly score means a greater deviation from the normal condition and a lower health status. The grade threshold is determined on the basis of the equipment's safety margin, historical failure patterns, process continuity requirements, and false alarm control requirements. The edge nodes perform a smooth evaluation of the health state in successive windows; the hazard label is only activated if the same level

persists for longer than the specified number of windows, thus reducing the variation in quality due to transient noise. Table 2 shows the relationship between the Health Status Interval, Risk Level, and Maintenance Response to Ensure Consistency between Health Assessment Results and Maintenance Priorities. This classification is divided into 4 classes: Normal, Alert, Alert, and Dangerous. Normal equipment is subject to routine monitoring; high frequency sampling is carried out in the Alert state; maintenance schedule is generated in the Alert state; and in Danger status, the Shutdown Check or Mandatory Action Queue.

Table 2. Equipment health levels and risk classification rules

Health Interval	Abnormal Score Range	Status Level	Operational Interpretation	Maintenance Response
$0.85 \leq H_k \leq 1.00$	$0 \leq A_k < 0.15$	Normal	Stable operation without significant deviation	Maintain routine monitoring
$0.70 \leq H_k \leq 0.85$	$0.15 \leq A_k < 0.30$	Watch	Local variables show minor but persistent deviation	Increase sampling and inspection frequency
$0.50 \leq H_k \leq 0.70$	$0.30 \leq A_k < 0.50$	Early warning	Multi-dimensional deviations indicate progressive degradation	Generate and schedule a maintenance task
$0 \leq H_k \leq 0.50$	$0.50 \leq A_k < 1.00$	Danger	Severe anomaly may interrupt continuous production	Perform immediate inspection or shutdown intervention

3.5. Adaptive decision optimization model for maintenance tasks

The health score, anomaly score, and risk level derived from the facility health assessment provide the basis for the maintenance decision. Maintenance tasks on intelligent production lines, however, cannot be carried out solely on the basis of risk. They also have to take into account the constraints of the production line cycle, the downtime window, the stock of spare parts, the scheduling of personnel, the critical equipment, and the utilization of

maintenance resources. The adaptive decision optimization model for maintenance tasks follows the logic of "constraint modeling — priority computation — policy update", which translates the resulting risk evaluation into actionable maintenance action choices. [25] Maintenance Constraint Modeling is used to describe the constraint relationship between Production Continuity and Maintenance Resources, so as to avoid conflicts between the Maintenance Plan and the Production Line Operations. Figure 3 links health state input, constraint matching, priority scoring, maintenance action selection, execution feedback, and strategy update into a closed loop process.

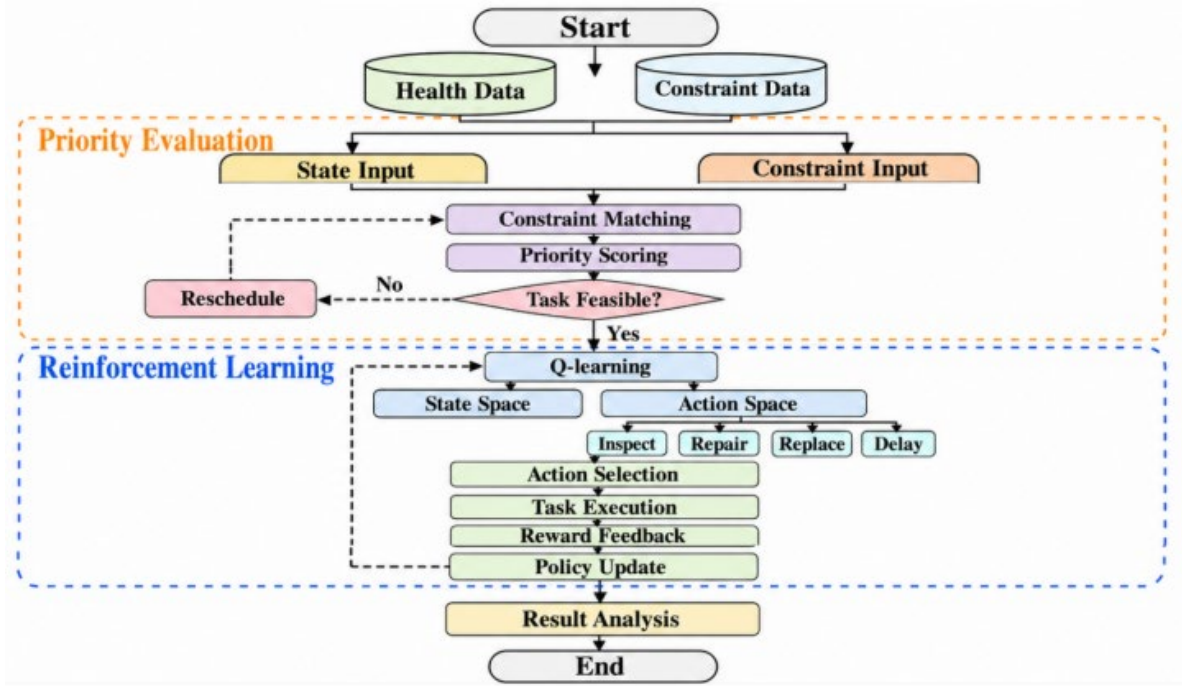


Figure 3. Flowchart of adaptive maintenance decision optimization

3.5.1 Modeling maintenance constraints

The Maintenance Constraint Modeling is used to transform the Facility Health Assessment Results into a standard for determining the feasibility of a Maintenance Task, which

addresses the problem of Intelligent Production Lines in which High Risk Equipment Needs Intervention, but Maintenance Resources, Production Schedules, and Downtime Windows are not Fully Synchronized. Let the decision state of the i th device in the k th monitoring window be $s_{i,k} = \{H_{i,k}, A_{i,k}, G_{i,k}, P_i\}$, where $H_{i,k}$

represents the health score, $A_{i,k}$ represents the anomaly score, $G_{i,k}$ represents the risk level, and P_i represents the device's importance in the production line. The set of maintenance actions is denoted as $M = \{m_1, m_2, \dots, m_n\}$. Each type of maintenance action must satisfy constraints regarding downtime, spare parts resources, labor hours, and production impact. The set of feasible maintenance actions is expressed as:

$$M_{i,k}^* = \{m_j \mid T_j \leq T_{win}, R_j \leq R_{ava}, L_j \leq L_{ava}, C_j \leq C_{max}, S_{i,k} \geq S_{min}\} \quad (11)$$

Here, $M_{i,k}^*$ represents the set of maintenance actions executable for the i th piece of equipment in its current state; T_j denotes the downtime required for maintenance action m_j ; T_{win} denotes the available downtime window; R_j and

R_{ava} denote spare part demand and available spare parts, respectively; L_j and L_{ava} denote labor hour demand and available labor hours, respectively; C_j denotes the cost of the maintenance action's impact on production continuity; C_{max} denotes the upper limit of the allowable cost; and $S_{i,k}$ denotes the current maintenance necessity score; S_{min} denotes the task trigger threshold. Constraint parameters are dynamically set based on equipment criticality, production line scheduling, and maintenance resource status, ensuring that maintenance tasks respond to abnormal risks without disrupting the continuous operation of the production line, thereby providing boundary conditions for subsequent maintenance priority ranking.

3.5.2 Maintenance priority calculation method

The maintenance priority calculation method is used to determine the task execution order within the set of feasible

maintenance actions ($M_{i,k}^*$), ensuring that equipment with deteriorating health status, high criticality at key workstations, and available maintenance resources is prioritized for inclusion in the maintenance queue. For the maintenance task of the i th piece of equipment in the k th monitoring window, the priority depends not only on the anomaly score ($A_{i,k}$) and health status ($H_{i,k}$), but is also influenced by the combined effects of risk level ($G_{i,k}$),

equipment importance (P_i), production line load ($U_{i,k}$), resource availability ($R_{i,k}$), and maintenance impact cost ($C_{i,k}$). The composite priority score is defined as:

$$\Pi_{i,k} = \lambda_1 A_{i,k} + \lambda_2 (1 - H_{i,k}) + \lambda_3 G_{i,k} + \lambda_4 P_i + \lambda_5 U_{i,k} + \lambda_6 R_{i,k} - \lambda_7 C_{i,k} \quad (12)$$

where $\Pi_{i,k}$ represents the maintenance priority of equipment i in monitoring window k . The terms $A_{i,k}$, $1 - H_{i,k}$, $G_{i,k}$, P_i , $U_{i,k}$, $R_{i,k}$, and $C_{i,k}$ denote the anomaly score, health deficit, risk level, workstation importance, production load, resource availability, and maintenance impact cost, respectively. All terms are normalized to $[0,1]$ before priority

calculation. The coefficients satisfy $\lambda_j \geq 0$ and $\sum_{j=1}^7 \lambda_j = 1$. Their values are calibrated on validation-set maintenance tasks using a two-stage constrained search. A coarse search with an interval of 0.05 identifies the candidate region, and a local search with an interval of 0.01 determines the final combination. Candidate coefficient vectors are evaluated using task matching, high-risk resolution, resource utilization, false-alarm rate, response time, and downtime:

$$J_p = 0.30M_{task} + 0.25R_{high} + 0.15U_{res} + 0.10(1 - F_{alarm}) + 0.10(1 - \tilde{T}_{resp}) + 0.10(1 - \tilde{D}_{down}) \quad (13)$$

where M_{task} is the maintenance task matching rate, R_{high} is the high-risk task resolution rate, U_{res} is resource utilization, F_{alarm} is the false-alarm rate, and $\tilde{T}_{resp}$ and $\tilde{D}_{down}$ are the normalized response time and downtime. The coefficient vector maximizing J_p is fixed before independent testing. The final values are $\lambda_1 = 0.22$, $\lambda_2 = 0.18$, $\lambda_3 = 0.16$, $\lambda_4 = 0.16$, $\lambda_5 = 0.10$, $\lambda_6 = 0.12$, and $\lambda_7 = 0.06$. This allocation gives priority to directly observed anomaly severity and health deterioration while retaining explicit contributions from workstation criticality, production conditions, resource feasibility, and maintenance impact cost. To prevent frequent fluctuations in the maintenance queue caused by a single high anomaly score, the score calculation is updated using a rolling window smoothing method:

$$\bar{\Pi}_{i,k} = \rho \Pi_{i,k} + (1 - \rho) \bar{\Pi}_{i,k-1} \quad (14)$$

Here, $\bar{\Pi}_{i,k}$ represents the smoothed priority score, and ρ is the update coefficient, used to balance the current risk changes against the stability of historical tasks. Edge nodes sort candidate maintenance tasks based on $\bar{\Pi}_{i,k}$ and pass high-priority tasks to the reinforcement learning strategy update module, providing a computable ranking basis for maintenance action selection.

3.5.3 Reinforcement Learning Strategy Updating

The maintenance strategy is formulated as a finite Markov decision process and solved using tabular Q-learning. This algorithm is selected instead of Deep Q-Network because the maintenance state is compact after discretization, the action space contains only a limited number of executable maintenance actions, and direct state-action value storage reduces the computational and memory requirements of edge deployment. At decision time t , the five-dimensional state is represented by $s_t = [\bar{P}_t, H_t, G_t, R_t, C_t]$, where $\bar{P}_t$ is the smoothed maintenance priority normalized to $[0,1]$, $H_t \in [0,1]$ is the equipment health score, and $G_t \in \{0, 1/3, 2/3, 1\}$ represents the Normal, Watch, Early Warning, and Danger levels, respectively. $R_t \in [0,1]$ denotes the comprehensive satisfaction of spare parts, labor hours, and available downtime windows, whereas $C_t \in [0,1]$ denotes the normalized production impact cost.

To construct a finite Q-table, $\bar{P}_t$, H_t , R_t , and C_t are each divided into five equal-width intervals, while G_t retains its four discrete levels, producing $5 \times 5 \times 4 \times 5 \times 5 = 2,500$ discrete state combinations.

The action space is defined as $A = \{a_0, a_1, a_2, a_3, a_4\}$, where a_0 maintains routine monitoring, a_1 increases the sampling and inspection frequency, a_2 schedules planned maintenance within an available downtime window, a_3 triggers immediate repair, and a_4 initiates-controlled shutdown and mandatory inspection. The Q-learning policy does not select directly from the complete action space. For each equipment state, the feasible subset $A_t^{feas} \subseteq A$ is first determined using the downtime, spare-part, labor-hour, maintenance-necessity, and production-impact constraints defined in Equation (10). The action with the highest state-action value is then selected only from this feasible subset.

The immediate reward balances equipment-risk reduction, response timeliness, downtime loss, and maintenance cost. All reward components are normalized before calculation:

$$r_t = 0.40\Delta R_t + 0.20(1 - \tilde{T}_t) - 0.25\tilde{D}_t - 0.15\tilde{C}_t \quad (15)$$

where $\Delta R_t = R_t - R_{t+1} \in [-1,1]$ denotes the reduction in equipment failure risk after executing the maintenance action; $\tilde{T}_t \in [0,1]$ is the normalized task response delay; $\tilde{D}_t \in [0,1]$ is the normalized downtime loss; and $\tilde{C}_t \in [0,1]$ is the normalized maintenance and task-switching cost. A positive risk reduction and timely task completion increase the reward, whereas prolonged downtime and excessive maintenance cost decrease it. The larger coefficient assigned to risk reduction ensures that equipment safety and

production continuity dominate the policy objective. The state–action value is updated using tabular Q-learning:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \eta \left[r_t + \gamma \max_{a' \in A_{t+1}} Q(s_{t+1}, a') - Q(s_t, a_t) \right] \quad (16)$$

where $Q(s_t, a_t)$ is the expected cumulative return of action a_t in state s_t , η is the learning rate, and γ is the discount factor. The policy is first pretrained offline using historical equipment states, maintenance work orders, execution results, downtime records, and resource-consumption records from the training partition. During offline training, an ε -greedy strategy is adopted, with ε gradually reduced from 1.00 to 0.05, the learning rate set to 0.10, and the discount factor set to 0.90. The validation partition is used to select the reward coefficients and determine convergence without accessing the test partition.

After deployment, the edge node applies a greedy policy to the current feasible action set and does not conduct random exploration on the production line. Confirmed state–action–reward–next-state tuples are periodically uploaded to the cloud together with maintenance completion and false-alarm records. The cloud performs conservative online fine-tuning every 12 h using a reduced learning rate of 0.02 and distributes the validated Q-table to the edge node. This offline-pretraining and online-fine-tuning procedure enables the strategy to adapt to changes in degradation patterns and maintenance resources while preventing unverified exploratory actions from disrupting production.

3.6 Edge-Cloud Feedback and Model Updating

The edge-cloud collaborative feedback and dynamic model update mechanism is designed to keep the monitoring model, health threshold, and maintenance policy consistent with changing manufacturing conditions. In the forward process, the edge node completes state-window generation, lightweight CNN-LSTM inference, anomaly score calculation, health-level mapping, and maintenance action triggering. In the feedback process, the maintenance decision module returns execution results, delayed fault confirmation, false-alarm records, and resource changes to the edge buffer. The edge uploads only fused feature summaries, anomaly segment indexes, health-level variations, maintenance execution results, and false-positive feedback, avoiding the transmission of high-frequency raw data. The cloud aggregates cross-shift and cross-equipment samples, recalibrates CNN-LSTM parameters, updates health thresholds, and adjusts reinforcement learning policy parameters. Model parameters are updated using an incremental fusion approach:

$$\theta_e^{t+1} = (1 - \lambda) \theta_e^t + \lambda \theta_c^t \quad (17)$$

where θ_e^t represents the current model parameters at the edge, θ_c^t represents the parameters calibrated in the cloud based on new samples, λ is the update fusion coefficient used to control the influence of new parameters on the edge model, and θ_e^{t+1} represents the updated edge model parameters. Based on the confidence of the sample, the change of the operation condition, and the false positive feedback from the edge, the fusion coefficient is set dynamically, so as to avoid the excessive bias of the model.

4. Experimental design and results analysis

4.1 Experimental environment and parameter settings

CNC processing units, industrial robots, transfer units, motor drive modules, and PLC control stations are selected for verification from a separate manufacturing facility. Data sources include real time sensor data, PLC operations log, edge gateway cache log, alarm event log, and maintenance work order book. The data is collected between October 2024 and March 2025 and includes operating conditions, such as routine manufacturing, switching and commissioning, minor anomalies, planned maintenance, and unplanned stoppages. The sampling frequency of the vibration signal is 1 kHz, and the sampling interval for the temperature, the current, and the rotation rate is 1 s. An event triggered approach was used to record the PLC state and the alarm log, and the maintenance record was structured according to workorder completion times. After time alignment, missing value imputation, abnormal noise filtering, and sliding window segmentation, a total of 48,600 status windows were generated, of which 34,200 normal samples, 8200 attention needs, 4,300 alerts, and 1,900 dangerous samples. The complete dataset was evaluated through 10 repeated stratified experiments. In each experiment, the 48,600 status windows were divided into training, validation, and test subsets at a ratio of 70%, 10%, and 20%, respectively. Stratification was performed according to the four equipment-health classes to maintain comparable class distributions across different runs. The same random split was used by all competing methods within each run, thereby enabling paired statistical comparisons and preventing differences in sample composition from affecting the performance comparison. Different random seeds were used for data partitioning, model initialization, and training in the 10 runs.

For each run, the validation subset contained 4,860 status windows and was used exclusively for model selection, anomaly-score coefficient calibration, maintenance-priority weight calibration, and reinforcement-learning parameter selection. The corresponding independent test subset contained 9,720 status windows and was excluded from all

parameter adjustments. The normalization parameters in Equations (7) and (11) were estimated only from the training subset of the current run. All selected parameters were fixed before evaluation on the corresponding test subset. The experimental environment, dataset configuration, model parameters, and parameter-setting rationales are summarized in Table 3.

The condition-monitoring and maintenance-decision metrics are reported as the mean and standard deviation over the 10 repeated experiments. The F1-score and maintenance task matching rate were selected as the key statistical

indicators because they directly represent health-state classification performance and maintenance-task allocation effectiveness. Two-sided paired t-tests were conducted between the proposed method and each baseline using results obtained from matched experimental runs. Holm-Bonferroni correction was applied separately to the comparisons subsequently presented in Tables 4 and 5, and an adjusted probability value of $p_{adj} < 0.05$ was regarded as statistically significant.

Table 3. Experimental environment and parameter settings

Category	Item	Setting	Rationale
Data	Equipment	CNC machining units, industrial robots, conveying units, motor drives, and PLC workstations	Covers the principal operational stages of the production line
Data	Collection period	October 2024–March 2025	Includes continuous production and maintenance cycles
Data	Total status windows	48,600	Obtained after time alignment and sliding-window segmentation
Data	Training/validation/test split	70%/10%/20%	Separates model fitting, parameter selection, and independent testing
Edge environment	Processor and memory	8-core CPU; 16 GB RAM	Supports lightweight on-site inference
Cloud environment	Processor and memory	NVIDIA RTX 3090 GPU; 64 GB RAM	Supports model training and parameter calibration
Window construction	Window length / step	20 s / 2 s	Balances anomaly coverage and inference latency
CNN	Kernel size / channels	3 / 64	Extracts short-term local fluctuations
LSTM	Hidden units	128	Represents temporal dependence and degradation evolution
Training	Learning rate / batch size	0.001 / 64	Maintains convergence stability
Reinforcement learning	Discount factor	0.90	Balances immediate maintenance gains and long-term effects

4.2 Condition Monitoring Performance

The real time condition monitoring performance analysis uses 9720 sets of condition window samples from the test set to verify the recognition results of the four conditions: normal, alert, alert, and dangerous. In this paper, a comparison is made between SVM, Random Forest, LSTM, CNN-LSTM, and Edge Collaborative CNN-LSTM models. Table 4 shows the differences between the two models. Based on the single variable alarm threshold, the threshold approach is capable of detecting apparent anomalies, such as temperature and current fluctuations, but is not sufficiently sensitive to the initial degradation state resulting from the superposition of vibration impacts and

rhythmic disturbances; SVM and Random Forest can make use of multidimensional engineered features to improve classification stability, but it is difficult to capture temporal evolutionary relationships within continuous windows; LSTM can capture trends in state changes, but do not sufficiently extract features of local abrupt changes; CNN-LSTM simultaneously uses convolution layers to extract short term local fluctuations, and uses LSTM layers to maintain cross window state memory, resulting in more stable performance in early warning and dangerous status identification. The Edge Collaborative CNN-LSTM introduces front-end preprocessing, sliding window caching, and lightweight reasoning mechanism, and achieves a FR

score of 0.946 and AUC of 0.973, with a mean latency of 126 ms.

Table 4. Comparison of performance among different condition monitoring models

Model	Accuracy ↑	Precision ↑	Recall ↑	F1-score ↑	AUC ↑	Latency (ms) ↓	P _{adj} for F1
Traditional threshold	0.821pm ± 0.010	0.806pm ± 0.012	0.784pm ± 0.014	0.795pm ± 0.009	0.842pm ± 0.009	58pm4	<0.001
SVM	0.867pm ± 0.008	0.854pm ± 0.010	0.831pm ± 0.011	0.842pm ± 0.008	0.889pm ± 0.008	214pm9	<0.001
Random forest	0.884pm ± 0.007	0.872pm ± 0.009	0.858pm ± 0.010	0.865pm ± 0.007	0.907pm ± 0.007	236pm11	<0.001
LSTM	0.913pm ± 0.006	0.901pm ± 0.007	0.894pm ± 0.008	0.897pm ± 0.006	0.941pm ± 0.005	168pm7	<0.001
CNN-LSTM	0.936pm ± 0.005	0.928pm ± 0.006	0.919pm ± 0.006	0.925pm ± 0.005	0.961pm ± 0.004	152pm6	<0.001
Edge-collaborative CNN-LSTM	0.957pm ± 0.004	0.951pm ± 0.004	0.941pm ± 0.005	0.946pm ± 0.003	0.973pm ± 0.003	126pm5	—

4.3 Validation of adaptive maintenance decision-making

The validation of the Adaptive Maintenance Decision Model is based on a sample of the Concern, Warning, and Danger State that are identified in the Test Set. It uses equipment health, anomaly score, risk level, equipment significance, production line load, spare parts stock, and man-hours as decision inputs to compare task handling performance of manual experience maintenance, fixed cycle maintenance, threshold triggered maintenance, predictive maintenance, and the proposed adaptive maintenance decision model. Table 5 shows the results for each policy on maintenance task matching rate, average response time, false alarm rate,

downtime, resource utilization, and high risk priority processing rate. Manual experience-based maintenance relies on the judgment of personnel on the spot and is able to deal with apparent failures, but does not respond well to early deterioration; fixed cycle maintenance is not aware of the real state of health in the fixed cycle, which results in excessive maintenance of low risk devices and delays in the handling of high risk devices; the threshold triggered maintenance is highly sensitive to single point anomalies, which leads to relatively high false alarm rates; predictive maintenance can use condition trends to improve the scheduling of tasks, but does not take sufficient account of downtime windows, spare parts resources, and workstation importance. The proposed model integrates anomaly scores, health status, equipment importance, and resource constraints into maintenance priority calculations. It uses reinforcement learning to update the action selection based on maintenance feedback, and achieves 94.1% maintenance task matching rate, the average response time is reduced to 18.6 minutes, and the high-risk tasks are given priority. This shows that the adaptive decision mechanism is able to translate the status monitoring results into a maintenance execution sequence, thus improving the alignment of maintenance tasks with production constraints.

Table 5. Comparison of performance among different maintenance decision-making methods

Maintenance method	Task matching (%) ↑	Response time (min) ↓	False alarm rate (%) ↓	Downtime (h) ↓	Resource utilization (%) ↑	High-risk resolution (%) ↑	P _{adj} for matching rate
Manual maintenance	78.4pm ± 1.6	42.8pm ± 3.1	12.6pm ± 1.2	38.5pm ± 2.8	71.2pm ± 2.0	80.7pm ± 1.9	<0.001
Scheduled maintenance	74.9pm ± 1.8	56.3pm ± 3.7	9.8pm ± 1.0	44.2pm ± 3.1	68.5pm ± 2.3	73.4pm ± 2.1	<0.001
Threshold-triggered maintenance	82.6pm ± 1.4	31.7pm ± 2.6	15.4pm ± 1.3	33.6pm ± 2.5	75.8pm ± 1.8	84.9pm ± 1.6	<0.001
Predictive maintenance	89.3pm ± 1.1	24.5pm ± 1.9	8.7pm ± 0.8	27.4pm ± 1.9	82.6pm ± 1.5	90.8pm ± 1.2	<0.001

Adaptive maintenance decision	94.1 pm ± 0.7	18.6 pm ± 1.4	5.9 pm ± 0.6	21.8 pm ± 1.6	88.9 pm ± 1.2	96.3 pm ± 0.9	—
-------------------------------	------------------------	------------------------	-----------------------	------------------------	------------------------	------------------------	---

4.4 Analysis of edge-cloud collaborative operation and maintenance efficiency

In this paper, we compare three deployment modes — cloud, edge, and edge-cloud — to prove that the dynamic model update mechanism described in section 3.6 can be used to support real time on the spot and global model calibration. As illustrated in Figure 4, the centralized cloud mode uploads full state sequences and log data, resulting in 286 MB of data transmission per unit time and an average response latency of 684 ms, which makes it difficult to meet the abnormal-response requirement of high-cycle manufacturing workstations. The edge-cloud independent processing mode

reduces the average response latency to 112 ms and the data upload volume to 38MB; however, because the model updates depend on a local sample, its calibration ability is inadequate when confronted with operational mode changes and changes in equipment degradation profile; the edge-cloud collaborative model only uploads the integrated feature summary, anomaly segment index, and maintenance feedback record, limiting the upload volume to 64MB per unit time with an average response latency of 138 ms. The model update cycle has been reduced from 72 hours (for manual offline updates) to 12 hours. The results show that the edge is able to deal with real time reasoning, anomaly scoring, and maintenance action triggering, thus reducing site response latency; the cloud handles sample aggregation, parameter adjustment, and policy synchronization, thus alleviating the problem of inadequate sample coverage at edge nodes. The collaborative mechanism balances transmission load, response rate, and continuous model updates, and provides a more stable project deployment method for intelligent production lines to run and maintain.

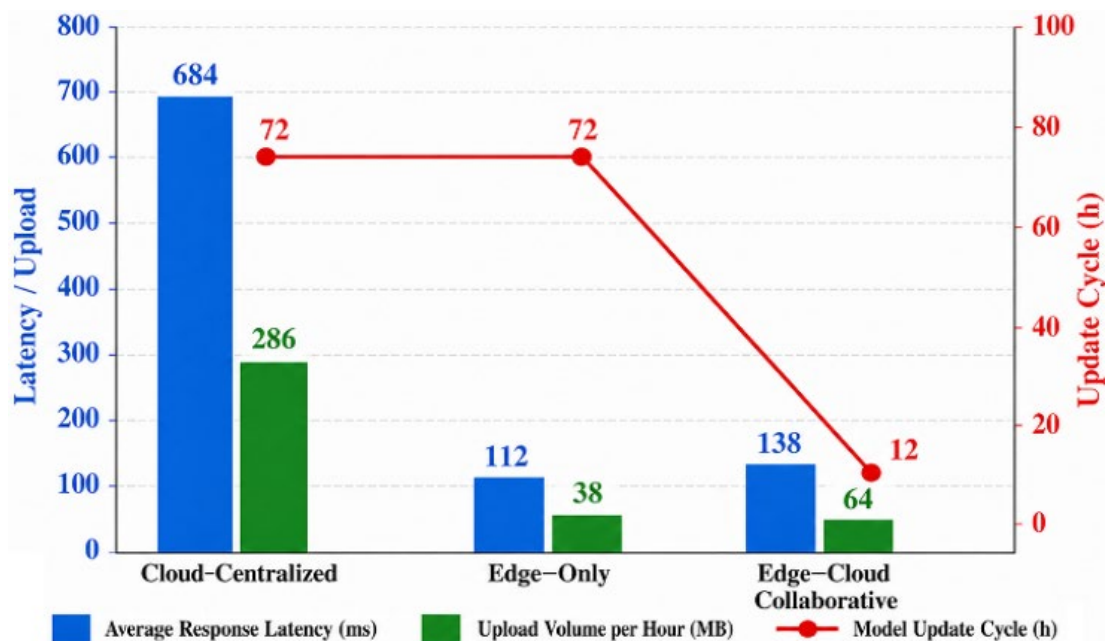


Figure 4. Comparison of edge-cloud collaborative operation and maintenance efficiency under different deployment modes

4.5 Ablation experiments and robustness analysis

The ablation experiments were conducted with the complete model as the baseline, and the key modules were removed in sequence, and the model stability was tested under 5% noise injection, 10% loss of samples, and 100 ms network latency. Figure 5 illustrates the variation of F1-SCORE, maintenance

decision-making precision, and mean response times for different ablation models. After removing the edge preprocessing module, the F1-SCORE was directly fed into the status window, causing the F1 score to fall to 0.902; the removal of the CNN module resulted in insufficient extraction of the local abnormal characteristics, such as vibration shocks, current disturbances, and so on, causing the

F1 score to fall to 0.915; the removal of the LSTM module reduced the capacity of the model to characterize the continuous degradation trend, and the stability of the early warning status recognition declined; the removal of the anomaly rating and health evaluation module resulted in a lack of a uniform risk metric for the maintenance task prioritization, and the maintenance decision precision fell to 0.887; the maintenance actions were unable to adapt to the

change in resources and load after the removal of the reinforcement learning strategy update module. The robust test results show that the proposed approach is able to maintain relatively stable monitoring and decision making even when data quality is degraded, operational status transitions and communication delays are affected.

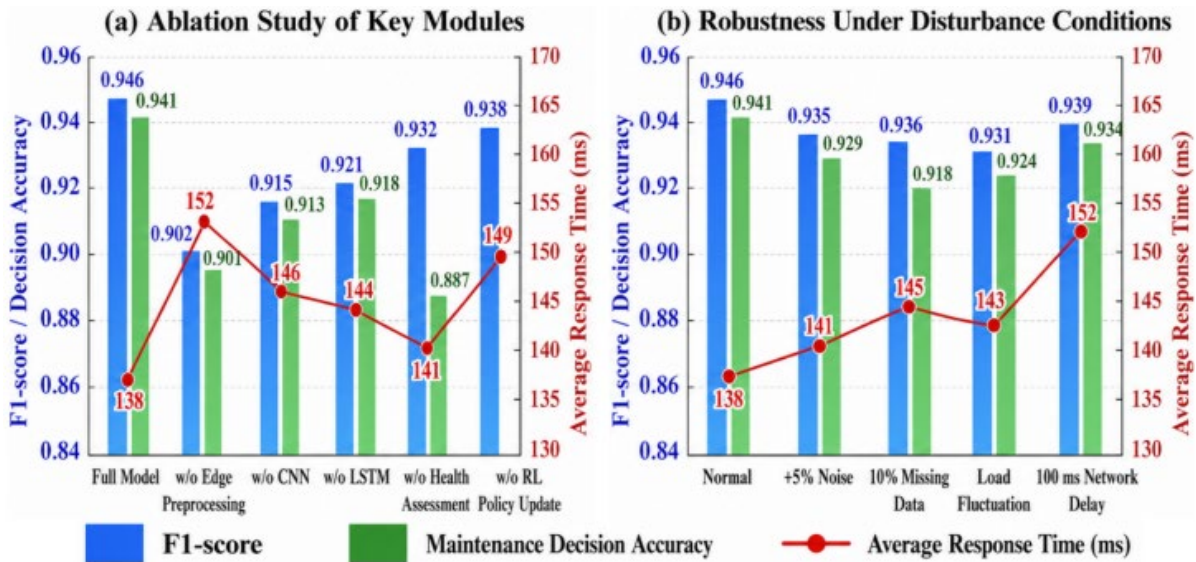


Figure 5. Comparison of ablation experiment and robustness test results

5. Conclusion

The experimental results demonstrate that the proposed method improves the coordination of equipment-condition monitoring, risk assessment, and maintenance decision-making. Through edge preprocessing, sliding-window organization, and lightweight CNN-LSTM inference of multi-source operational data, the condition-recognition F1-score reaches (0.946 $\pm$ 0.003), the AUC reaches (0.973 $\pm$ 0.003), and the average detection latency is (126 $\pm$ 5) ms. These results indicate that local impact features and progressive temporal degradation can be jointly represented under edge-computing constraints. Anomaly scoring, health mapping, and maintenance-priority calculation establish a quantitative transformation from model outputs to executable O&M actions. The adaptive maintenance decision model achieves a task matching rate of (94.1% $\pm$ 0.7%) and a high-risk task resolution rate of (96.3% $\pm$ 0.9%). The edge-cloud collaborative mechanism further balances response latency, data-upload volume, and model-update frequency.

The current method still has several identifiable limitations. The experimental data were collected from one manufacturing facility and covered a restricted set of equipment types and labeled fault conditions, so its transferability to production lines with different takt patterns, sensor configurations, and unseen faults has not been fully

established. The fixed 20 s sliding window may not optimally represent degradation processes with different temporal scales, while the discretized Q-learning policy is limited by its predefined state and action spaces. Moreover, the 12 h cloud-update interval may respond slowly to abrupt operating-condition drift. Future work will conduct cross-facility and leave-one-equipment-out validation, introduce open-set fault recognition, uncertainty-aware health assessment, and adaptive multi-scale windows, and investigate safe constrained reinforcement learning for larger maintenance state spaces. Drift-triggered edge-cloud updating and federated parameter calibration will also be examined to improve adaptation without transmitting high-frequency raw data.

References

- [1] Bala A, Rashid R Z J A, Ismail I, et al. Artificial intelligence and edge computing for machine maintenance-review[J]. Artificial Intelligence Review, 2024, 57(5): 119.
- [2] Somu N, Dasappa N S. An edge-cloud IIoT framework for predictive maintenance in manufacturing systems[J]. Advanced Engineering Informatics, 2025, 65: 103388.
- [3] Sathupadi K, Achar S, Bhaskaran S V, et al. Edge-cloud synergy for AI-enhanced sensor network data: A real-time predictive maintenance framework[J]. Sensors, 2024, 24(24): 7918.

- [4] Abdullahi I, Longo S, Samie M. Towards a distributed digital twin framework for predictive maintenance in industrial internet of things (IIoT)[J]. *Sensors*, 2024, 24(8): 2663.
- [5] Mennilli R, Mazza L, Mura A. Integrating machine learning for predictive maintenance on resource-constrained PLCs: A feasibility study[J]. *Sensors*, 2025, 25(2): 537.
- [6] Artiushenko V, Lang S, Lerez C, et al. Resource-efficient Edge AI solution for predictive maintenance[J]. *Procedia Computer Science*, 2024, 232: 348-357.
- [7] Shoorkand H D, Nourelfath M, Hajji A. A hybrid CNN-LSTM model for joint optimization of production and imperfect predictive maintenance planning[J]. *Reliability Engineering & System Safety*, 2024, 241: 109707.
- [8] Varalakshmi K, Kumar J. Optimized predictive maintenance for streaming data in industrial IoT networks using deep reinforcement learning and ensemble techniques[J]. *Scientific Reports*, 2025, 15(1): 27201.
- [9] Garcia A, Oregui X, Franco J, et al. Time series manufacturing data edge monitoring and visualization to support industrial maintenance teams[J]. *SN Computer Science*, 2023, 5(1): 131.
- [10] Khan T, Khan U, Khan A, et al. Data-driven digital twin framework for predictive maintenance of smart manufacturing systems[J]. *Machines*, 2025, 13(6): 481.
- [11] Visconti P, Rausa G, Del-Valle-Soto C, et al. Machine learning and IoT-based solutions in industrial applications for smart manufacturing: a critical review[J]. *Future Internet*, 2024, 16(11): 394.
- [12] Bitam T, Yahiaoui A, Boubiche D E, et al. Artificial intelligence of things for next-generation predictive maintenance[J]. *Sensors (Basel, Switzerland)*, 2025, 25(24): 7636.
- [13] Rakholia R, Suárez-Cetrulo A L, Singh M, et al. Integrating AI and IoT for predictive maintenance in Industry 4.0 manufacturing environments: A practical approach[J]. *Information*, 2025, 16(9): 737.
- [14] Pekşen M F, Yurtsever U, Uyaroğlu Y. Enhancing electrical panel anomaly detection for predictive maintenance with machine learning and IoT[J]. *Alexandria Engineering Journal*, 2024, 96: 112-123.
- [15] Dritsas E, Trigka M. A survey on the applications of cloud computing in the industrial internet of things[J]. *Big data and cognitive computing*, 2025, 9(2): 44.
- [16] Rahman M S, Ghosh T, Aurna N F, et al. Machine learning and internet of things in industry 4.0: A review[J]. *Measurement: Sensors*, 2023, 28: 100822.
- [17] Aboshosha A, Haggag A, George N, et al. IoT-based data-driven predictive maintenance relying on fuzzy system and artificial neural networks[J]. *Scientific reports*, 2023, 13(1): 12186.
- [18] Jan Z, Ahamed F, Mayer W, et al. Artificial intelligence for industry 4.0: Systematic review of applications, challenges, and opportunities[J]. *Expert systems with applications*, 2023, 216: 119456.
- [19] Wang L, Li B, Zhao X. Multi-objective predictive maintenance scheduling models integrating remaining useful life prediction and maintenance decisions[J]. *Computers & Industrial Engineering*, 2024, 197: 110581.
- [20] Chen J, Zhou X. Reinforcement learning based maintenance scheduling of flexible multi-machine manufacturing systems with varying interactive degradation[J]. *Reliability Engineering & System Safety*, 2025, 260: 111018.
- [21] Geurtsen M, Leenen C, Adan I, et al. Deep reinforcement learning for optimal planning of production line maintenance with deterioration[J]. *Reliability Engineering & System Safety*, 2026, 266(pd): 111767.
- [22] Palma G, Cecchi G, Rizzo A. Large language models for predictive maintenance in the leather tanning industry: Multimodal anomaly detection in compressors[J]. *Electronics*, 2025, 14(10): 2061.
- [23] Zokirov J, Khurazov G, Temirova M, et al. Deep Learning Enhanced Predictive Maintenance Framework Using Industrial Internet of Things Sensors for Smart Manufacturing Systems[J]. *International Journal of Industrial Engineering and Management*, 2025, 16(4): 389-410.
- [24] Farooq U, Ademola M, Shaalan A. Comparative analysis of machine learning models for predictive maintenance of ball bearing systems[J]. *Electronics*, 2024, 13(2): 438.
- [25] Rojek I, Jasiulewicz-Kaczmarek M, Piechowski M, et al. An artificial intelligence approach for improving maintenance to supervise machine failures and support their repair[J]. *Applied Sciences*, 2023, 13(8): 4971.