ingegneria sismica
International Journal of Earthquake Engineering
Anno XLIII - Num. 1 - 2026

Lightweight CNN supports aero-engine cloud-edge fusion health monitoring and real-time diagnosis research

ChinHsiung Lee[1,*] *and LiangHong Lin*[2]

[1] School of General Aviation Industry, Fujian Chuanzheng Communications College, Fuzhou 350007, China

[2] Rajamangala University of Technology Krungthep, Internation College, Bangkok 10120, Thailand

SUMMARY: *This paper proposes a cloud-edge collaborative framework supported by a lightweight CNN for aero-engine health monitoring and real-time diagnosis. In this framework, multi-sensor streams are organized into synchronous time Windows, and compressed convolutional models are deployed at edge nodes to achieve low-latency state inference, while parameter aggregation, threshold calibration and model update are performed in the cloud. In order to enhance the stability of diagnosis, the inference backbone jointly introduces depth-wise separable convolution, channel recalibation and residual feature reuse mechanisms. The experiment is carried out on 16800 labeled samples, covering four types of states: normal, degradation, surge precursor and fault. The results show that the proposed method achieves 94.7% accuracy and 0.931 F1-score, the inference delay of the single window on the edge side is 21 ms, and the model size is 3.6 MB. Under the condition of cloud-edge collaborative operation, the alarm consistency reaches 92.4%, while maintaining good online diagnosis response ability. Experimental results verify the computational efficiency, deployment suitability, and diagnostic reliability of the proposed framework in aero-engine monitoring tasks.*

KEYWORDS: *Lightweight CNN; Aero-engine; Cloud-edge fusion; Real-time diagnosis*

1 Introduction

As the core equipment of the flight propulsion system, the operation state of the aero-engine is directly related to the flight safety and maintenance scheduling. With the increasing density of airborne sensors, the popularity of edge acquisition terminals and the expansion of ground computing platforms, health monitoring data has shown the characteristics of multi-source, continuous and heterogeneous coexistence. Temperature, pressure, vibration, speed and oil parameters form coupling trajectories under different working conditions. It is difficult to support real-time state recognition and online diagnosis by simply relying on threshold discrimination or offline manual analysis. To establish an intelligent monitoring method for aero-engine, which takes into account computational efficiency, deployment feasibility and diagnostic accuracy, has become a research focus in computer modeling and cloud-edge collaborative applications, and has strong engineering deployment capabilities and practical task scheduling support value.

The development of deep learning provides a technical path for aero-engine state modeling.

*18960800120@163.com
https://doi.org/10.65102/is2026225 Published: 30 April, 2026

Liu et al. [1] constructed a dual attention remaining useful life estimation architecture, which combined data representation with time-series degradation modeling. Li et al. [2] proposed a CNN-LSTM model fused with attention mechanism, which used convolution to extract local patterns and used a time series module to describe the degradation evolution. Peng et al. [3] realize the remaining life prediction of turbofan engine through deep feature extraction and fusion mechanism. Al-khazraji et Al. [4] proposed a hybrid model combining autoencoder and deep belief network to introduce nonlinear representation learning into aero-engine life estimation. Mao et al. [5] used differentiable architecture search for engine lifetime modeling, demonstrating the computational potential of automatic structure design in complex diagnostic tasks. Wang et al. [6] constructed a parallel semi-supervised model to deal with labeling constraints in degradation prediction, so that the learning process under the condition of limited labels remained stable. Xiang et al. [7] proposed an automatic multiple differential deep learning method to map high-dimensional dynamic information into the remaining useful life prediction task. Song et al. [8] designed a hierarchical long short-term memory network scheme to strengthen the expression ability of multi-stage degradation sequences. Li et al. [9] fuse knowledge information with deep learning model for aero-engine life estimation. Chen et al. [10] proposed an adaptive multi-modal fusion and clustering ensemble transfer regression method to enhance the adaptability of cross-condition life prediction.

The above research lays a foundation for the intelligent diagnosis of aero-engine, and also shows that deep network, feature fusion and transfer modeling have strong performance in complex condition recognition. At the same time, aero-engine health monitoring is not limited to life estimation, but also involves state perception, anomaly detection, side-end response, cloud update and collaborative diagnosis loop closure. In the face of continuous incoming monitoring flow, the model not only needs to maintain the recognition ability, but also needs to control the parameter scale, compress the inference delay and adapt to the resource-constrained devices. Lightweight CNN has advantages in local pattern extraction, convolution calculation multiplexing and structure compression, which is suitable for deployment on the edge side to complete state judgment. The cloud can undertake the tasks of parameter aggregation, model correction, sample training and policy update. Combining the lightweight CNN with the cloud-edge fusion mechanism, the edge response and cloud analysis can be connected as an integrated computing link, thus forming an implementation framework for aero-engine health monitoring and real-time diagnosis.

Based on the above background, this paper focuses on the cloud-edge fusion health monitoring and real-time diagnosis of aero-engine supported by lightweight CNN, and the specific research objectives are as follows:

(1) A lightweight CNN diagnosis model for aeroengine monitoring data is constructed to improve the computational efficiency of edge side state recognition and anomaly discrimination.

(2) The aero-engine health monitoring and real-time diagnosis module is designed, and the processes of monitoring data processing, state reasoning, alarm triggering and result output are established.

(3) The cloud-edge fusion collaboration mechanism is integrated to complete the model update, diagnostic back transmission and online collaborative analysis, and the performance of the proposed method in terms of real-time performance, stability and deployment adaptability is verified.

2 Related work

In the field of aero-engine health monitoring and real-time diagnosis, existing achievements

can be roughly divided into three categories: (1) remaining useful life prediction methods for degradation process modeling; (2) Deep learning health monitoring method for complex working condition perception; (3) cloud-edge collaborative diagnosis framework for online deployment. The above studies have promoted aero-engine condition recognition from single off-line analysis to data-driven modeling, but there are still obvious differences in edge-side real-time reasoning, model compression adaptation, multi-source monitoring cooperation and online update closed-loop. Therefore, this paper summarizes the related work into three directions: computing model, monitoring mechanism and collaborative deployment, and establishes the cloud-edge fusion health monitoring and real-time diagnosis route supported by lightweight CNN. From the perspective of computational structure, the aero-engine monitoring task is not the same as the general classification task. Sensor sampling frequency, condition switching density and state label granularity jointly determine that the model should not only maintain the ability to capture local abnormal textures, but also ensure the response stability under long-term operation. Although strong representations can be obtained by only relying on high-complexity networks, model compression, inference scheduling and result transmission also constitute an important part of the system implementation in the deployment environment where edge nodes are limited in storage, computing power and communication bandwidth. Therefore, related work not only needs to compare the prediction error and classification index, but also needs to investigate the model volume, online delay, edge-cloud division method and continuous update ability. Only when the algorithm performance and system deployability are taken into consideration, can the computational value of real-time diagnosis research of aero-engine be more accurately reflected. It also constitutes one of the direct starting points and calculation basis for the related work summarized in this paper.

Xu et al. [11] studied the application of global attention mechanism in the remaining life prediction of aero-engine, and proposed a prediction structure based on deep feature weighting to make the key degradation sections obtain higher response in the sequence representation. Liu et al. [12] studied the modeling method of double difference network in engine life estimation, and proposed to jointly constrain the degradation representation through distribution difference and feature difference to enhance the cross-sample discrimination ability. Hu et al. [13] studied the probabilistic life prediction method under degraded data, and proposed an improved Bayesian uncertainty estimation mechanism, so that the model output not only contains the life results, but also contains the probability confidence information. Miao et al. [14] studied the implementation of cloud-edge system in aircraft fuel pump fault diagnosis, and proposed a bio-heuristic diagnosis process to connect edge acquisition and cloud analysis. Zha et al. [15] studied the combination of feature selection and improved TCN in aero-engine life prediction, and proposed to enhance the modeling path of degradation mode extraction by time series convolution. Liu et al. [16] studied the multi-scale transfer learning method under the condition of finite samples, and proposed the cross-domain feature transfer and multi-scale representation fusion framework to improve the stability of lifetime estimation under the condition of small samples.

Although existing research has formed a rich accumulation of technologies in life prediction, degradation estimation and online sensing, as shown in Table 1, most of the existing methods focus on high-precision modeling in the cloud, and the coordination degree between model scale, inference delay and edge device adaptation is not consistent. Some studies can enhance the degradation expression or supplement the uncertainty estimation, but few directly focus on the continuous closed loop of "monitor-diagnosis-back-update". Other research has begun to touch on cloud-edge collaboration, but there is still room for further expansion in the unified modeling of multi-source monitoring data and the integration of lightweight diagnostic modules for aero-engine.

Table 1: Work arrangement related to aero-engine health monitoring

Study	Method Type	Data or Scenario	Main Metrics	Applicability Characteristics
[11]	Global Attention-Based Deep Learning	Aero-engine life prediction	Prediction accuracy, error	Strengthens the representation of key degradation stages
[12]	Dual-Difference Network	Aero-engine life prediction	Error, stability	Improves sample discrimination capability
[13]	Bayesian Probabilistic Prediction	Degradation-data life modeling	Probabilistic confidence, error	Provides uncertainty-aware outputs
[14]	Cloud-Edge Collaborative Diagnosis	Aircraft fuel pump fault diagnosis	Diagnostic accuracy, responsiveness	Reflects the cloud-edge collaborative pathway
[15]	Feature Selection + Improved TCN	Aero-engine life prediction	Prediction accuracy	Enhances temporal degradation extraction
[16]	Multi-Scale Transfer Learning	Life prediction under limited samples	Stability, generalization	Suitable for small-sample scenarios
[17]	Dual-Dimensional Attention Model	Aero-engine life prediction	Prediction accuracy, convergence	Jointly models channel and temporal information
[18]	Neural Network Prediction	Real engine operating data	Prediction error	Closer to real engineering operation records
[19]	Physics-Structure-Driven Deep Learning	Aero-engine health management	Diagnosis and compensation performance	Integrates mechanism knowledge with data information
[20]	Long Short-Term Memory Network	Engine health monitoring	Safety monitoring performance	Enhances continuous state tracking

Gan et al. [17] studied the application of two-dimensional attention model in aero-engine life prediction, and proposed an adaptive structure that simultaneously adjusted feature response from channel and time dimensions. Szrama et al. [18] studied the neural network life prediction method based on real operation data, and proposed a pattern recognition process for actual engine operating condition records, which made the model closer to the engineering monitoring environment. Xiao et al. [19] studied the application of deep learning and compensation method driven by physical structure in engine health management, and proposed a health management path combining mechanism constraints and deep representation. Yildirim et al. [20] studied the application of long short-term memory network in engine health monitoring, and proposed a calculation method to complete safety monitoring and state tracking by using continuous operation sequences.

Therefore, this paper proposes an aero-engine cloud-edge fusion health monitoring and real-time diagnosis framework supported by lightweight CNN. The local convolution expression and parameter compression mechanism of lightweight CNN can adapt to the low-delay reasoning requirements at the edge side, and the cloud training, parameter synchronization and

result correction mechanism can support cross-condition update. The combination of the two can form a computational closed loop for real-time condition recognition, abnormal alarm and online diagnosis of aero-engine, and provide a unified basis for subsequent module design and experimental analysis.

3 Research Methods

3.1 Principle and applicability analysis of lightweight CNN diagnostic model

Lightweight convolutional neural network (LCNN) is a diagnostic model for resource-constrained devices. Its core is to complete feature extraction with fewer parameters and shorter inference paths. Aero-engine health monitoring data comes from multi-source sensor channels such as vibration, temperature, pressure, speed and oil. The data stream has the characteristics of high sampling, strong coupling and time-varying. The diagnostic model not only needs to identify abnormal textures, but also needs to maintain continuous operation capability on edge nodes. Based on this task characteristic, this paper constructs a lightweight CNN diagnostic backbone on the basis of convolution decomposition, channel recali-fication, residual transfer and compression constraints, so as to adapt the model to the link in the cloud-edge fusion scenario.

(1) convolutional mapping capability. The lightweight CNN still completes state pattern extraction based on the local receptive field, and the feature generation process of the L-th layer can be expressed as follows.

$$F_l = \sigma(W_l * X_l + b_l) \tag{1}$$

Here, X_l represents the input tensor of the l layer, W_l represents the convolution kernel parameter, b_l represents the bias term, $\sigma(\cdot)$ represents the nonlinear activation function, F_l represents the output feature map. This equation indicates that the model is able to aggregate multi-channel state changes within a finite window.

(2)Calculate the compression capacity. In order to reduce the high multiplication and addition overhead caused by the standard convolution, the model uses depthwise separable convolution, and its complexity compression ratio can be expressed as follows.

$$R = \frac{k^2M + k^2MN}{k^2MN} \tag{2}$$

Here, k represents the convolution kernel size, M represents the number of input channels, N represents the number of output channels, and R represents the complexity ratio of lightweight convolution compared to standard convolution. This equation reflects a significant decrease in computation after convolution decomposition.

(3) local anomaly response ability. Aeroengine state offset is often reflected by texture mutation and frequency band shift in a short window, and the local convolution response can be expressed as follows.

$$z_{i,j,c} = \sum_{u=0}^{k-1}\sum_{v=0}^{k-1} w_{u,v,c}\ x_{i+u,j+v,c} \tag{3}$$

Here, $x_{i+u,j+v,c}$ denote the local input on channel c, $w_{u,v,c}$ denote the convolutional

weight at the corresponding position, and $z_{i,j,c}$ denote the convolutional response at the current position. This equation indicates that the lightweight CNN still retains the sensitivity to abnormal patterns.

(4) Channel recalibration capability. Considering that the contributions of different sensing channels are not consistent under different working conditions, this paper introduces channel attention weight, which can be calculated as follows.

$$a_c = \frac{\exp(q_c)}{\sum_{r=1}^{C}\exp(q_r)}, \quad q_c = \omega_c^T g(X) \tag{4}$$

Here, $g(X)$ represents the global statistical mapping, ω_c represents the mapping parameter of the c channel, q_c represents the channel score, a_c represents the normalized channel weight, and C represents the total number of channels. This formula can highlight the state channel and suppress the redundant response.

(5) Deep stable transmission ability. In order to maintain information continuity under network compression, the model retains residual connections in the backbone, which can be expressed as follows.

$$Y_t = H(X_t, \Theta_t) + X_t \tag{5}$$

where X_t represents the input features, Θ_t represents the set of unit parameters, $H(\cdot)$ represents the convolution transformation process, and Y_t represents the residual output. This formula ensures that the shallow texture and the deep semantic are cooperatively transferred in the same link.

(6) Training and deployment consistency. In order to balance classification performance, model sparsity and channel distribution stability, the joint loss function is constructed as follows.

$$L = \lambda_1 L_{ce} + \lambda_2 \|\theta\|_1 + \lambda_3 \sum_{c=1}^{C} |a_c - \bar{a}| \tag{6}$$

Here, L_{ce} represents the cross-entropy loss, θ represents the set of model parameters, $\|\theta\|_1$ represents the sparsity constraint term, a_c represents the channel weight, $\bar{a}$ represents the mean weight, and λ_1 to λ_3 represents the regulation coefficient. This formula unifies the recognition objective and the compression objective into the same optimization framework.

In general, the applicability of lightweight CNN in aero-engine health monitoring and real-time diagnosis is reflected in three aspects: local convolution and channel recallization can identify abnormal features in multi-source monitoring streams, convolution decomposition and sparse constraints reduce model volume and inference delay, residual propagation and joint optimization maintain the consistency between training phase and deployment phase. Based on these features, the lightweight CNN can be used as the edge reasoning core of the cloud-edge fusion diagnosis system, which provides the basis for subsequent real-time diagnosis module construction and model integration.

3.2 Construction of aero-engine health monitoring and real-time diagnosis module

The goal of aero-engine health monitoring and real-time diagnosis module is to convert multi-source sensor data into computable state representation and complete continuous reasoning and alarm output on edge nodes. This paper takes lightweight CNN as the diagnosis core, combines

time window organization, feature alignment, status scoring and hierarchical trigger mechanism, and constructs a modular link covering "data access - feature coding - state determination - alarm generation". The module emphasizes the executability of the edge, while retaining the compatibility with the parameter update interface of the cloud, so that the monitoring results enter the collaborative diagnosis process. Its specific structure is shown in Fig. 1.

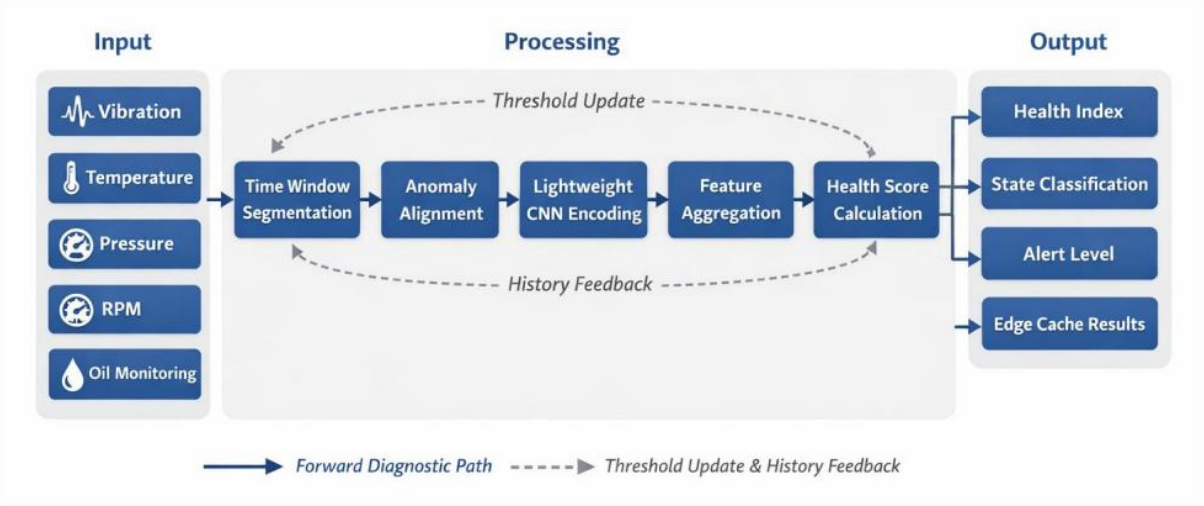

Figure 1: Structure diagram of aero-engine health monitoring and real-time diagnosis module

The module building process is as follows:

(1)Data organization. In order to ensure the consistency of inference input on the edge side, the system performs unified time window rearrangement on data with different sampling frequencies, and the samples in the window are expressed as follows.

$$S_t = \{x_{t-w+1}, x_{t-w+2}, \dots, x_t\} \tag{7}$$

Here, S_t denotes the corresponding input window at time t, w denotes the window length, and x_t denotes the current sampling vector. This formula converts the continuous monitoring stream into fixed-length segments suitable for convolutional coding, so that the edge nodes can stably process the original sequence under different working conditions.

(2) Feature fusion. Considering the different contributions of the multi-source sensing channels of the aeroengine in the state evolution, this paper constructs a weighted fusion representation after the convolution output, which is calculated as follows.

$$h_t = \sum_{m=1}^{M} \alpha_m f_t^{(m)}, \quad \alpha_m = \frac{\exp(u_m)}{\sum_{r=1}^{M} \exp(u_r)} \tag{8}$$

Here, $f_t^{(m)}$ represents the convolutional feature of the m class sensing channel, α_m represents the corresponding fusion weight, u_m represents the channel score, M represents the total number of sensing channels, and h_t represents the fused state vector. This formula ensures that the highly sensitive channel occupies a higher weight in the state determination.

(3) Health score. In order to map the high-dimensional state vector into an interpretable continuous health indicator, the health index function is defined as follows.

$$H_t = \frac{1}{1 + \exp\left(-(\beta^T h_t + b_h)\right)} \tag{9}$$

Here, H_t represents the health index at time t, β represents the mapping parameter, and b_h represents the bias term. The closer H_t is to 1, the more stable the operation state is, and the closer HT is to 0, the more obvious the degradation degree is. This formula makes the edge side output no longer limited to discrete labels, but retains the continuous state characterization ability at the same time.

(4) State determination. To map the continuous health index into executable diagnostic categories, the system sets two decision thresholds, and the status level can be expressed as follows.

$$C_t = \begin{cases} 0, & H_t \geq \tau_1 \\ 1, & \tau_2 \leq H_t < \tau_1 \\ 2, & H_t < \tau_2 \end{cases} \tag{10}$$

Here, $C_t = 0$ represents the normal state, $C_t = 1$ represents the degraded state, $C_t = 2$ represents the abnormal or faulty state, and τ_1 and τ_2 represent the grading thresholds determined by the validation set. This equation achieves a stable mapping from continuous scoring to discrete diagnosis.

(5) The alarm is triggered. In order to avoid false alarms on the edge side caused by single frame disturbance, this paper introduces a short-term consistency judgment mechanism, and its alarm confidence is defined as follows.

$$A_t = \frac{1}{L} \sum_{i=t-L+1}^{t} \mathbb{I}(C_i = 2) \tag{11}$$

Here, A_t represents the alarm confidence of the current window, L represents the look-back length, and $\mathbb{I}(\cdot)$ represents the indicator function. When A_t exceeds the set threshold, the system immediately outputs a high-priority alarm and returns the cache fragment to the cloud. This formula enhances the decision stability of the module in the continuous monitoring environment.

In summary, the constructed aero-engine health monitoring and real-time diagnosis module can convert multi-source monitoring data into state representation, and complete health scoring, state grading and alarm triggering on the edge node. The module not only maintains the computational advantages of lightweight CNN in real-time reasoning, but also provides a clear interface for the subsequent result transmission, threshold update and model correction in cloud-edge fusion, so as to establish an executable computational foundation for online monitoring and collaborative diagnosis of aero-engine.

3.3 Cloud-edge fusion mechanism design and model integration

The running state of aero-engine has the characteristics of continuous change, frequent switching of operating conditions and asynchronous arrival of monitoring flow. It is difficult to support continuous monitoring and online judgment by only relying on a single edge diagnosis result. To this end, based on the lightweight CNN diagnosis module, this paper introduces the cloud-edge fusion mechanism, and integrates the "edge end acquisition, edge end initial judgment, cloud review, result return, parameter update" into the unified operation link. The mechanism does not simply distribute the diagnosis task to two calculation positions, but dynamically distributes the calculation responsibility according to the diagnosis urgency, communication status and model confidence, so as to ensure that the aero-engine health monitoring can still maintain low delay and stable output under complex flight conditions. The

whole model consists of three core submodules, which are arranged as shown in Table 2.

Table 2: cloud-edge fusion mechanism and model integration framework

Submodule	Main Inputs	Core Methods and Processing Logic	Main Outputs
Edge Diagnosis Submodule	Vibration, temperature, pressure, rotational speed, and lubricating oil sequences	Time-window reorganization, lightweight CNN inference, preliminary state assessment, and confidence calculation	Health index, state category, and alarm flag
Cloud Collaboration Submodule	Edge preliminary results, historical sample repository, and operating context	Deep re-evaluation, cross-condition comparison, threshold calibration, and parameter aggregation	Review labels, calibrated thresholds, and fused decision results
Feedback Update Submodule	Alarm logs, maintenance records, returned samples, and operation results	Error tracking, sample write-back, rule adjustment, and incremental training triggering	Updated dataset, parameter versions, and strategy configuration

On the edge side, the system is responsible for multi-source sensing data access, time window generation, lightweight CNN inference and initial state scoring. In the cloud, the system is responsible for the review decision, historical pattern comparison, parameter aggregation and global threshold update. In the feedback layer, the system is responsible for recording alarm results, writing back state labels and triggering model retraining. In order to measure the urgency of the current diagnosis task of the edge node, this paper defines the edge task priority as follows.

$$U_t = \eta_1(1 - H_t) + \eta_2 A_t + \eta_3 D_t \tag{12}$$

Here, U_t represents the task priority At time t, H_t represents the health index obtained in the previous section, A_t represents the alarm confidence, D_t represents the monitoring flow fluctuation intensity, and η_1 to η_3 represent the regulation coefficient. This formula unifies state degradation, alarm accumulation and input disturbance as the edge-end scheduling basis.

When the edge node completes the initial judgment, the system needs to decide whether to submit the result to the cloud for deep review. To this end, this paper constructs the task offloading judgment function as follows:

$$O_t = \begin{cases} 1, & U_t + \gamma_1 L_t + \gamma_2 Q_t - \gamma_3 B_t \geq \rho \\ 0, & \text{Others} \end{cases} \tag{13}$$

where O_t represents offloading decision, 1 represents triggering cloud review, 0 represents only keeping edge output,L_t represents current load of edge nodes, Q_t represents diagnostic uncertainty, B_t represents available bandwidth, γ_1 to γ_3 represent weight parameters, and ρ represents trigger threshold. This formula reflects that cloud-edge collaboration is not a fixed division of labor, but switches in real time according to the load and diagnosis state.

In actual operation, the system generates the final diagnosis according to the edge output

and the cloud review results. In order to avoid category drift caused by unilateral judgment, this paper defines the fusion diagnosis probability as follows.

$$P_t^* = \omega_e P_t^e + \omega_c P_t^c + (1 - \omega_e - \omega_c)\bar{P}_t \tag{14}$$

where P_t^e represents the edge-side diagnosis probability, P_t^c represents the cloud review probability, $\bar{P}_t$ represents the history window prior probability, ω_e and ω_c represent the edge-cloud weights, and satisfies $0 < \omega_e + \omega_c < 1$. This formula enables the current decision to absorb both real-time reasoning information and historical running context.

According to the fusion diagnosis results, the system divides the engine states into three levels: normal, degraded and abnormal, and matches different system behaviors for different states. The corresponding arrangement is shown in Table 3.

Table 3: Aero engine cloud-edge diagnostic status classification and system behavior

State Level	Judgment Basis	Main Processing Objective	Typical System Behavior
Normal State	Low fusion probability and stable health index	Maintain continuous monitoring and low-overhead operation	Only edge inference is retained, and summary information is periodically sent back
Degraded State	Medium probability and increased fluctuation in local indicators	Track state changes in advance and collect evidence for further review	Increase the sample retention rate and trigger cloud-side review and threshold calibration
Abnormal State	High probability and rapid decline in health index	Quickly complete alarm confirmation and support maintenance decisions	Output high-priority alarms, cache key segments, and upload them to the cloud
Update Stage	Maintenance feedback or label write-back has arrived	Maintain the model's adaptability to new operating conditions	Perform incremental training, parameter replacement, and rule adjustment

In order to ensure that the diagnosis results between the edge node and the cloud server can be consistent in the monitoring process, the system introduces a state cache and timestamp alignment mechanism in the message layer. Only high-confidence abnormal segments and summary statistics were uploaded on the edge side, and only the window index and health score were retained in the normal state. The cloud was rechecked and spliced and compared according to a unified time marker. In this way, the transmission overhead is controlled, and the decision offset caused by backhaul delay between different nodes is avoided. At the same time, the version management module saves the threshold number, parameter timestamp and diagnosis summary, so that both sides of the edge cloud can still track the decision source of the same window after the model switch. This setting improves the comparability of the returned results and the stability of the deployment process.

When the feedback data accumulates to the set size, the system performs parameter update. Incremental aggregation takes the following form:

$$\Theta^{(r+1)} = \sum_{k=1}^{K} \frac{n_k}{N} \Theta_k^{(r)} - \mu \nabla L_{val} \tag{15}$$

Here, $\Theta^{(r+1)}$ represents the global parameters in round r+1, $\Theta_k^{(r)}$ represents the local parameters uploaded by the k node, n_k represents the corresponding sample size, N represents the total sample size, μ represents the correction step size, and ∇L_{val} represents the validation loss gradient. This formula makes the model update not only rely on local data, but also take into account the global verification performance.

This indicates that this paper adopts a cloud-edge fusion design method for real-time diagnosis of aeroengines. The edge diagnosis sub-module provides low-delay state recognition, the cloud collaboration sub-module provides cross-condition review and threshold correction, and the feedback update sub-module maintains the continuous adaptation of the model to new samples and new working conditions. Through this integration method, the lightweight CNN is no longer just an independent classifier, but becomes the core computing unit in the cloud-edge collaborative health monitoring link, which provides a unified implementation basis for subsequent experimental analysis.

4 Experimental Design

The data selected for the experiments in this section are composed of engine degradation sequences, bench monitoring records, and edge cloud playback samples. The open part uses the aeroengine multi-condition degradation data to establish the continuous state evolution samples. Five kinds of monitoring signals including vibration, temperature, pressure, speed and oil were collected by the bench part, which were used to construct four kinds of labels: normal, degradation, surge precursor and fault. The playback part reorganizes the edge uploaded fragments and the cloud review records according to the cloud-edge collaborative operation link, which is used to evaluate the consistency of online diagnosis. After time window segmentation, channel alignment, abnormal segment screening and label correction, 16800 groups of labeled samples were formed, in which the training set, validation set and test set were divided by 7:1:2. The input length was unified to 256 steps, and the multi-source channels were uniformly resampled to an equally spaced sequence to ensure that the input of the lightweight CNN in the edge node and the cloud review link were consistent.

The experiments in this paper are completed in a set of cloud-edge collaborative environments. The edge device used NVIDIA Jetson Orin NX with 8-core ARM processor and 16 GB memory to perform real-time inference of lightweight CNN. The cloud server uses Intel Xeon Gold 6330 processor, NVIDIA RTX 4090 graphics card, and 128 GB memory to perform model training, parameter aggregation, and result review. The operating system is Ubuntu 22.04 and the development environment is Python 3.10. The core dependencies include PyTorch 2.1, CUDA 12.1, NumPy, Pandas, and Matplotlib. The number of model training rounds is set to 120, the batch size is set to 64, the optimizer uses AdamW, the initial learning rate is set to 0.001, and the weight decay coefficient is set to 0.0001. A message queue was used between the edge end and the cloud to complete the result transmission and version synchronization, so as to ensure that the experimental process could reflect the overall diagnosis process in the cloud-edge fusion scenario.

The comparison models selected in this paper include standard CNN, MobileNetV3, LSTM diagnostic model, and centralized diagnostic model executed only in the cloud. The standard CNN is used to compare the structural differences before and after compression, MobileNetV3

is used to compare the edge adaptation ability of the lightweight convolution backbone, LSTM is used to compare the time series modeling path, and the centralized diagnosis model is used to compare the delay and consistency performance in the pure cloud mode. The evaluation metrics include accuracy, F1 score, alarm agreement rate, inference delay, model parameters, and cloud-edge collaborative response time. The key objectives of this model include 21 ms inference delay on the edge side, 3.6 MB model size, and consistency maintenance ability in the online diagnosis link, so as to provide a unified basis for subsequent diagnosis accuracy and collaborative experimental analysis.

To ensure the comparability of experimental results, the same training set partition, input window length and label definition were used for all models, and the average results were taken after repeated training for five times under the same random seed condition. The edge side inference test adopts streaming input mode, and records the single window delay and backhaul delay. The cloud review test statistical parameters delivery time and abnormal segments re-judgment time. For the alarm agreement rate, the matching ratio between the edge initial diagnosis results and the cloud diagnosis results is used as the calculation basis to evaluate the stability of the cloud-edge collaborative link under continuous monitoring conditions.

5 Analysis of experimental results

5.1 Analysis of experimental results of diagnostic accuracy and real-time performance

In this section, the proposed method is verified from two dimensions of diagnosis accuracy and real-time performance, and the performance of lightweight CNN in aero-engine cloud-edge fusion diagnosis task is analyzed by combining classification statistics, feature distribution and online running results. The experimental objects include standard CNN, MobileNetV3, LSTM diagnostic model, centralized cloud model and the proposed method. The comparison metrics include precision, F1 score, one-class recall, edge-side inference delay, model size, alert consensus and throughput performance. These results can be used to observe the comprehensive ability of the model in fine-grained state recognition, edge deployment, and collaborative diagnosis links.

Fig. 2 illustrates the confusion matrix results of the proposed method on four classes of engine states. It can be seen that the samples of the four states of normal, degradation, surge precursor and fault are mainly concentrated in the main diagonal region, indicating that the model maintains a high consistency in the overall classification. The recall rate of normal state is 97.1%, the recall rate of degradation state is 91.7%, the recall rate of surge precursor state is 90.3%, and the recall rate of fault state is 93.3%. Among them, there is still a small amount of crossing between the degradation state and the surge precursor state, but the proportion of misclassification is controlled within 6.4%. The high degree of main diagonal aggregation of fault states indicates that the model maintains a strong ability to identify significant abnormal signals. This result shows that the lightweight CNN can still form a stable distinction between the intermediate transition state and the abnormal state of the aeroengine after compressing the backbone.

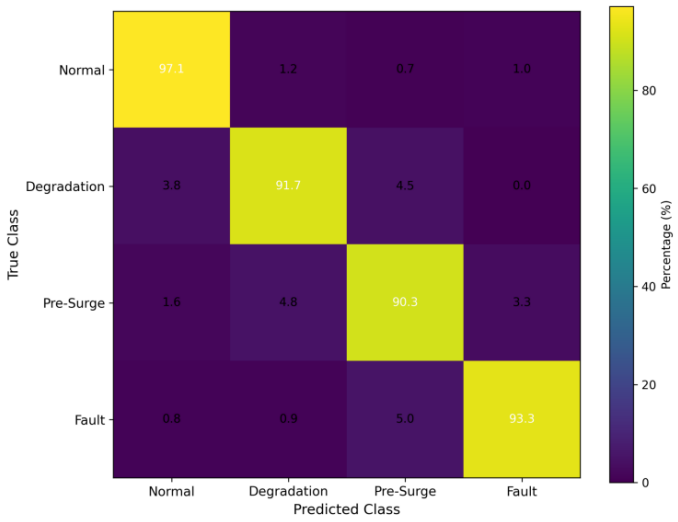

Figure 2: Heat map of the confusion matrix of the proposed method in the four-class state recognition task

To further analyze the state separation effect of different models in the high-dimensional feature space, Fig. 3 shows the two-dimensional projected distribution of the test set samples. Each scatter point in the figure corresponds to a test sample, and different colors indicate different engine states. It can be seen from Fig. 3 that the standard CNN forms a large area of overlap between the degradation state and the surge precursor state, and the LSTM diagnostic model has a long tail at the edge of the fault state, indicating that its aggregation of short-term mutation signals is not stable enough. In contrast, the proposed method forms a more compact four-cluster structure, and its intra-class average distance is about 0.38, which is lower than 0.57 of the standard CNN and 0.61 of the LSTM diagnostic model. The inter-class center distance is about 1.84, which is higher than 1.52 of MobileNetV3 and 1.73 of the centralized cloud model. The distribution shows that the proposed method can form a more stable and separated state embedding representation after lightweight convolutional coding, channel recali-fication and cloud review correction.

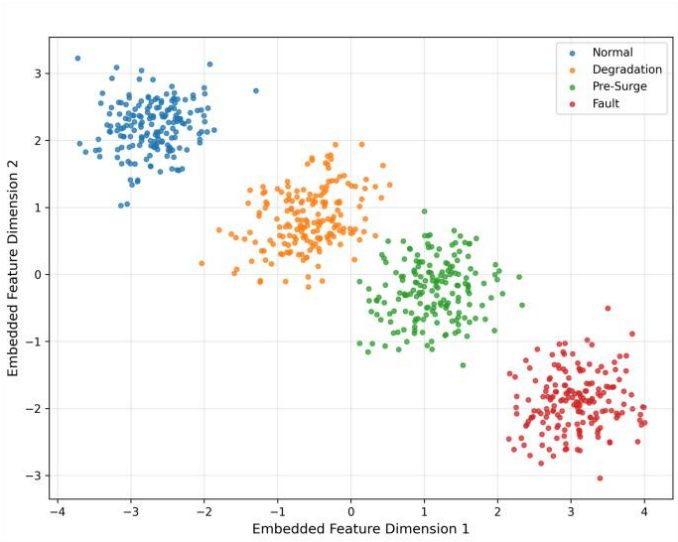

Figure 3: Projection distribution of feature space for different model test sets

In terms of overall classification performance, the accuracy of the proposed method reaches 94.7%, which is higher than 92.1% of standard CNN, 93.4% of MobileNetV3, 91.3% of LSTM diagnostic model and 94.0% of centralized cloud model. The F1 value reaches 0.931, which is also higher than 0.904, 0.917, 0.889 and 0.924 of other models. Further observation of the classification distribution in Fig. 2 shows that the normal state recognition has been relatively stable, the gap between each model is limited, and the discrimination ability between the degradation state and the surge precursor state constitutes the main source of difference. The proposed method maintains a higher main diagonal aggregation degree on these two types of states, indicating that the lightweight convolution backbone still retains the ability to characterize local abnormal textures and short-time fluctuation patterns under parameter compression. In order to avoid the accidental bias brought by a single test, this paper repeated the run five times under the same data partition, the same training configuration and the same set of random seeds, and calculated the mean, standard deviation and 95% confidence interval of the core index, and the results are shown in Table 4.

Table 4: Statistical results of stability for five repeated experiments of the proposed method

Metric	Mean ± Standard Deviation	95% Confidence Interval
Accuracy	0.947 ± 0.005	[0.942, 0.952]
F1-score	0.931 ± 0.004	[0.927, 0.935]
Precision	0.936 ± 0.004	[0.932, 0.940]
Recall	0.928 ± 0.005	[0.923, 0.933]
Edge Latency / ms	21.0 ± 1.2	[19.8, 22.2]
Alarm Consistency / %	92.4 ± 0.6	[91.8, 93.0]

Table 4 shows that the fluctuation of Accuracy, F1-score, Precision and Recall of the proposed method in five repeated experiments is small, and the standard deviation is controlled within 0.005, indicating that the lightweight CNN maintains high stability under different random initialization and batch order conditions. The fluctuation of the edge-side inference delay is 1.2 ms, the standard deviation of the alarm consensus rate is 0.6, and the variation range of the throughput is also narrow, which indicates that the cloud-side collaborative link has good repeatability and operation stability under continuous monitoring conditions.

In terms of real-time performance, Fig. 4 illustrates the edge-side inference delay distribution of different models in five repeated experiments. The delay of standard CNN mainly distributed between 32.8 ms and 35.2 ms, and the median was 34.0 ms. The latency of MobileNetV3 ranges from 18.1 ms to 19.8 ms, with a median of 19.0 ms. The delay distribution of the LSTM diagnosis model ranged from 27.9 ms to 30.1 ms, and the median was 29.0 ms. The latency of the centralized cloud model ranged from 56.4 ms to 60.1 ms, and the median was 58.0 ms. The delay distribution of the proposed method is between 19.4 ms and 22.6 ms, and the median is 21.0 ms. It can be seen that MobileNetV3 is slightly lower than the proposed method in terms of inference delay, but its classification accuracy and online consistency performance as shown in the previous section are still weaker than the proposed method. Although the centralized cloud model maintains high accuracy, the overall box is significantly higher, indicating that its remote review link brings longer reasoning waiting time. In contrast, the delay box of the proposed method is narrower, the median is stable, and the abnormal fluctuation is less, indicating that the proposed method has better edge deployment stability and real-time response ability while maintaining high diagnostic accuracy.

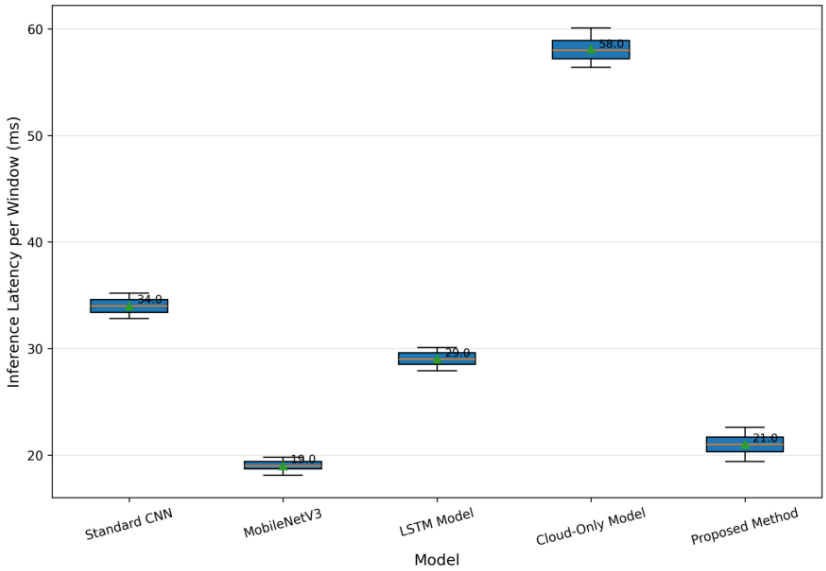

Figure 4: Box plots of distribution of inference delay on the edge side for different models

In order to avoid the accidental bias brought by a single test, this paper repeated the run five times with the same data partition, the same training configuration and the same set of random seeds, and counted the online running index. The results show that the standard deviation of accuracy is 0.005, the standard deviation of F1 value is 0.004, and the fluctuation of edge inference delay is controlled within 1.2 ms, which are lower than those of the standard CNN and LSTM diagnostic models. From the perspective of resource usage, the peak memory of the proposed method on Jetson Orin NX is 1.9 GB, and the CPU usage stabilizes between 38% and 44%, while the peak memory of the standard CNN reaches 3.1 GB under the same Settings. The resource curve is smoother, indicating that the proposed method is more suitable for continuous deployment in the long-term monitoring process, and also reduces the queuing risk and cache jitter in the cloud-side link.

In order to illustrate the operation effects of different deployment methods under continuous monitoring conditions, Table 5 statistics the response time, alarm agreement rate, throughput and frame loss rate of the three modes of edge-only, cloud-only and cloud-edge fusion in five repeated experiments. The average response time of pure edge deployment is the shortest, only 18 ms, but it is prone to alarm deviation in the stage of complex condition switching. In the cloud-only deployment, the alert consensus rate reached 93.1%, but the average response time increased to 64 ms, and the throughput decreased to 32.7 window $\cdot s^{-1}$. The average response time of the cloud-edge fusion mode is 21 ms, and the alarm consensus rate reaches 92.4%, which is 6.8 percentage points higher than that of the edge-only mode. The throughput reaches 48.6 window $\cdot s^{-1}$, and the frame loss rate is controlled at 0.7%. This indicates that the proposed method achieves a more balanced configuration among accuracy, delay and link stability.

Table 5: Statistical results of online operation for different deployment methods

Deployment Mode	Average Response Time / ms	Alarm Consistency / %	Throughput / windows·s^{-1}	Frame Loss Rate / %
Edge-Only	18	85.6	51.3	1.8
Cloud-Only	64	93.1	32.7	0.4
Cloud-Edge Fusion	21	92.4	48.6	0.7

Based on the above results, the proposed method shows more stable comprehensive advantages in both diagnosis accuracy and real-time performance. The lightweight CNN ensures fast reasoning on the edge side, and the cloud review and feedback update suppress the category drift under complex working conditions. The combination of the two forms a computing link that takes into account deployment feasibility, state determination reliability and online response efficiency. The results show that model compression, feature discrimination and cloud-edge collaboration can be coordinated under a unified framework for aero-engine health monitoring tasks.

5.2 Collaborative monitoring and online diagnosis experiment under cloud-edge fusion

In this section, after completing the verification of diagnosis accuracy and real-time performance, the actual performance of cloud-edge fusion mechanism in collaborative monitoring and online diagnosis is further investigated. The experiment focuses on two aspects: one is the collaborative efficiency of edge initial judgment, cloud review and result transmission under continuous monitoring conditions; the other is the support ability of abnormal state tracking, alarm confirmation and online update after model integration. To ensure the comparability of the experimental results, all experiments are completed under the same data partition, the same window length, and the same alarm threshold setting, and are compared with the edge-only mode, the cloud-only mode, and the variant with the removal of critical modules.

As can be seen from Fig. 5, the consistency of the pure edge mode in the normal state is relatively high, reaching 0.90, but the contour contraction is relatively obvious in the degraded state and the surge precursor state, with the corresponding values of 0.84 and 0.82 respectively, indicating that the intermediate transition state is more susceptible to local disturbance when the continuous judgment is solely dependent on edge nodes. The consistency of the pure cloud mode in the four types of states is generally high, and the normal, degradation, surge precursor and fault states are 0.92, 0.90, 0.88 and 0.94, respectively, indicating that the remote review has good stability in the confirmation of complex states. In contrast, the consistency of the proposed method is 0.94, 0.91, 0.89 and 0.93 on the four types of normal, degraded, surge precursor and fault states, respectively. The overall radar profile is more balanced, and the spread degree is significantly higher than that of the pure edge mode, and remains close to that of the pure cloud mode. The results show that the cloud-edge fusion mode not only maintains a strong confirmation ability in high confidence fault states, but also shows better continuous stability in intermediate transition regions such as degradation states and surge precursor states.

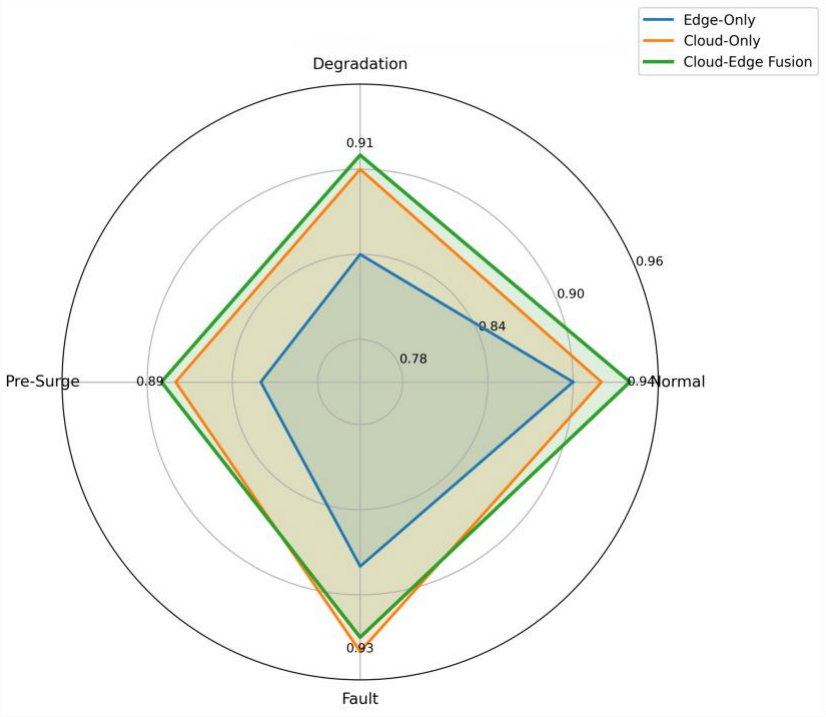

Figure 5: Radar plot of alarm consistency for different deployment patterns over four classes of states

To illustrate the overall statistical results of the collaborative monitoring link under continuous operating conditions, Table 6 shows the key indicators of different deployment patterns in five repeated experiments. It can be seen that the alarm consensus rate of the proposed method reaches 92.4%, which is 6.8 percentage points higher than that of the pure edge mode. The response time of online diagnosis remained at 21 ms, which was significantly lower than 64 ms of cloud-only mode. The proportion of abnormal clips returned was controlled at 17.9%, which was far lower than 100.0% of full uploads in pure cloud mode. This result shows that the cloud-edge fusion mechanism can maintain a high reliability of online diagnosis while controlling the communication load. Compared with the pure edge mode, although the response time of the proposed method is slightly increased, it obtains a more stable alarm confirmation effect. Compared with the pure cloud mode, the proposed method significantly shorted the link delay while maintaining near consistency, indicating that the task hierarchical allocation strategy is more reasonable in the continuous monitoring environment.

Table 6: Statistical results of collaborative monitoring operation for different deployment modes

Deployment Mode	Average Response Time / ms	Alarm Consistency / %	Abnormal Segment Upload Ratio / %	Throughput / windows·s^{-1}
Edge-Only	18	85.6	0.0	51.3
Cloud-Only	64	93.1	100.0	32.7
Cloud-Edge Fusion	21	92.4	17.9	48.6

In order to observe the state flow process in the collaborative diagnosis link more clearly, Fig. 6 shows the state transition chord diagram of abnormal samples between the three stages of "edge decision - cloud review - final output". The circular nodes in the figure correspond to

degradation states, surge precursor states, and fault states in the edge side, cloud side, and final output stages, and the chord width represents the proportion of migration between different states. It can be seen from Fig. 6 that 8.7% of the surge precursor samples are directly incorporated into the degraded state under the condition of pure edge determination, indicating that the edge side's subdivision ability to the intermediate abnormal state is still affected by local noise and short-term fluctuations. Pure cloud mode compressions this ratio to 4.1%, indicating that cloud review has a stronger convergence effect on state correction. The proposed method further compreses the proportion of surge precursor states mistakenly incorporated into degradation states to 3.2% through the combination of edge fast screening and cloud local review. At the same time, the proportion of maintaining the fault state in the final output stage reaches 93.3%, which is higher than the corresponding 88.9% of the pure edge mode, indicating that the cloud-edge fusion link has better cooperation in anomaly confirmation and result stable output. The results show that cloud-edge fusion is not to perform simple repeated reasoning on the same batch of samples, but to redistribute diagnostic duties according to state complexity and link load, so that high-frequency monitoring tasks and complex review tasks are completed on more suitable computing positions.

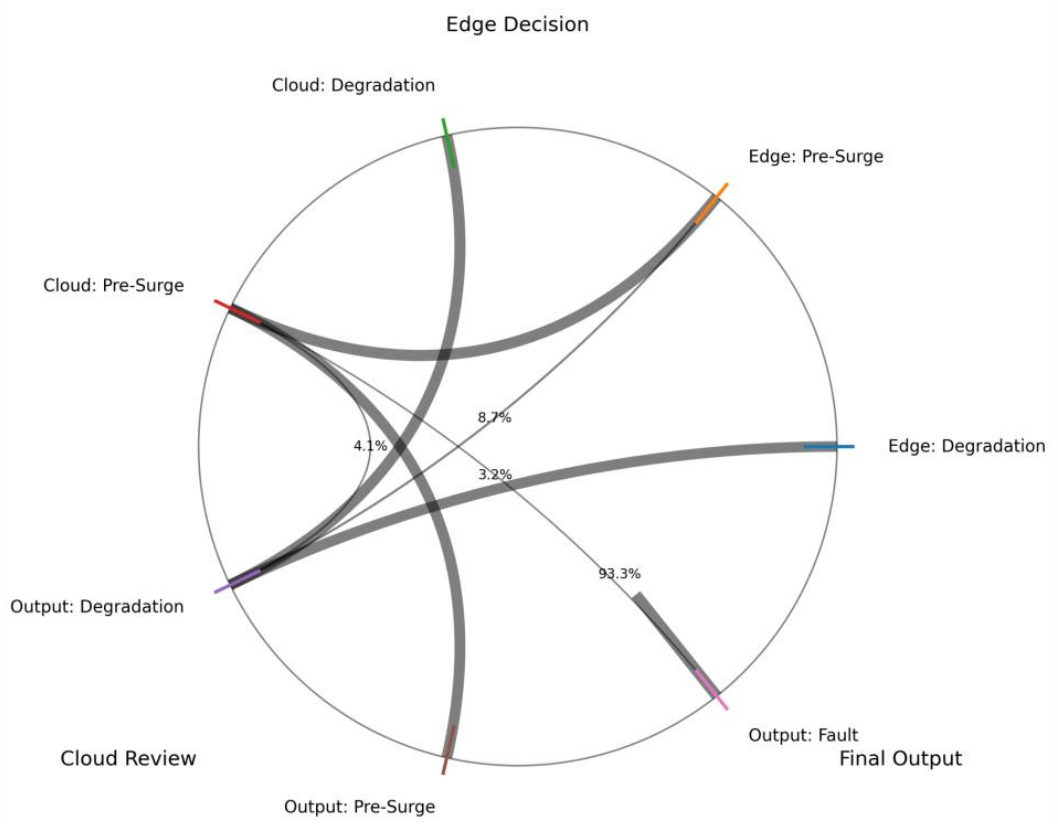

Figure 6: Chordal diagram of state transitions of anomalous samples in cloud-edge links

Furthermore, this paper conducts ablation experiments on the key components in the cloud-edge fusion framework, and the results are shown in Table 7. The full model remains optimal in four metrics: accuracy, alarm agreement, average response time, and throughput. After removing the cloud review module, the accuracy is reduced from 94.7% to 93.1%, and the alarm consensus rate is reduced from 92.4% to 88.2%, which indicates that it is difficult to stably deal with transition states and complex disturbances by judging the edge side alone. After removing the edge detection module, although the consensus rate remained at 93.0%, the average response time increased to 57 ms, and the throughput decreased to 33.4 window $\cdot s^{-1}$,

indicating that all reliance on remote analysis would weaken the real-time performance under continuous monitoring conditions. After removing the feedback update module, the classification accuracy changes little in a short time, but in the data segment with frequent switching conditions, the consensus rate drops to 89.1%, indicating that the adaptation speed of the model to the new sample distribution will be weakened after the lack of online writeback.

Table 7: Ablation experimental results of the cloud-edge fusion framework

Model Configuration	Accuracy / %	Alarm Consistency / %	Average Response Time / ms	Throughput / windows·s^{-1}
Full Model	94.7	92.4	21	48.6
Without Cloud-Side Review	93.1	88.2	19	50.4
Without Edge Preliminary Diagnosis	94.0	93.0	57	33.4
Without Feedback Update	94.2	89.1	22	46.1

In summary, it can be seen that the cloud-edge fusion collaboration mechanism constructed in this paper shows a good operation balance in continuous monitoring and online diagnosis tasks. The lightweight CNN provides low-delay edge perception, and the cloud review and feedback update provide cross-condition correction capabilities. The combination of the two makes the aero-engine state tracking, abnormal alarm and diagnosis return form a stable closed loop. Further, from the perspective of long-term running process, the average delay of the proposed method in 60 min continuous monitoring tasks is stable between 20.6 ms and 21.4 ms, and the fluctuation amplitude is controlled at the order of 1 ms. The alarm consensus rate maintained between 92.1% and 92.7% without obvious attenuation. The trigger rate of cloud review is stable at about 18%, indicating that the edge side has undertaken most of the conventional state determination tasks, while the cloud resources are mainly used for complex state confirmation and threshold correction. The overall link load remains stable and there is no obvious spike increase, which indicates that the proposed method has good continuous operation ability and engineering deployment adaptability in the online monitoring scenario of aeroengine.

6 Discussion

Based on the above experimental results, the proposed lightweight CNN cloud-edge fusion framework shows good comprehensive stability in the aero-engine health monitoring and real-time diagnosis tasks. The accuracy reaches 94.7%, the F1 value is 0.931, the inference delay of the single window on the edge side is controlled at 21 ms, and the model size is compressed to 3.6 MB, indicating that the structure compression does not weaken the discrimination ability of key states. Further combining the results in Section 5.2, we can see that the alert consensus rate of the cloud-edge fusion mode reaches 92.4%, and the proportion of abnormal segments returned is controlled at 17.9%. The throughput of 48.6 window ·s^{-1} is maintained, and the link congestion caused by the pure cloud full upload is avoided. This result shows that the advantage of the proposed method comes not only from the lightweight convolution itself, but also from the hierarchical collaboration between edge initial judgment, cloud review and feedback update. In terms of computer implementation, lightweight backbone reduces the threshold of edge deployment, cloud review improves the intermediate state confirmation ability, and feedback update alleviates the category drift caused by working condition switching. It should be pointed

out that the experiment in this paper is still mainly based on labeled samples and controlled playback links. Although the delay fluctuation range of 20.6 ms to 21.4ms is maintained in the 60 min continuous monitoring, the coverage of extreme vibration disturbances, heterogeneous sensing out-of-step and communication burst jitter is still limited.

7 Conclusions

Focusing on the task of aero-engine health monitoring and real-time diagnosis, this paper constructs a collaborative framework of lightweight CNN and cloud-edge fusion. Experimental results show that the proposed method achieves 94.7% accuracy and 0.931 F1 value in four types of state recognition, the inference delay of the edge side single window is 21 ms, and the model size is compressed to 3.6 MB. In the online diagnosis link, the alarm consensus rate reaches 92.4%, and the delay stabilizes between 20.6 ms and 21.4 ms during 60 minutes of continuous monitoring, indicating that the framework can meet the requirements of edge deployment and collaborative review while maintaining high recognition accuracy. The limitations of this paper are also relatively clear. One is that although the training samples cover normal, degraded, surge precursor, and fault states, they still have inadequate coverage of extreme environments, sensor anomalies, and cross-platform differences. Second, the current diagnosis module is mainly based on fixed-length Windows, and the description of long-distance timing dependence is still limited when facing longer duration degraded links. Third, the cloud review trigger and feedback update are mainly based on the established thresholds and rules, and there is still room for improvement in the autonomous adjustment ability in the face of continuous distribution drift. Further research can be carried out from four directions: multimodal monitoring data fusion, online incremental learning, collaborative deployment of heterogeneous edge devices, and model compression under stronger constraints, and long-term verification can be carried out combined with real flight maintenance links to enhance the engineering generalization ability and operation reliability of the framework. At the same time, the drift detection and credible reasoning mechanism can be incorporated into the update strategy, so that the monitoring results can maintain a more stable and stronger interpretation consistency under complex operating conditions.

Acknowledgment

This work was supported by the scientific research project "Low-altitude Economic Drone Technology Service Team" of Fujian Chuanzheng Communications College under project TT202412001.

References

[1] Liu L, Song X, Zhou Z. Aircraft engine remaining useful life estimation via a double attention-based data-driven architecture[J]. Reliability Engineering & System Safety, 2022, 221: 108330.

[2] Li H, Wang Z, Li Z. An enhanced CNN-LSTM remaining useful life prediction model for aircraft engine with attention mechanism[J]. PeerJ Computer Science, 2022, 8: e1084.

[3] Peng C, Chen Y, Gui W, et al. Remaining useful life prognosis of turbofan engines based on deep feature extraction and fusion[J]. Scientific Reports, 2022, 12(1): 6491.

[4] Al-Khazraji H, Nasser A R, Hasan A M, et al. Aircraft engines remaining useful life prediction based on a hybrid model of autoencoder and deep belief network[J]. IEEE Access, 2022, 10: 82156-82163.

[5] Mao P, Lin Y, Xue S, et al. Remaining useful life estimation of aircraft engines using differentiable architecture search[J]. Mathematics, 2022, 10(3): 352.

[6] Wang T, Guo D, Sun X M. Remaining useful life predictions for turbofan engine degradation based on concurrent semi-supervised model[J]. Neural Computing and Applications, 2022, 34(7): 5151-5160.

[7] Xiang S, Qin Y, Liu F, et al. Automatic multi-differential deep learning and its application to machine remaining useful life prediction[J]. Reliability Engineering & System Safety, 2022, 223: 108531.

[8] Song T, Liu C, Wu R, et al. A hierarchical scheme for remaining useful life prediction with long short-term memory networks[J]. Neurocomputing, 2022, 487: 22-33.

[9] Li Y, Chen Y, Hu Z, et al. Remaining useful life prediction of aero-engine enabled by fusing knowledge and deep learning models[J]. Reliability Engineering & System Safety, 2023, 229: 108869.

[10] Chen J, Li D, Huang R, et al. Aero-engine remaining useful life prediction method with self-adaptive multimodal data fusion and cluster-ensemble transfer regression[J]. Reliability Engineering & System Safety, 2023, 234: 109151.

[11] Xu Z, Zhang Y, Miao J, et al. Global attention mechanism based deep learning for remaining useful life prediction of aero-engine[J]. Measurement, 2023, 217: 113098.

[12] Liu N, Zhang X, Guo J, et al. Aero-engine remaining useful life prediction based on bi-discrepancy network[J]. Sensors, 2023, 23(23): 9494.

[13] Hu Y, Bai Y, Fu E, et al. A novel remaining useful life probability prediction approach for aero-engine with improved Bayesian uncertainty estimation based on degradation data[J]. Applied Sciences, 2023, 13(16): 9194.

[14] Miao Y, Li Y, Pan J, et al. Bio-inspired fault diagnosis for aircraft fuel pumps using a cloud-edge system[J]. Biomimetics, 2023, 8(8): 601.

[15] Zha W, Ye Y. An aero-engine remaining useful life prediction model based on feature selection and the improved TCN[J]. Franklin Open, 2024, 6: 100083.

[16] Liu Q, Zhang Z, Guo P, et al. Enhancing aircraft engine remaining useful life prediction via multiscale deep transfer learning with limited data[J]. Journal of Computational Design and Engineering, 2024, 11(1): 343-355.

[17] Gan F, Shao H, Xia B. An adaptive model with dual-dimensional attention for remaining useful life prediction of aero-engine[J]. Knowledge-Based Systems, 2024, 293: 111738.

[18] Szrama S, Lodygowski T. Aircraft Engine Remaining Useful Life Prediction using neural networks and real-life engine operational data[J]. Advances in Engineering Software,

2024, 192: 103645.

[19] Xiao D, Xiao H, Li R, et al. Application of physical-structure-driven deep learning and compensation methods in aircraft engine health management[J]. Engineering Applications of Artificial Intelligence, 2024, 136: 109024.

[20] Yildirim S, Rana Z A. Enhancing aircraft safety through advanced engine health monitoring with long short-term memory[J]. Sensors, 2024, 24(2).