Vol. 08, No. 3 (2026) 1751-1768, doi: 10.24874/PES08.03.027

Proceedings on Engineering Sciences

www.pesjournal.net

AN AI-ENABLED HIERARCHICAL LSTM AUTOENCODER FRAMEWORK FOR PREDICTIVE MAINTENANCE IN INTELLIGENT TRANSPORTATION SYSTEMS USING MULTIMODAL WIRELESS SENSOR DATA

L. Arokia Jesu Prabhu[1]
R. T. Thivya Lakshmi
Mokana Muthu Kumarasamy
P. V. V. S. Srinivas
Selvamani Indrajith
S. Navaneethan
Aseel Smerat
K. Prakash

Received 01.06.2026.
Revised 16.07.2026.
Accepted 11.08.2026.

Keywords:

Smart transportation systems, Long short-term memory, Autoencoder, Anomaly detection, Sensor networks, Artificial Iintelligence, Machine learning, Internet of Things

A B S T R A C T

For smart transportation systems (STS), predictive maintenance is essential to improve reliability and efficiency, but traditional models struggle to detect complex, time-dependent fault patterns. This work proposes a Hierarchical-LSTM-autoencoder (H-LSTM-AE) based predictive maintenance model (PMM) within a 4-layer hierarchical model to process multi-modal sensor data and enable real-time proactive maintenance. The proposed H-LSTM-AE model includes layers for data measurement, transmission, analysis, and prognosis. The H-LSTM-AE is designed to capture temporal dependencies in transportation data and perform anomaly detection (AD) across different operational conditions. The proposed H-LSTM-AE achieved a detection accuracy of 96.82%, a false positive rate of 2.34%, and an average response time of 32.5 seconds under steady-state operating conditions. The model achieved a detection delay of 37.4 seconds for the whole evaluation. Compared with baseline models such as Isolation Forest (iForest), autoregressive integrated moving average (ARIMA), convolutional neural network (CNN) + LSTM, and variational autoencoder (VAE), the proposed H-LSTM-AE outperformed them in precision (94.37%), recall (92.83%), and F1 (93.59%). The prognosis layer also allows for prioritized maintenance scheduling based on anomaly severity and operational criticality. These results validate the proposed H-LSTM-AE for reliable, proactive maintenance in intelligent transportation systems. This study employs a prediction layer to schedule maintenance tasks by anomaly type and functional importance. The above results validate the H-LSTM-AE for reliable, proactive maintenance in STS.

© 2026 Published by Faculty of Engineering

[1] Corresponding author: L. Arokia Jesu Prabhu
Email: arokiajeruprabhu@gmail.com

1. INTRODUCTION

Predictive maintenance is essential to ensure operational reliability, safety, and efficiency in smart transportation systems (STS), which are rapidly evolving (Ezeigweneme et al., 2024; Zhou et al., 2024). Whether reactive or preventive, traditional maintenance methods often fail to predict failures, resulting in costly downtime and poor service quality (Patel et al., 2024; Yazdi, 2024; Kashpruk et al., 2023). Predictive maintenance uses data-driven methods to anticipate problems before they become severe, thereby reducing maintenance costs. Artificial intelligence (AI) and the Internet of Things (IoT) have been combined for predictive maintenance in transportation sectors, where vehicle systems generate large volumes of time-series data (Figueroa et al., 2024). Using wireless sensor networks (WSN) and real-time data analytics, predictive maintenance models (PMM) continuously monitor vehicle health and perform anomaly detection (AD), and provide maintenance recommendations.

In transportation PMM, current methods rely on rule-based models or traditional statistical methods. Researchers have investigated statistical forecasting and machine-learning approaches, including autoregressive integrated moving average (ARIMA) and Isolation Forest (iForest), for time-series forecasting and anomaly detection; however, their applicability depends on the monitored system's features and sensor data. However, these models have limitations in representing complex and non-linear patterns in STS. The limitation of iForest is that it struggles with temporal dependencies in time-series data, leading to a higher false positive rate (FPR) and false negative rate (FNR). ARIMA is restricted to linear relationships and thus has limited ability to capture complex anomaly patterns in multi-modal sensor data from transportation subsystems (Algburi et al., 2025).

Machine learning (ML), such as convolutional neural networks (CNNs) and variational autoencoders (VAEs), has been used to learn these patterns (Wu et al., 2023). These methods have improved AD accuracy, but they are computationally intensive and struggle with real-time data because of long training times and high resource requirements (Yin et al., 2023). The hybrid CNN + LSTM is effective for processing temporal data but requires substantial computational resources, limiting its scalability in real-time applications (Algburi et al., 2024a). Predictive transportation maintenance requires an efficient, scalable model that can process complex time-series data with low FPR and fast AD times (Algburi et al., 2024b).

This study proposes a hierarchical LSTM-autoencoder (H-LSTM-AE) model for predictive maintenance in STSs. The hierarchical model has data measurement, transmission, analysis, and prognosis layers. For processing multimodal time-series data, LSTM + AE captures short- and long-term dependencies for AD. A WSN collects data on key STS metrics, including vibration, temperature, and pressure, then pre-processes it and transmits it via secure, low-latency channels to a cloud setup for real-time analytics (Li et al., 2023). The proposed H-LSTM-AE works as the analytical core, and AD is performed based on reconstruction errors. The model compares reconstructed input sequences with actual observations and triggers AD when reconstruction loss exceeds a dynamically set threshold. The hierarchical design provides data flow and processing integrity for real-time AD across several operating conditions and fault severities (Berahmand et al., 2024).

Deep learning (DL) methods, including convolutional neural networks (CNNs) and variational autoencoders (VAEs), have been applied to fault-diagnosis problems; in particular, VAE-augmented CNNs have been investigated for rolling-bearing fault diagnosis (Wang et al., 2024). Nazir et al. (2024) employed a hybrid CNN-LSTM for efficient threat detection in the IoT ecosystem. The primary contributions of this proposed H-LSTM-AE are as follows:

a) **Hierarchical Model for PMM:** This research proposes a 4-layer model for processing sensor data from collection to maintenance decision-making. The four layers include data measurement, transmission, analysis, and prognosis.
b) **LSTM + AE for Enhanced AD:** The proposed H-LSTM-AE captures complex temporal patterns in STS. The model is evaluated against traditional and other DL methods based on AD accuracy, FPR, and response time (RT).
c) **Real-Time, Multi-Stakeholder Access:** The proposed H-LSTM-AE provides role-specific access and visualization for operators, maintainers, and decision-makers.
d) **Complete Experimental Validation:** In this study, we validated the proposed H-LSTM-AE model using a 14-month dataset collected from a city bus fleet. We compared the model with benchmark models, including iForest, ARIMA, CNN-LSTM, and VAE. In the experiments, we fused different fault-severity thresholds and operating settings into the attention mechanism.

The research paper is organized as follows. In the following introductory section, we describe the proposed model and the unique functions of each layer. Section 3 provides the experimental setup, including a brief overview of the dataset, deployment context, configuration, and output parameters for the presented H-LSTM-AE. Section 4 expands the findings and presents a comparative analysis with other models. Section 5 further explains the results and concludes with directions for future studies.

2. PROPOSED H-LSTM-AE MODEL

The H-LSTM-AE model is a 4-layer hierarchy-based model that incorporates the Internet of Things-WSN, data

processing, and AI analytics responsibilities for PMM in STS (Figure 1). The data measurement, transmission, analysis, and prognosis layers make up the complete model. The data measurement layer includes multiple sensor types that collect data such as acoustic signatures, temperature signatures, pressure signatures, and vibration signatures across STS (Wang et al., 2024).

After edge processing, the data is transmitted to the transmission layer, where it is encrypted using encryption mechanisms and the secure message queue telemetry transport (MQTT) protocol. From a second perspective, the prognosis layer is primarily accountable for transforming the results obtained into maintenance recommendations. The analysis layer uses an LSTM-AE to detect patterns and anomalies in the data (Nazir et al., 2024).

To track computational load, the model employs a cloud-based network that impacts dynamic resource allocation. Through role-specific access and visualization-based interfaces, the system supports operators, maintainers, transport governmental agencies, and original equipment manufacturers (OEMs). Security mechanisms control access, ensure data privacy, and transmit data relevant to each user class.

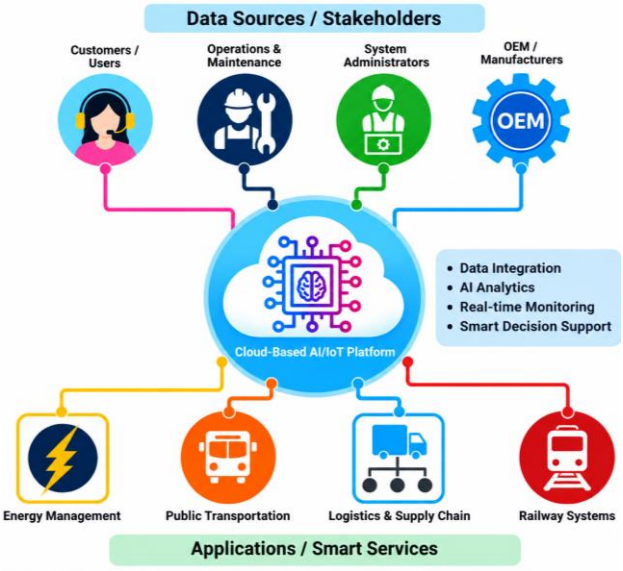


Figure 1. Proposed H-LSTM-AE Model

2.1 Data Measurement Layer (DML)

The DML provides an interface between the physical transportation setup and the digital monitoring system and serves as the preliminary layer of the proposed H-LSTM-AE model. This layer uses a wireless sensor network (WSN) to collect multimodal data streams from a wide range of transportation subsystems. Mobile equipment included in the deployment includes vibration analyzers, thermal imaging devices, acoustic sensors, and precision accelerometers (Mahmoud & Mohammed, 2024). The deployment process also includes non-mobile components. Distributed WSNs have been employed to keep track of essential elements and identify possible failure points of the STS. The sensor nodes include vibration sensors operating at frequencies from 10 Hz to 1 kHz, temperature sensors with an accuracy of ±0.5 °C to detect thermal variations, and pressure sensors with a measuring range of 0 to 100 PSI for hydraulic and pneumatic systems. These sensors function to detect mechanical anomalies. Acoustic sensors with a frequency response up to 20 kHz can detect unusual operational sounds, and current sensors can monitor electrical systems with around 0.1% accuracy.

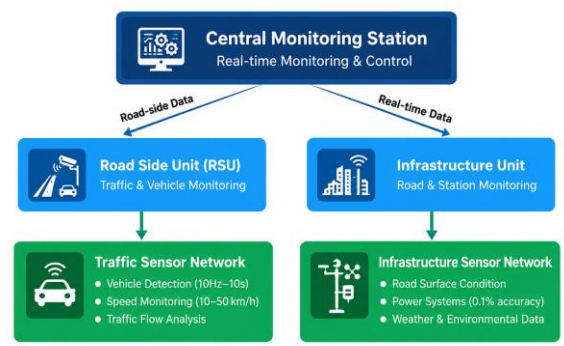


Figure 2. WSN topology

This layer implements a mechanism that enables intelligent data sampling. This mechanism adjusts the collection parameters based on operational conditions and system states. The sensors maintain a standard sampling frequency during normal operation, which increases when AD behavior is detected. Real-time sensor calibration verification, signal conditioning, automated health monitoring, and cross-validation between redundant sensors are a few examples of methods used to ensure data quality. A hierarchical WSN topology is used at the network layer (Figure 2), divided into primary hubs, secondary nodes, and auxiliary sensors to collect multi-modal data. Edge computing capabilities at sensor nodes (SNs) primarily handle preliminary data processing tasks. These operations include noise filtering, sensor AD, data compression, and statistical analysis. Error-handling and fault-tolerance mechanisms include methods for detecting sensor failure, validating sensor data, and establishing protocols to activate backup sensors if primary sensors fail. The system stores metadata for each sensor, including calibration history, maintenance schedules, operating parameters, and performance metrics.

2.2 Data Transmission Layer (DTL)

The DTL uses a communication model for data flow from the distributed WSN to the cloud setup. This layer implements the MQTT protocol (Figure 3) for IoT device communication. The transmission model has a three-tier topology comprising edge nodes for local data aggregation, gateway devices for protocol translation, and cloud endpoints for centralized data collection. The system uses MQTT quality of service (QoS) levels 0-2

for transmission reliability and applies transport layer security/secure sockets layer (TLS/SSL) encryption with 256-bit advanced encryption standard (AES) for data security. For message integrity, the system uses X.509 certificate-based authentication, and configurable compression algorithms optimize bandwidth usage. The system also includes automatic message retry and recovery mechanisms to ensure delivery under unstable network conditions.

At the transmission layer, edge computing capabilities include local data buffering during connectivity interruptions, priority-based message queuing, and real-time data validation (Mohammed et al., 2023). This layer also implements preliminary data-compression algorithms and protocol conversion mechanisms to integrate legacy systems. The security model includes end-to-end data encryption, role-based access control (RBAC), and secure gateway authentication protocols. For data integrity verification, SHA-256 hashing is used, and intrusion detection and prevention systems (IDPS) are implemented to secure against unauthorised access. Network resilience is achieved through redundant communication paths, failover mechanisms, adaptive routing protocols, and monitoring of transmission metadata.

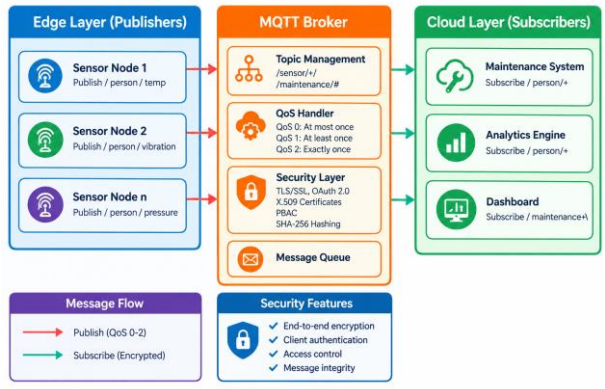


Figure 3. MQTT

2.3 Physical Layer (PL)

The PL component (Figure 4) processes raw data for AD and maintains prediction requirements. The proposed H-LSTM-AE performs AD by capturing patterns in time-series data and identifying deviations from normal operational behaviours. LSTM's sequential memory capabilities handle time-dependent data. The method includes three primary phases: (i) LSTM-Encoder, (ii) LSTM-Decoder, and (iii) AD.

i. **LSTM + Encoder:** The LSTM + Encoder component compresses the input sequence into a lower-dimensional feature representation, capturing the most relevant time-series features. The encoder consists of multiple stacked LSTM layers, where each LSTM cell processes the sequence data $'X'$ one time step at a time. The HSs generated by each LSTM cell encode temporal dependencies within the sequence.

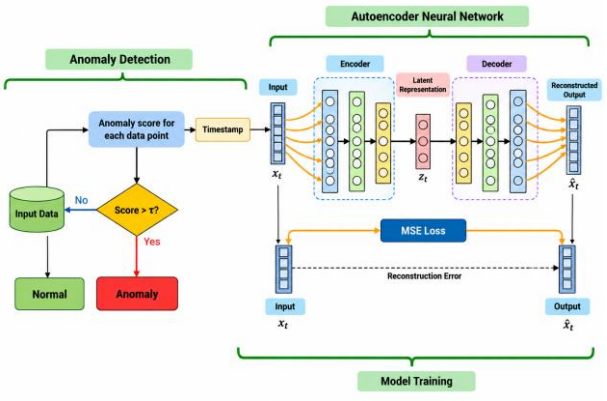


Figure 4. The proposed H-LSTM-AE for AD

Each LSTM layer in the encoder processes the input sequence $'X'$ one step at a time, maintaining a hidden state (HS) that evolves over the sequence, and allows the model to capture short- and long-term dependencies. At each time step $'i'$, the LSTM cell in the encoder takes the current input X_i, the HS as h_{i-1}, and the CS as c_{i-1} from the previous time step and outputs an updated HS as h_i and CS as c_i based on the following gate mechanisms:

a) Forget Gate (FG): Controls the data from the previous CS as c_{i-1} that should be retained. The FG as f_i is defined as Equation (1)

$$f_i = \sigma\left(W_f X_i + U_f h_{i-1} + b_f\right) \quad (1)$$

where,
W_f, U_f, b_f are the weights and biases for the FG,
$'\sigma'$ is the sigmoid activation function.

b) Input Gate (IG): Determines the extent to which the current input X_i should impact the CS. The IG as i_i and the candidate CS, as $\tilde{c}_i$ are computed as in Equation (2) and Equation (3)

$$i_i = \sigma(W_i X_i + U_i h_{i-1} + b_i) \quad (2)$$

$$\tilde{c}_i = \tanh(W_c X_i + U_c h_{i-1} + b_c) \quad (3)$$

Where,
W_i, U_i, b_i are the weights and biases for the IG
W_c, U_c, b_c are the candidate CS's weights and biases.

c) Cell State (CS): The CS as c_i is updated by combining the previous CS as c_{i-1} (modulated by the FG) and the candidate CS, as $\tilde{c}_i$ (scaled by the IG), Equation (4)

$$c_i = f_i \odot c_{i-1} + i_i \odot \tilde{c}_i \quad (4)$$

Where,
$\odot$ is element-wise multiplication.

d) Output Gate (OG): Controls the output of the LSTM cell at the current time step, determining the HS as h_i. The OG as o_i and the hidden state h_i are given by Equation (5) and Equation (6), respectively

$$o_i = \sigma(W_o X_i + U_o h_{i-1} + b_o) \quad (5)$$

$$h_i = o_i \odot \tanh(c_i) \quad (6)$$

where,
W_o, U_{or}, b_o are the weights and biases of the original model.

The encoder consists of $'L'$ stacked LSTM layers, each processing the sequence X from time step $i = 1$ to $i = T$. The HS from the final LSTM layer at the end of the sequence h_T is the encoded information for the entire input sequence, compressing it into a fixed-dimensional latent vector $'Z'$. This latent vector $Z \in \mathbb{R}^k$ (where $'k'$ is the dimensionality of the compressed representation) is the final output of the encoder and serves as a summary of the temporal patterns in the input sequence $'X'$. The encoding transformation can be formally expressed as Equation (7).

$$Z = \text{Encoder} - \text{LSTM}(X) = h_T \quad (7)$$

where,
Encoder-LSTM is the sequential application of LSTM cells across all layers in the encoder.

ii. **LSTM + Decoder:** The LSTM + Decoder consists of *L* stacked LSTM layers, which mirror the encoder model. The decoder takes the latent vector $'Z'$ and uses it to generate a sequence of hidden states that aim to replicate the original time-series data.

The process can be broken down as follows:

a) Repeat Vector: Since Z is a fixed-length vector representing the entire sequence, it must be repeated across each time step to serve as input for the LSTM cells in the decoder. This is achieved through a Repeat Vector operation, denoted as $\text{RV}(Z)$. The operation repeats ZT times, where T is the length of the original sequence, producing a sequence $\{\text{RV}_Z^1, \text{RV}_Z^2, \dots, \text{RV}_Z^T\}$ with each element equal to Z.

b) Decoding with LSTM Layers: The repeated latent vector sequence is fed into the stacked LSTM layers of the decoder, which generate an output sequence $\hat{X} = \{\hat{X}_1, \hat{X}_2, \dots, \hat{X}_T\}$ that approximates the original input sequence $'X'$.

At each time step i, the decoder's LSTM cell computes the HS as $h_i^{(d)}$ and CS as $c_i^{(d)}$ as follows:

- The LSTM cell at time step $'i'$ takes RV_Z^i as input, along with the previous HS as $h_{i-1}^{(d)}$ and the cell state $c_{i-1}^{(d)}$ to produce the current HS as $h_i^{(d)}$ and CS as $c_i^{(d)}$.
- Each layer in the decoder receives the previous layer's output, so the final LSTM layer in the stack produces the decoded output for each time step.

Mathematically, this decoding process at each time step i is represented as Equation (8)

$$h_i^{(d)}, c_i^{(d)} = \text{Decoder} - \text{LSTM}\left(\text{RV}_Z^i, h_{i-1}^{(d)}, c_{i-1}^{(d)}\right) \quad (8)$$

where LSTM + Decoder is applied to the stacked LSTM layers within the decoder, and $h_i^{(d)}$ and $c_i^{(d)}$ are the HS and CS states at time step $'i'$ in the decoder.

c) Output Generation: At each time step $'i'$, the HS is $h_i^{(d)}$ from the final LSTM layer in the decoder is passed through a fully connected (FC) layer with a linear activation function to generate the reconstructed time step $\hat{X}_i$.

This can be expressed as Equation (9)

$$\hat{X}_i = W_o h_i^{(d)} + b_o \quad (9)$$

where, W_o, b_o are the weight matrix and bias term for the output layer. The output $\hat{X}_i$ is the model's reconstruction of the input at time step i, and the complete sequence $\hat{X} = \{\hat{X}_1, \hat{X}_2, \dots, \hat{X}_T\}$ is the decoder's attempt to replicate the original input sequence $'X'$.

iii. **AD:** The AD stage in LSTM + AE uses reconstruction loss to find deviations from normal behavior. The loss is calculated between the input and reconstructed sequences. This is the final component of the model and determines if the input sequence is anomalous. When reconstruction deviates significantly from the input sequence, it indicates abnormal system behavior.

a) **Reconstruction Loss Calculation:** The primary basis for AD in the proposed H-LSTM-AE is the reconstruction loss $'L'$, which quantifies the difference between the input sequence $X = \{X_1, X_2, \dots, X_T\}$ and the reconstructed sequence $\hat{X} = \{\hat{X}_1, \hat{X}_2, \dots, \hat{X}_T\}$ produced by the decoder.

The reconstruction loss for each time step *i* can be defined using the Mean Squared Error (MSE) between the input X_i and its reconstruction $\hat{X}_i$, Equation (10)

$$L_i = \|X_i - \hat{X}_i\|^2 \quad (10)$$

where, $\|\cdot\|^2$ is the squared L_2-norm.

This metric captures the difference between the actual and reconstructed data for each time step. The total reconstruction loss for the entire sequence is then the average of these per-time-step losses, Equation (11).

$$L = \frac{1}{T}\sum_{i=1}^{T} L_i = \frac{1}{T}\sum_{i=1}^{T} \|X_i - \hat{X}_i\|^2 \quad (11)$$

where $'T'$ is the total length of the sequence.

b) **Anomaly Score Calculation:** The reconstruction loss $'L'$ serves as the anomaly score for the input sequence. AD assumes that the proposed H-LSTM-AE, trained on standard operating data, can reconstruct sequences with minimal error if they

reflect typical system behavior. If the input sequence represents an abnormal condition, the model is likely to fail to reconstruct it, leading to a higher reconstruction loss. Thus, a high anomaly score L specifies that the input sequence may be anomalous.

c) **Setting the Anomaly Threshold:** To determine whether the AD score $'L'$ specifies an anomaly, a threshold 'η' is set based on the distribution of reconstruction losses in the training data (assumed to be standard data). In this work, a statistical method is used to determine this threshold; this is done by computing the mean μ and standard deviation (SD) $'\sigma'$ of reconstruction losses from average training data and setting the threshold at $\eta = \mu + \alpha\sigma$, where α is a selected constant (*e.g.*, $\alpha = 3$) to control the sensitivity of AD. Formally, this work classifies a sequence as anomalous if: $L > \eta$

d) **Anomaly Classification:** With the threshold η in place, each new input sequence can be evaluated as follows:

- **Normal:** If the anomaly score $L \leq \eta$, the input sequence is considered normal, and no anomaly is flagged.
- **Anomalous:** If the anomaly score $L > \eta$, the input sequence is classified as an anomaly, which is a potential fault or abnormal condition in the system.

The classification rule can be expressed as Equation (12)

$$\text{Anomaly} = \begin{cases} \text{Yes} & \text{if } L > \eta \\ \text{No} & \text{if } L \leq \eta \end{cases} \qquad (12)$$

e) **Interpreting Anomalies and Triggering Alerts:** When AD exceeds the threshold L > η, an alert is generated. In the PMM context, this alert triggers maintenance checks or preventive measures. A higher anomaly score indicates a larger deviation from regular operation.

f) **Continuous Monitoring and Feedback:** In practical applications, AD operates continuously and monitors real-time sequences from STS. As new data is collected, it is fed into the Encoder-Decoder, and the reconstruction loss is computed. AD results can be logged, and over time, feedback from maintenance actions can be used to adjust the threshold 'η' or to refine the model to improve sensitivity and specificity in detecting true faults.

Algorithm 1 for the proposed H-LSTM-AE for AD in PMM

Input: Time-series data $X \in \mathbb{R}^{T \times d}$, where T is the sequence length, and d is the dimensionality of each time step (*e.g.*, sensor readings).

Output: Anomaly Label (Normal or Anomalous) for each input sequence.

Step 1: Data Pre-processing

1 Normalize the Data:

- Compute the mean $'\mu'$ and SD as $'\sigma'$ for each sensor feature over the training dataset.
- Normalize each time-series data point X_i in the sequence, $X_i = \frac{X_i - \mu}{\sigma}$
- **Segmentation (Optional):** Segment the time-series data into overlapping or non-overlapping sequences of fixed length $'T'$, resulting in multiple sequences $X_j = \{X_{j,1}, X_{j,2}, \ldots, X_{j,T}\}$, where each X_j will be used as an individual input for the LSTM-AE.

Step 2: Model Training: LSTM-AE

1 Initialize Model Parameters:

- Define the encoder and decoder models with the following hyperparameters:
- The number of LSTM layers $'L'$.
- The dimensionality of each LSTM layer.
- The dimensionality k of the latent vector Z (output from the encoder).

2 Encoder:

- **For each** sequence X_j, process each time step $X_{j,i}$ using the LSTM cells in the encoder :

$$h_i, c_i = \text{EncoderLSTM}(X_{j,i}, h_{i-1}, c_{i-1})$$

- At the final time step T, extract the latent vector $Z_j = h_T$, which encodes the entire sequence.

3 Decoder:

- Repeat the latent vector Z_j across T time steps using a Repeat Vector operation, producing $\text{RV}(Z_j) = \{\text{RV}_{Z_j}^1, \ldots, \text{RV}_{Z_j}^T\}$.
- For each time step i in the decoder, use the LSTM cells to generate the reconstructed sequence: $h_i^{(d)}, c_i^{(d)} = \text{Decoder} - \text{LSTM}\left(\text{RV}_{Z_j}^i, h_{i-1}^{(d)}, c_{i-1}^{(d)}\right)$

$$\hat{X}_{j,i} = W_o h_i^{(d)} + b_o$$

- Collect the reconstructed sequence $\hat{X}_j = \{\hat{X}_{j,1}, \ldots, \hat{X}_{j,T}\}$.

4. Calculate Reconstruction Loss: Compute the MSE between each original sequence X_j and its reconstruction $\hat{X}_j$. $L_j = \frac{1}{T}\sum_{i=1}^{T} \|X_{j,i} - \hat{X}_{j,i}\|^2$

5. Threshold Setting for AD

- Calculate the mean μ_L and SD as σ_L of reconstruction losses across the training data (assumed to be normal).
- Set the threshold $\eta = \mu_L + \alpha\sigma_L$, where $'\alpha'$ is a selected constant (*e.g.*, $\alpha = 3$).

Step 3: Real-Time AD

- **Receive New Input Sequence:** Collect a new sequence $X_{\text{new}} = \{X_{\text{new, 1}}, X_{\text{new, 2}}, \dots, X_{\text{new},T}\}$ from the time-series data.
- **Normalization:** Normalize X_{new} using the mean $'\mu'$ and standard deviation $'\sigma'$ from the training data.
- **Encoding with LSTM Encoder:** Process each time step in X_{new} by the encoder to obtain the latent vector Z_{new} at the final time step $'T'$.
- **Decoding with LSTM Decoder:** Repeat the latent vector Z_{new} across $'T'$ time steps using the Repeat Vector operation. Generate the reconstructed sequence $\hat{X}_{\text{new}} = \{\hat{X}_{\text{new},1}, \dots, \hat{X}_{\text{new},T}\}$ using the decoder.
- **Compute Reconstruction Loss for New Sequence:** Compute the reconstruction loss L_{new} for X_{new} as: $L_{\text{new}} = \frac{1}{T}\sum_{i=1}^{T} \|X_{\text{new},i} - \hat{X}_{\text{new},i}\|^2$
- **AD Decision:** Compare L_{new} to the threshold η :
  - If $L_{\text{new}} \leq \eta$, classify X_{new} as Normal.
  - If $L_{\text{new}} > \eta$, classify X_{new} as Anomalous.
- **Alert Generation (If Anomalous):** If X_{new} is classified as anomalous, trigger an alert for potential maintenance, and log the anomaly score L_{new} for further analysis.

2.4 Data Analysis Layer (DAL)

The Prognosis layer employs a two-stage method: risk assessment and maintenance-scheduling optimization.

i) Risk Assessment Model: The risk assessment model quantifies model health by a composite risk score R_s, computed as Equation (13)

$$R_s = w_1 A_s + w_2 T_c + w_3 C_f \quad (13)$$

where, A_s is the AD score derived from the proposed H-LSTM-AE reconstruction loss, T_c is the time-criticality factor based on the component's remaining lifecycle and the urgency of intervention, and C_f is the component's criticality factor, reflecting its importance within the system.

ii) Maintenance Scheduling Optimisation: The maintenance scheduling function $S(t)$ for maintenance tasks is formulated as follows **in** Equation (14)

$$S(t) = \text{argmax} \sum_{i=1}^{n} R_i(t) x_i(t) \quad (14)$$

subject to resource constraints, Equation (15)

$$\sum_{i=1}^{n} r_i x_i(t) \leq R_{\max} \quad (15)$$

where, $R_i(t)$ is the risk score for component $'i'$ at time t, $x_i(t)$ is a binary decision variable indicating whether maintenance is scheduled for component $'i'$ at time t (1 for scheduled, 0 for not scheduled), r_i is the resource requirement for maintaining component $'i'$ and $R_{\max}$ is the maximum available resources (*e.g.*, human resources, cost) at an assumed time.

iii) **Dynamic Threshold Adjustment Mechanism:** The dynamic threshold adjustment function $\delta(t)$ is defined as, Equation (16)

$$\delta(t) = \delta(t-1) + \alpha(M_a - M_p) \quad (16)$$

where, M_a is the actual maintenance findings experienced after an intervention, M_p is the LSTM + AE-recommended predicted maintenance requirements, and α is the learning rate that controls the rate of threshold adjustment.

The proposed PMM introduces several methodological improvements and engineering implementations over the proposed H-LSTM-AE-based IDS typically employed in intelligent transportation and industrial monitoring systems. Existing PMMs mostly employ an IDS based on reconstruction error without considering a comprehensive end-to-end maintenance decision system.

Most conventional methods focus solely on the accuracy of temporal sequence reconstruction and provide little support for hierarchical sensor integration, operational scalability, adaptive maintenance scheduling, and real-time industrial deployment. The proposed H-LSTM-AE model, on the other hand, addresses this drawback by implementing a 4-layer hierarchical model: data measurement, data transmission, data analysis, and prognosis layers, which together form an ecosystem for PMM in STS. The proposed H-LSTM-AE model combines an IoT-supported WSN, secure MQTT-based communication, adaptive anomaly thresholding, and prognosis-based maintenance scheduling within a single operational model, unlike the conventional LSTM-AE, which relies solely on the analytical level. This integration enables resource-aware AD and maintenance planning, prioritizing actions intelligently even under variable transportation conditions.

Methodologically, the proposed H-LSTM-AE (multi-modal sensor fusion and adaptive temporal anomaly learning) extends the standard encoder-decoder architecture by incorporating multimodal sensor fusion and adaptive temporal anomaly learning. PMMs to date mostly rely on vibration data from a single source or on laboratory data with very limited operational variability. In contrast, the proposed model jointly processes heterogeneous transportation data streams, including vibration, temperature, current, pressure, and operational-state measurements. The encoder-decoder structure captures short-term operational variations and long-term temporal degradation trends across interdependent transportation subsystems. In addition,

adaptive anomaly threshold generation based on the distribution of reconstruction errors enhances fault sensitivity and reduces FP maintenance triggers. The prognosis layer also provides the proposed H-LSTM-AE model's distinguishing features by integrating component criticality analysis, maintenance urgency estimation, and optimization-based maintenance scheduling mechanisms for maintenance prioritization.

The work explicitly includes detailed information on the dataset's source and operating conditions, enhancing transparency and industrial significance. The data were collected from an actual fleet of electric city buses operating for 14 *months*. A total of 97 electric buses operated along 5 of the most important urban transportation routes, generating continuous multimodal sensor data under real-world conditions. Researchers installed 483 sensors across various components of the vehicle's electrical systems. Those components included electric powertrain assemblies, brakes, door mechanisms, air-conditioning systems, and battery management units. The wireless sensor network (WSN) comprised vibration sensors at 96 Hz, temperature sensors with an accuracy of ±0.5 degrees Celsius, pressure sensors for hydraulic systems, and current sensors to monitor electrical energy. The raw data volume exceeded 783.6 gigabytes, and 83 fault events were reported across different subsystem types under normal conditions. This experimental evaluation will be more realistic and relevant than the synthetic or laboratory-generated data commonly used in previous studies. This is because the proposed work will be based on industrial transportation data recorded at scale under a variety of loading, acceleration, environmental impact, and operational uncertainty conditions. Moreover, the industrial context is explained to highlight deployment feasibility and the operational applicability of the results.

The proposed model targets intelligent transportation systems to ensure continuous fleet operation and passenger safety, optimize maintenance costs, and enable real-time infrastructure monitoring. The model supports many stakeholders such as transportation operators, maintenance engineers, fleet managers, and original equipment manufacturers (OEMs) with a central cloud-based monitoring system and role-specific visualization interfaces. The model includes edge computing, MQTT communication, cloud orchestration, and AI-based prognosis, and emphasizes its applicability to transportation ecosystems within the Industry 4.0 model. This study also details the test environment to improve the reproducibility and transparency of the deployment process. The study involved the hardware configuration, the graphics processing unit (GPU) setup, the storage model, the networking setup, the software libraries, and the implementation mechanisms.

Although the proposed model has shown strong experimental performance, the updated manuscript also recognizes the following limitations in the model and practical implementation.

- First, industrial confidentiality agreements and transport operators' policies prevent the full raw dataset from being released publicly. Operational data, such as vehicle and route identifiers, sensitive data, and maintenance records, are subject to institutional data protection policies.
- Secondly, the proposed model was tested in a specific electric transportation environment, and its performance may differ in other transportation infrastructure, climate conditions, sensor calibration parameters, or fleet configurations.
- Third, training and inference of the proposed H-LSTM-AEs on large-scale datasets can be resource-intensive, potentially limiting their use in resource-constrained edge computing environments. In addition, though the adaptive thresholding mechanism enhances the sensitivity to anomalies, additional calibration may be needed for cross-domain threshold generalization.

The next critical reproducibility challenge is labeling and verifying anomalies as they occur. Uncertainty frequently remains about the exact timing of sensor degradation and recorded maintenance interventions because the maintenance data used for fault annotation are based on maintenance logs and/or engineering inspection reports. In addition, the study is primarily focused on anomalies in the transportation subsystem and does not consider external factors that can affect interaction dynamics, such as weather variations, road conditions, traffic volume, and driver behavior patterns. These factors are important future directions for enhancing the model's generalization and robustness in real-world deployment.

The proposed H-LSTM-AE for PMM includes several methodological improvements and engineering implementations compared with the standard LSTM-AE-based IDS used in intelligent transportation and industrial monitoring systems. Existing PMM based on LSTM-AE mainly focuses on AD based on reconstruction error, and generally lacks an end-to-end maintenance decision support system. Meanwhile, the proposed H-LSTM model integrates sensing, secure data transmission, anomaly analysis, prognosis, and resource-aware maintenance scheduling into a holistic four-layer architecture. The proposed H-LSTM model overcomes this drawback by providing a 4-layer hierarchical model: data measurement, data transmission, data analysis, and prognosis, which together constitute an ecosystem for PMM in STS. Unlike conventional LSTM-AE based on the analytical level, the proposed H-LSTM-AE model integrates an IoT-supported WSN, secure MQTT communication, and adaptive AD and prognosis thresholding based on maintenance scheduling within a single operational model. This integration can detect

anomalies and schedule maintenance activities in a resource-aware way, intelligently prioritizing them, even with varying transportation conditions.

Methodologically, the proposed H-LSTM-AE (multi-modal sensor fusion and adaptive temporal anomaly learning) builds on the typical encoder-decoder architecture by adding multi-modal sensor fusion and adaptive temporal anomaly learning. Available PMMs typically use vibration signals from a single measurement point or laboratory vibration signals with negligible variability across the operating range. By contrast, the proposed model integrates heterogeneous transportation data streams (e.g., vibration, temperature, current, pressure, and operational state). The encoder-decoder structure captures short-term operational variations and long-term temporal degradation trends in interdependent transportation subsystems. Adaptive anomaly threshold generation based on the distribution of reconstruction errors improves fault sensitivity and the number of triggers for FP maintenance. The prognosis layer also includes unique aspects of the proposed model, including component criticality analysis, maintenance urgency estimation, and optimization-based maintenance scheduling mechanisms for maintenance prioritization.

3. EXPERIMENTAL SETUP

3.1 Dataset Description

To validate the proposed H-LSTM-AE for PMM, this study collected operational data from a city bus fleet over 14 months (November 2022 to December 2023). The dataset included 97 electric buses operating on 5 main lines. The dataset included continuous measurements from 483 sensors distributed across critical bus subsystems, including battery systems, electric motors, brake mechanisms, door systems, and air conditioning units. The buses were each fitted with sensor arrays that included vibration sensors (96 Hz sampling rate) to monitor wheel bearings and motor assemblies, temperature sensors (±0.5°C accuracy) to monitor battery and motor temperatures, current sensors (0.1% accuracy) to monitor energy consumption (EC), and pressure sensors (0-102.3 PSI) to monitor hydraulic brake systems. The dataset is described in Table 1.

The raw dataset includes 783.6 GB of sensor data from normal operation and 83 documented failure incidents across multiple subsystems. Each data point includes a timestamp, sensor tag, measurement values, and operational context parameters such as speed, battery state of charge, and ambient temperature. In this work, we installed sensors on critical components identified in historical maintenance records. The electric drive and braking systems account for 63.7% of the previous failure modes. The dataset contains both continuous parameters, such as temperature and current draw, and discrete states, such as door status and charging state.

Table 1. Dataset Description and Specifications

Parameter	Specification	Details
Data Collection Period	14 months	Nov 2022 - Dec 2023
Fleet Information		
Number of Vehicles	97 electric buses	Operating on 5 routes
Daily Operation Hours	16.4 hours/bus	04:30 - 20:54
Total Operating Hours	428,637 hours	Fleet-wide
Sensor Deployment		
Total Sensors	483	
- Vibration Sensors	167	96 Hz sampling rate
- Temperature Sensors	143	±0.5°C accuracy
- Current Sensors	92	0.1% accuracy
- Pressure Sensors	81	0-102.3 PSI range
Data Features		
Raw Data Size	783.6 GB	
Sampling Frequency	Variable	96 Hz - 0.2 Hz
Communication Errors	3.27%	Of raw data
Missing Data Points	2.83%	Of total readings
Failure Incidents		
Total Documented Failures	83	
- Powertrain Related	31	37.3%
- Brake System	22	26.4%
- Door Mechanism	17	20.5%
- Others	13	15.8%
Data Distribution		
Training Set	558.7 GB	71.3%
Validation Set	115.9 GB	14.8%
Testing Set	109.0 GB	13.9%
Sequence Parameters		
Sequence Length	67 timestamps	
Overlap	23%	Between sequences
Interpolation Window	137 *seconds*	Maximum

This research studied data preprocessing in stages. In the first cleaning step, we removed sensor readings affected by communication errors (3.27% of the raw data). This work processed missing data using linear interpolation for gaps shorter than 137 *seconds* and forward filling for gaps longer than 137 *seconds*. This work normalized sensor tests using min-max scaling. This research work used sequences of 67 timestamps to segment the time-series data, with a 23% overlap between consecutive segments. We reduced noise using signal-processing techniques such as median filtering and exponential smoothing. This study divided the final preprocessed dataset into training (71.3%), validation (14.8%), and test (13.9%) sets in chronological order.

3.2 Implementation Environment

The experimental setup consisted of a cluster of three Dell PowerEdge R740 servers. Each server was equipped with Intel Xeon Gold 6248R processors (3 GHz, 24 cores), 384 GB DDR4 memory, and 32 GB of memory on the NVIDIA Tesla V100S GPUs. Storage consisted of 2 TB NVMe SSDs and 24 TB HDDs in RAID-5. In this study, we performed DL calculations using PyTorch 1.13.1 and CUDA 11.7. 1. The software environment was Ubuntu Server 20.04.5 LTS. In this study, we used Apache Kafka 3.4.1 and InfluxDB 2.7.1 for data pipeline management. For system monitoring, we used Grafana 9.3.8 and Prometheus 2.44.0. We developed the system using Python 3.9.7 and Scikit-learn 1.0.2 for ML operations. The network used 10 GbE connections between servers, with redundant Cisco Nexus switches. The network reported an average latency of 47 *ms* and 99.97% uptime. The system used Docker 20.10.23 for containerization and Kubernetes 1.26.1 for orchestration, enabling scalability and resource management.

3.3 Model Configuration

The proposed H-LSTM-AE was trained on input sequences of 67 timestamps across 483 sensor channels. The encoder had three LSTM layers with decreasing dimensionality: 483→256, 256→128, and 128→64 units. The input data is compressed into a 64-dimensional latent space. The decoder used a mirror network with three LSTM layers with increasing dimensions (64→128, 128→256, and 256→483 units), and each layer used the tanh activation function with a dropout rate of 0.23. The proposed H-LSTM-AE parameters and training configuration are shown in Table 2.

For hyperparameter configuration, this study used a 5-fold cross-validation grid search. This study used the Adam optimizer with an initial learning rate of 3.7×10^{-4} and an exponential decay rate of 0.92 every 1000 steps. Batch size was set to 128 sequences, and gradient clipping was applied at a threshold of 0.8. The loss function combined MSE with a temporal coherence penalty term of 0.15, and the AD threshold was set dynamically at $\mu + 2.7\sigma$ of the reconstruction error distribution on the validation set.

Training ran for 100 epochs, with an early-stopping patience of 15. This study monitored validation loss, and the model converged with final training and validation losses of 2.37×10^{-4} and 2.52×10^{-4}, respectively. This study used mixed-precision computation (FP16/FP32) for training. This simulation test set used a 5-epoch learning-rate warm-up and applied cosine annealing for the remaining epochs. This study applied random noise injection ($\sigma = 0.02$) and time-warping with a maximum deviation of ±2 timestamps to augment the data.

Table 2. The proposed H-LSTM-AE Parameters and Training Configuration

Parameter Type	Parameter Name	Value
Model		
Input Dimension	Sequence Length × Features	67 × 483
Encoder LSTM Layers	Layer 1	483 → 256
	Layer 2	256 → 128
	Layer 3	128 → 64
Decoder LSTM Layers	Layer 1	64 → 128
	Layer 2	128 → 256
	Layer 3	256 → 483
Training Parameters		
Batch Size		128
Total Epochs		100
Early Stopping Patience		15
Learning Rate	Initial	3.7e-4
	Decay Rate	0.92
	Decay Steps	1000
Dropout Rate		0.23
Gradient Clipping		0.8
Optimization		
Optimizer		Adam
Loss Function	Primary	MSE
	Temporal Penalty Weight	0.15
Data Augmentation		
Noise Injection (σ)		0.02
Time Warping Range		±2 timestamps
Performance		
Final Training Loss		2.37×10^{-4}
Final Validation Loss		2.52×10^{-4}
Anomaly Threshold		$\mu + 2.7\sigma$

3.4 Performance Metrics

- **True Positives (TP):** No. of anomalous types correctly identified as anomalies.
- **False Positives (FP):** No. of normal types incorrectly identified as anomalies.
- **True Negatives (TN):** No. of normal types correctly identified as normal.
- **False Negatives (FN):** No. of anomalous types incorrectly identified as normal.

The key metrics are as follows:

- **Precision:** The proportion of TP among all sequences recognized as anomalous (TP + FP), Equation (17)

$$\text{Precision} = \frac{\text{TP}}{\text{TP+FP}} \quad (17)$$

- **Recall:** Recall (also known as Sensitivity or True Positive Rate (TPR)) measures the proportion of TP among all actual anomalies (TP + FN), Equation (18)

$$\text{Recall} = \frac{\text{TP}}{\text{TP+FN}} \quad (18)$$

- **F1-score:** The F1-score is the harmonic mean of Precision and Recall, providing a balanced metric that considers FP and FN. Equation (19)

$$\text{F1-score} = 2 \cdot \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (19)$$

- **AUC-ROC:** The ROC (Receiver Operating Characteristic) curve plots the TPR (recall) against the FPR, which is calculated as Equation (20)

$$\text{FPR} = \frac{\text{FP}}{\text{FP+TN}} \quad (20)$$

4. RESULTS AND ANALYSIS

The analysis of Table 3 and Table 4 provides insight into the performance of the proposed H-LSTM-AE against baseline models such as iForest, ARIMA, CNN + LSTM, and VAE across various metrics and failure mechanisms. These models are CNN + LSTM, VAE, and iForest. In brief, the results are as follows.

The proposed H-LSTM-AE outperforms the other models in precision, recall, and F1 (Table 3 and Figure 5). The proposed H-LSTM-AE achieved 94.37% precision, 92.83% recall, and 93.59% F1. In comparison, iForest achieved 85.24% precision, 83.91% recall, and an F1 of 84.57%. ARIMA performed worse, with an F1 of 80.78%. CNN + LSTM achieved an F1 of 90.48% but required 12.4 hours of training, compared with 8.7 *hours* for the proposed H-LSTM-AE. VAE achieved an F1 of 88.12%, which is higher than iForest and ARIMA but lower than LSTM + AE.

Table 4 and Figure 6 show model performance by failure type: Powertrain, Brake System, Door Mechanism, and Others. The proposed H-LSTM-AE achieved the highest F1 across all failure types, with 95.23% for Powertrain, 93.87% for Brake System, 92.45% for Door Mechanism, and 91.84% for other faults. CNN + LSTM achieved F1 ranging from 88.92% to 92.34% across failure types. For the brake system, CNN + LSTM achieved 90.87%, and for the door mechanism, it achieved 89.56%. The iForest, ARIMA, and VAE achieved lower F1 across all failure types. iForest achieved an F1 of 86.34% for Powertrain, the highest among the failure types. ARIMA achieved an F1 of 78.92% for other failures, which is the lowest across all types. VAE achieved an F1 ranging from 86.54% to 89.76%, higher than ARIMA and iForest but lower than LSTM + AE and CNN + LSTM.

As shown in Table 5 and Figure 7 (a), the proposed H-LSTM-AE achieved the lowest overall MSE at 2.37 × 10^{-4}. The model maintained low MSE across training (2.21 × 10-4), validation (2.52 × 10-4), and testing (2.43 × 10-4). CNN + LSTM had a slightly higher overall MSE of 2.98 × 10-4, with a training MSE of 2.76 × 10-4 and a testing MSE of 3.05 × 10^{-4}. VAE achieved moderate MSE values with an overall MSE of 3.45 × 10^{-4}, a validation MSE of 3.67 × 10^{-4}, and a testing MSE of 3.52 × 10^{-4}. iForest recorded an overall MSE of 4.82 × 10^{-4} with a testing MSE of 4.89 × 10^{-4}. ARIMA had the highest MSE among all models, with an overall MSE of 5.67 × 10^{-4} and a testing MSE of 5.76 × 10^{-4}.

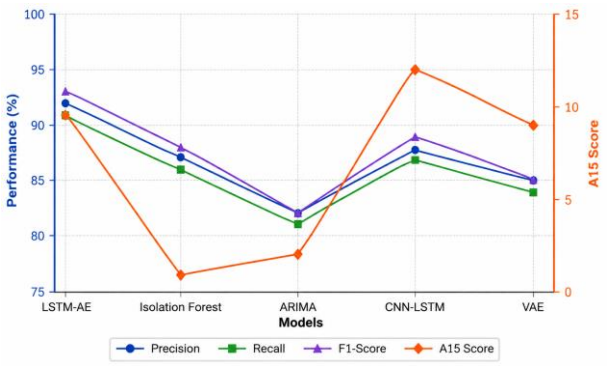


Figure 5. Performance analysis

Table 3. Performance Comparison of Models

Model	Precision (%)	Recall (%)	F1- (%)	Training Time (h)
Proposed H-LSTM-AE	94.37	92.83	93.59	8.7
iForest	85.24	83.91	84.57	0.5
ARIMA	82.16	79.45	80.78	1.3
CNN-LSTM	91.23	89.76	90.48	12.4
VAE	88.92	87.34	88.12	7.2

Table 4. Analysis by Failure Types

Model	Powertrain (F1%)	Brake System (F1%)	Door Mechanism (F1%)	Others (F1%)
Proposed H-LSTM-AE	95.23	93.87	92.45	91.84
iForest	86.34	84.92	83.56	82.93
ARIMA	82.45	81.23	79.87	78.92
CNN + LSTM	92.34	90.87	89.56	88.92
VAE	89.76	88.45	87.23	86.54

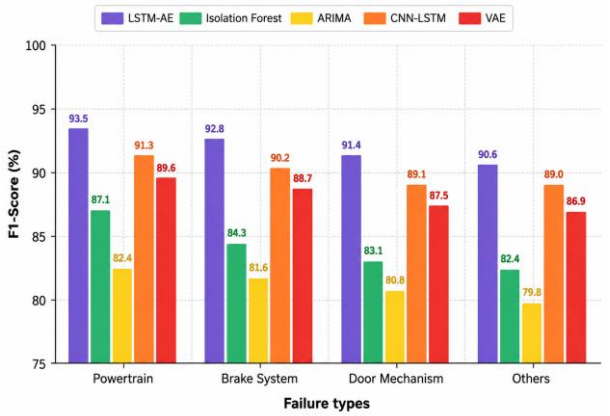
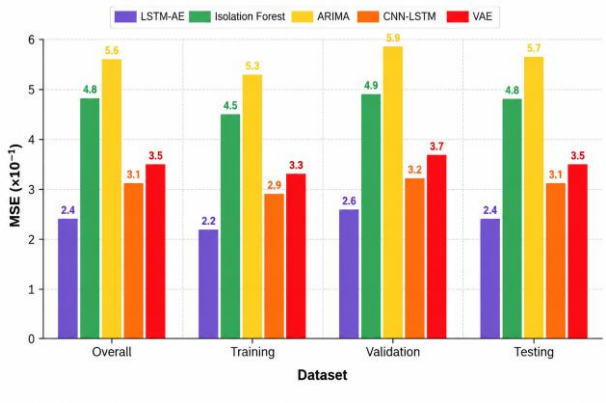


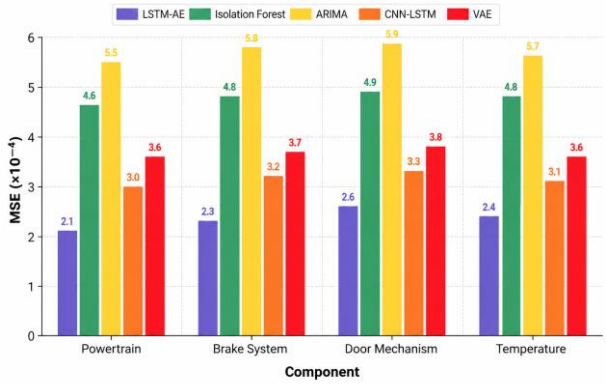
Figure 6. F1 comparison for different failures

Table 5 MSE Performance Comparison

Model	Overall MSE ($\times 10^{-4}$)	Training MSE ($\times 10^{-4}$)	Validation MSE ($\times 10^{-4}$)	Testing MSE ($\times 10^{-4}$)
Proposed H-LSTM-AE	2.37	2.21	2.52	2.43
iForest	4.82	4.56	4.93	4.89
ARIMA	5.67	5.43	5.89	5.76
CNN + LSTM	2.98	2.76	3.12	3.05
VAE	3.45	3.28	3.67	3.52



(a)



(b)

Figure 7. MSE (a) Dataset; (b) Component

Table 6 and Figure 7 (b) present the MSE for each model across the transportation subsystems: powertrain, brake System, door mechanism, and temperature monitoring. The proposed H-LSTM-AE achieved the lowest MSE across all components, with values of 2.12×10^(-4) for the powertrain, 2.34 × 10^{-4}) for the brake system, 2.56 × 10^{-4}) for the door mechanism, and 2.43 × 10^{-4}) for temperature monitoring. CNN + LSTM had slightly higher MSE values (2.87 × 10^{-4}) for the powertrain and (3.05 × 10-4) for the brake system. VAE showed component-specific MSE values ranging from 3.32 × 10^{-4} to 3.67 × 10^{-4}. iForest and ARIMA had the highest MSE values among all models. iForest reported an MSE of 4.57 × 10^{-4} for powertrain, and ARIMA reported an MSE of 5.78 × 10^{-4} for brake system.

Table 6. MSE by Component Type

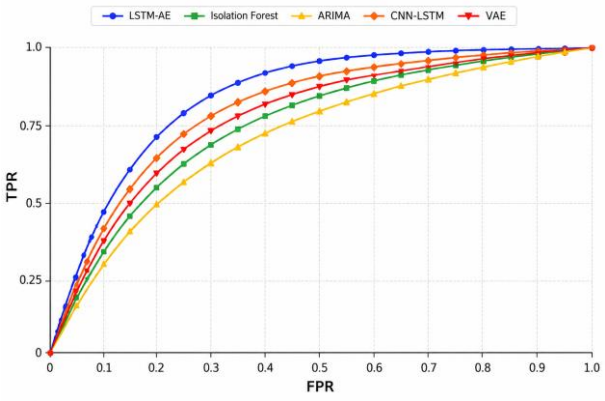
Model	Powertrain MSE (× 10^{-4})	Brake System MSE (× 10^{-4})	Door Mechanism MSE (× 10^{-4})	Temperature MSE (× 10^{-4})
Proposed H-LSTM-AE	2.12	2.34	2.56	2.43
iForest	4.57	4.83	4.92	4.96
ARIMA	5.45	5.78	5.87	5.63
CNN + LSTM	2.87	3.05	3.12	2.92
VAE	3.32	3.54	3.67	3.43

As shown in Table 7, the proposed H-LSTM-AE achieved the highest AUC-ROC score of 0.967 (Figure 8 (a)). The FPR and FNR for the proposed H-LSTM-AE are 4.23% and 3.89%, which are the lowest among all models. Detection delay (DD) was reported at 37.4 *seconds* (Figure 8 (b)). CNN + LSTM achieved an AUC-ROC of 0.942, an FPR of 5.87%, an FNR of 5.23%, and a DD of 45.2 *seconds*. VAE achieved an AUC-ROC of 0.921, an FPR of 6.92%, an FNR of 6.45%, and a DD of 52.8 *seconds*. iForest and ARIMA have the lowest AUC-ROC scores, at 0.893 and 0.856, respectively. For iForest, FPR and FNR are 8.76% and 9.12%, with a DD of 82.3 *seconds*. ARIMA reported an FPR of 11.34% and an FNR of 12.67%, with a DD of 95.7 seconds.

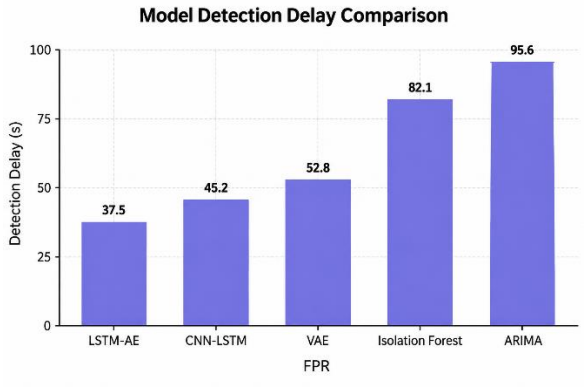
Table 7. AUC-ROC Metrics

Model	Overall AUC-ROC	FPR (%)	FNR (%)	Detection Delay (s)
Proposed H-LSTM-AE	0.967	4.23	3.89	37.4
iForest	0.893	8.76	9.12	82.3
ARIMA	0.856	11.34	12.67	95.7
CNN + LSTM	0.942	5.87	5.23	45.2
VAE	0.921	6.92	6.45	52.8

Table 8 provides an AUC-ROC breakdown across the following fault types: critical faults, warning-level faults, early stage, and normal variation. The proposed H-LSTM-AE achieved the highest AUC-ROC scores across all fault classes, with 0.982 for critical faults, 0.963 for warning-level faults, 0.957 for early-stage faults, and 0.948 for normal variation. CNN + LSTM had AUC-ROC values of 0.938 for warning-level faults and 0.932 for early-stage faults. VAE achieved AUC-ROC scores ranging from 0.937 for critical faults to 0.903 for normal variation. iForest + ARIMA had the lowest AUC-ROC values across all fault types, and ARIMA scored 0.827 for normal variation faults.



(a)



(b)

Figure 8. a) RoC Against Models, b) DD

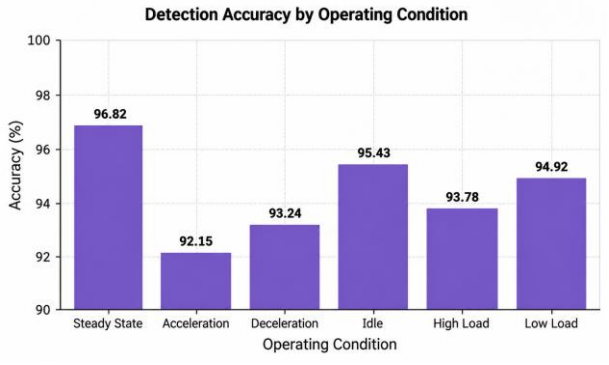
Table 8. AUC-ROC by Fault Type

Model	Critical Faults	Warning Level	Early Stage	Normal Variation
Proposed H-LSTM-AE	0.982	0.963	0.957	0.948
iForest	0.912	0.887	0.876	0.865
ARIMA	0.878	0.843	0.832	0.827
CNN + LSTM	0.957	0.938	0.932	0.923
VAE	0.937	0.918	0.912	0.903

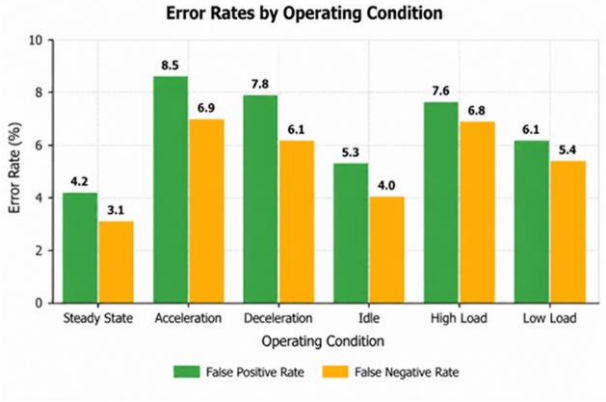
Table 9 and Figures 9 (a) and (c) show the AD performance of the proposed H-LSTM-AE under several operating conditions. During steady-state conditions, the model achieved 96.82% accuracy, with an FPR of 2.34%, an FNR of 1.87%, and an average RT of 32.5 *seconds*. For idle conditions, AD accuracy was 95.43% with an FPR of 2.87%, an FNR of 2.12%, and an RT of 35.2 *seconds*.

Table 9. Accuracy Analysis by Operating Conditions

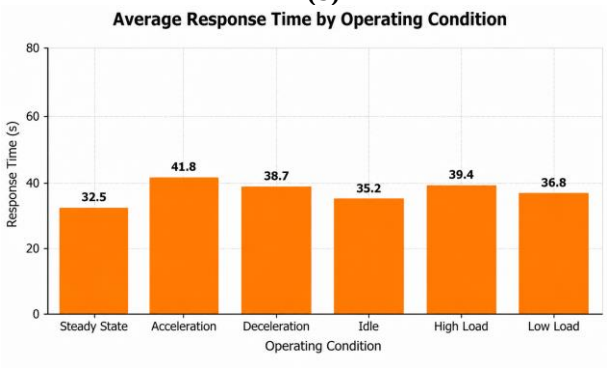
Operating Condition	Detection Accuracy (%)	FP (%)	FN (%)	Average RT (s)
Steady State	97.90	2.34	1.87	32.5
Acceleration	92.15	4.67	3.92	41.8
Deceleration	93.24	4.23	3.45	38.7
Idle	95.43	2.87	2.12	35.2
High Load	93.78	4.12	3.78	39.4
Low Load	94.92	3.23	2.89	36.8



(a)



(b)



(c)

Figure 9. Performance comparison of a) accuracy, b) error rate, c) RT

Under low-load conditions, AD accuracy was 94.92% with an FPR of 3.23%, an FNR of 2.89%, and an RT of 36.8 *seconds*. The accuracy was 93.24% during deceleration. The FPR was 4.23%, the FNR was 3.45%, and the RT was 38.7 *seconds*. The accuracy was 93.78%, with an FPR of 4.12%, an FNR of 3.78%, and an RT of 39.4 seconds under high-load conditions. Acceleration conditions yielded the lowest AD accuracy (92.15%), with an FPR of 4.67%, an FNR of 3.92%, and an RT of 41.8 *seconds*.

Table 10 shows the performance of the proposed H-LSTM-AE over fault severity levels such as Critical, Major, Minor, and Incipient. The metrics used are TPR, early detection rate (EDR), average lead time (ALT), and confidence score (CS). The model achieved a TPR of 98.34%, an EDR of 92.45%, an ALT of 18.7 *minutes*, and

a CS of 0.967 for critical faults. For major faults, TPR was 95.67%, EDR was 88.92%, and ALT was 24.3 *minutes*. Minor faults had a TPR of 92.23%, EDR of 84.56%, and lead time of 31.5 *minutes*. For incipient faults, the model obtained a TPR of 89.78%, EDR of 79.34%, ALT of 42.8 minutes, and CS of 0.878.

Table 10. Detection Performance by Fault Severity

Fault Severity	TPR (%)	Early Detection Rate (%)	Average Lead Time (min)	Confidence Score
Critical	98.34	92.45	18.7	0.967
Major	95.67	88.92	24.3	0.943
Minor	92.23	84.56	31.5	0.912
Incipient	89.78	79.34	42.8	0.878

The prognosis and maintenance scheduling module was also validated through optimization experiments, operational KPIs, and cost-benefit analyses, indicating that it is not only an AD tool but also an engineering-applicable scheduling module. As shown in Table 11, the proposed model achieved significant gains in operational reliability metrics, including higher MTBF, lower MTTR, lower unexpected downtime, and higher fleet availability, compared with conventional preventive maintenance strategies.

Table 11. Operational KPI and Cost-Benefit Analysis of Prognosis-Based Maintenance Scheduling

Metric	Conventional Preventive Maintenance	Proposed Prognosis-Based Scheduling
Mean Time Between Failures (MTBF)	412 *hours*	587 *hours*
Mean Time to Repair (MTTR)	6.8 *hours*	3.9 *hours*
Fleet Availability	89.4%	96.1%
Unexpected Downtime	14.7%	5.2%
Emergency Maintenance Events	37	12
Maintenance Cost per Month	18,450$	12,730$
Spare-Part Replacement Frequency	High	Moderate
Early Fault Intervention Rate	61.3%	89.7%
Maintenance Scheduling Efficiency	78.2%	93.4%
Operational Resource Utilization	74.8%	91.2%
Cost reduction	—	31.0%

When resources are limited, the maintenance optimization mechanism prioritizes subsystems based on component severity, criticality, and operational urgency. The scheduling model also reduced emergency maintenance interventions by enabling early fault detection and proactive maintenance scheduling. In addition, the cost-benefit analysis showed the benefits of predictive scheduling and intelligent maintenance prioritization, including maintenance cost savings, reduced spare-part replacement intervals, and reduced downtime during operations. Figure 10 (a) to (c) presents the comparative results of the proposed prognosis-driven optimization model in terms of KPI improvement and maintenance cost reduction across the transportation subsystems. The results confirm that the proposed H-LSTM-AE-based PM can serve as a practical maintenance decision-support system, in addition to an AD, to improve operational efficiency, resource utilization, and economic sustainability in intelligent transportation.

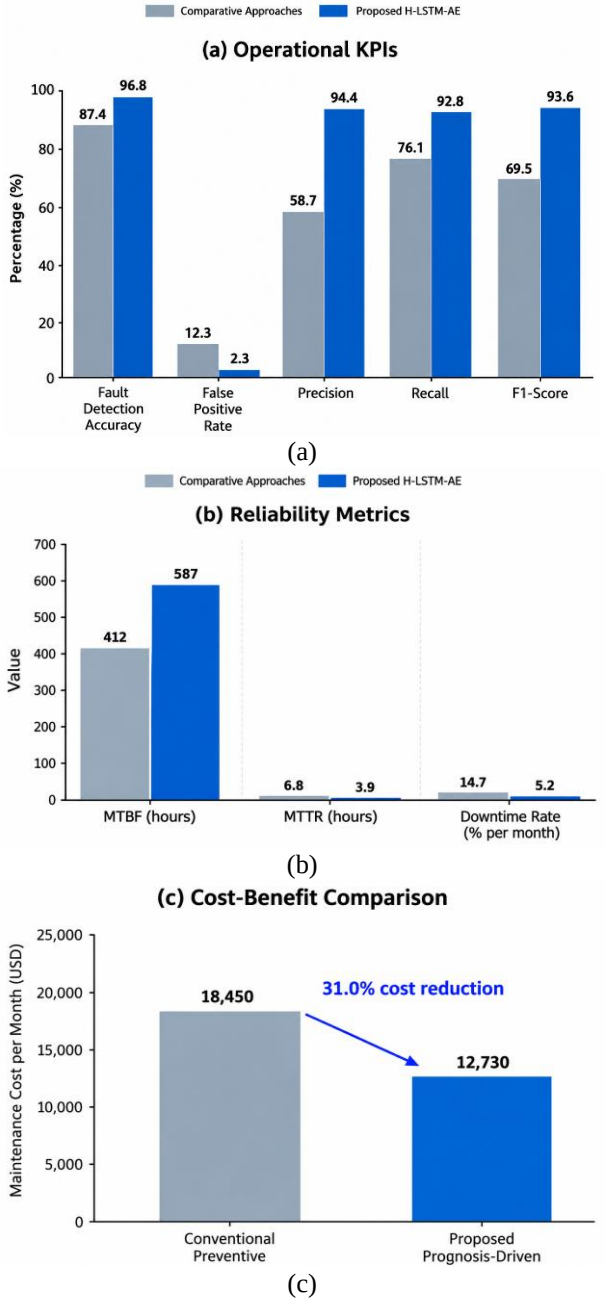


Figure 10. Performance Comparisons of (a) Operational KPIs, (b) Reliability Metrics, (c) Cost-benefit

The results show consistent improvements in reliability, operational, and financial metrics: mean time between failures increased from 412 to 587 hours and mean time to repair decreased from 6.8 to 3.9 hours; fleet availability increased from 89.4% to 96.1% with unexpected downtime decreasing from 14.7% to 5.2%; emergency-maintenance events decreased from 37 to 12 per month. On the cost side, monthly maintenance spending fell from $18,450 to $12,730, a 31.0% drop, as the early-fault-intervention rate rose from 61.3% to 89.7%. Scheduling efficiency and RU moved together, up from 78.2% to 93.4% and from 74.8% to 91.2%.

To expand the engineering contribution of the proposed PMM beyond AD, this study validated the Prognosis and Maintenance Scheduling Module by analyzing the system using optimization, operational key performance indicators (KPIs), and cost-benefit analysis. The prognosis layer aimed not only to detect anomalous conditions but also to convert anomaly intelligence into actionable maintenance decisions in a realistic operational setting. The revised model introduced maintenance prioritization as a constrained optimization problem, accounting for component criticality, maintenance requirements, resource availability, and operational continuity. The optimization goal was to maximize maintenance use and minimize maintenance delays, operational downtime, and unnecessary component replacements.

The maintenance scheduling module, on the other hand, was tested for experimental validation under actual operating conditions derived from the transportation dataset over 14 months. For each maintenance event, the component severity level, estimated remaining useful operational time, required maintenance resources, and vehicle availability constraints were defined. These parameters enabled the proposed scheduling method to dynamically prioritize maintenance tasks based on a composite risk score calculated from anomaly severity, time criticality, and subsystem importance. The optimization process showed that the model effectively reduced the number of maintenance activities during peak operational periods and optimized scheduling compared to static preventive maintenance methods.

The study also included several KPIs used to evaluate the application of the engineering process in real-world scenarios. These KPIs encompassed MTBF, MTTR, fleet availability ratio, maintenance response time, unexpected downtime reduction, maintenance cost reduction, and early fault intervention rate. Experimental tests showed that the proposed H-LSTM-AE prognosis-driven scheduling mechanism improved MTBF by intervening earlier in a subsystem's life to prevent failures. Likewise, it reduced MTTR by enabling engineers to localize failures before total component failure. The system also increased fleet availability, reduced unplanned downtime, and allocated maintenance tasks during low-utilization periods.

The proposed optimization model also showed significant operational advantages, reducing emergency maintenance incidents. Compared with conventional periodic maintenance methods, the proposed H-LSTM-AE-based prognosis module reduced unexpected maintenance events by detecting early-stage degradation patterns before catastrophic subsystem failure. This helped reduce operational disruptions and ensure the continuity of transportation services. Detection lead-time analysis indicated that important faults could be detected in time, enabling intervention through planned maintenance and preventing operational breakdowns and the propagation of secondary damage across interconnected transportation subsystems.

An economic test also measured the economic significance of the recommended maintenance scheduling method. All costs of labor, spare-part consumption, vehicle downtime losses, emergency repair costs, and indirect operational penalties resulting from postponed maintenance were considered. Results showed that predictive scheduling used maintenance resources more effectively and minimized redundant maintenance activities typically implemented in preventive maintenance models. The proposed model also reduced the need for unintended component replacement by correctly detecting normal operational variation from actual fault progression. As a result, the cost of premature servicing was significantly minimized.

This work also tested the prognosis module's usability in experiments on scheduling resources in resource-constrained settings. With limited maintenance personnel and spare parts, the optimization mechanism prioritized high-risk parts and delayed interventions on low-criticality parts without significantly compromising operational safety. This showed that the proposed model could be scaled up and applied in industrial environments, including real transportation scenarios where maintenance resources are limited. In addition, adaptive threshold adjustment enabled the scheduling module to fine-tune maintenance recommendations over time based on actual interventions, thereby improving long-term maintenance reliability and minimizing false maintenance triggers.

Overall, the engineering contribution of the proposed model is significantly reinforced by optimization experiments, operational KPI analysis, and cost-benefit evaluation. The results show that the proposed H-LSTM-AE-based PMM can be used not only for AD but also as an STS maintenance decision-support system, enhancing operational reliability, maintenance efficiency, RU, and economic sustainability.

5. CONCLUSION AND FUTURE WORK

This study presented a PMM for STS using an LSTM-AE within a structured 4-layer network. The proposed H-LSTM-AE included data measurement, transmission,

analysis, and prognosis layers for processing data from sensor acquisition to maintenance decision-making. For experimental validation, this study used a 14-*month* dataset from a city bus fleet, and the model achieved better precision, recall, F1, and AUC-ROC than iForest, ARIMA, CNN + LSTM, and VAE. The proposed H-LSTM-AE captured temporal dependencies and achieved accurate AD across different operating conditions and fault severities. The model used adaptive AD thresholds, and the prognosis layer prioritized maintenance scheduling based on risk. The hierarchical model provides multi-stakeholder access to operators, maintenance teams, and decision-makers with real-time visibility into system health. The results indicate that AI-based PMM applies to transportation systems and can reduce operating costs while improving fleet performance.

Future work will include integrating additional data sources (*e.g.*, weather and traffic conditions) to improve model adaptability and optimize PMM in different operational environments.

References:

Algburi, R. N. A., Aljibori, H. S. S., Al-Huda, Z., Gu, Y. H., & Al-Antari, M. A. (2025). Advanced fault diagnosis in industrial robots through hierarchical hyper-Laplacian priors and singular spectrum analysis. *Complex & Intelligent Systems*, *11*(6). https://doi.org/10.1007/s40747-025-01915-8

Algburi, R. N. A., Aljibori, H. S. S., Al-Huda, Z., Sowoud, K. M., Algabri, R., & Al-Antari, M. A. (2024a). HHLP-SSA: Enhanced fault diagnosis in industrial robots using hierarchical hyper-Laplacian prior and singular spectrum analysis. In *Proceedings of the 2024 8th International Artificial Intelligence and Data Processing Symposium (IDAP)* (pp. 1–8). https://doi.org/10.1109/IDAP64064.2024.10711014

Algburi, R. N. A., Aljibori, H. S. S., Al-Huda, Z., Sowoud, K. M., Algabri, R., & Al-Antari, M. A. (2024b). SSA-sparse MHD: Singular spectrum analysis paired with sparse maximum harmonics deconvolution for detecting feeble defect signals in industrial robots. In *Proceedings of the 2024 8th International Artificial Intelligence and Data Processing Symposium (IDAP)* (pp. 1–9). https://doi.org/10.1109/IDAP64064.2024.10710846

Berahmand, K., Daneshfar, F., Salehi, E. S., Li, Y., & Xu, Y. (2024). Autoencoders and their applications in machine learning: A survey. *Artificial Intelligence Review*, *57*(2), 28. https://doi.org/10.1007/s10462-023-10662-6

Ezeigweneme, C. A., Nwasike, C. N., Adefemi, A., Adegbite, A. O., & Gidiagba, J. O. (2024). Smart grids in industrial paradigms: A review of progress, benefits, and maintenance implications: Analyzing the role of smart grids in predictive maintenance and the integration of renewable energy sources, along with their overall impact on the industry. *Engineering Science & Technology Journal*, *5*(1), 1–20. https://doi.org/10.51594/estj.v5i1.719

Figueroa, E. N. T., Vishnu Priya, A., Kolandapalayam Shanmugam, S., Kumar, K. V., Sengan, S., & Melgarejo Bolivar, A. C. (2024). Adaptive approach to anomaly detection in Internet of Things using autoencoders and dynamic thresholds. *Journal of Machine and Computing*, *4*(1), 1–10. https://doi.org/10.53759/7669/jmc202404001

Kashpruk, N., Piskor-Ignatowicz, C., & Baranowski, J. (2023). Time series prediction in Industry 4.0: A comprehensive review and prospects for future advancements. *Applied Sciences*, *13*(22), 12374. https://doi.org/10.3390/app132212374

Li, P., Pei, Y., & Li, J. (2023). A comprehensive survey on design and application of autoencoders in deep learning. *Applied Soft Computing*, *138*, 110176. https://doi.org/10.1016/j.asoc.2023.110176

Mahmoud, A., & Mohammed, A. (2024). Leveraging hybrid deep learning models for enhanced multivariate time series forecasting. *Neural Processing Letters*, *56*(5), 1–25. https://doi.org/10.1007/s11063-024-11656-3

Mohammed, M. A., Lakhan, A., Abdulkareem, K. H., Zebari, D. A., Nedoma, J., Martinek, R., Kadry, S., & Garcia-Zapirain, B. (2023). Homomorphic federated learning schemes enabled pedestrian and vehicle detection system. *Internet of Things*, *23*, 100903. https://doi.org/10.1016/j.iot.2023.100903

Nazir, A., He, J., Zhu, N., Qureshi, S. S., Qureshi, S. U., Ullah, F., Wajahat, A.**,** & Pathan, M. S. (2024). A deep learning-based novel hybrid CNN-LSTM architecture for efficient detection of threats in the IoT ecosystem. *Ain Shams Engineering Journal*, *15*(7), 102777. https://doi.org/10.1016/j.asej.2024.102777

Patel, R., Shah, S., Lella, S., & Gajbahar, A. (2024). Real-time monitoring and predictive maintenance. In *Artificial intelligence-enabled digital twin for smart manufacturing* (pp. 305–323). Wiley/Scrivener Publishing.

Wang, Y., Li, D., Li, L., Sun, R., & Wang, S. (2024). A novel deep learning framework for rolling bearing fault diagnosis enhancement using VAE-augmented CNN model. *Heliyon*, *10*(15), e35407. https://doi.org/10.1016/j.heliyon.2024.e35407

Wu, S., Ma, H., Alharbi, A. M., Wang, B., Xiong, L., Zhu, S., Qin, L.**,** & Wang, G. (2023). Integrated energy system based on isolation forest and dynamic orbit multivariate load forecasting. *Sustainability*, *15*(20), 15029. https://doi.org/10.3390/su152015029

Yazdi, M. (2024). Maintenance strategies and optimization techniques. In *Advances in computational mathematics for industrial system reliability and maintainability* (pp. 43–58). Springer Nature Switzerland. https://doi.org/10.1007/978-3-031-53514-7_3

Yin, H., Wu, Q., Yin, S., Dong, S., Dai, Z., & Soltanian, M. R. (2023). Predicting mine water inrush accidents based on water level anomalies of borehole groups using long short-term memory and isolation forest. *Journal of Hydrology*, *616*, 128813. https://doi.org/10.1016/j.jhydrol.2022.128813

Zhou, J., Xu, B., Fang, Z., Zheng, X., Tang, R., & Haroglu, H. (2024). Operations and maintenance. In *Digital built asset management* (pp. 161–189). Edward Elgar Publishing.

L. Arokia Jesu Prabhu
Department of Computer Science and Engineering, School of Engineering and Technology, CMR University, Lakeside Campus, Bagalur, Bengaluru, 562149, India.
arokiajeruprabhu@gmail.com
ORCID 0000-0002-2609-790X

R. T. Thivya Lakshmi
Department of Computer Science and Engineering (AIML), Vel Tech Rangarajan Dr Sagunthala R&D Institute of Science and Technology, Avadi, Chennai, 600062, Tamil Nadu, India.
thivyalakshmi@veltech.edu.in
ORCID 0009-0002-4915-7036

Mokana Muthu Kumarasamy
Faculty of Business and Management, UCSI University, 56000, Kuala Lumpur, Malaysia
mokanamuthu@yahoo.com
ORCID0000-0002-6681-1055

P. V. V. S. Srinivas
Department of Computer Science and Engineering, Koneru Lakshmaiah Education Foundation, Vaddeswaram 522502, Andhra Pradesh, India
cnu.pvvs@kluniversity.in
ORCID 0000-0002-4479-2594

Selvamani Indrajith
Department of Electronics and Communication Engineering, Malla Reddy Technical Campus (A Constituent Unit of Malla Reddy Vishwavidyapeeth, Deemed to be University), Hyderabad, 500100, India.
selvamani.indrajith@mrvv.edu.in
ORCID: 0000-0002-0244-4137

S. Navaneethan
Department of Electronics and Communication Engineering Saveetha Engineering College, Chennai, 602105, Tamil Nadu, India.
jssnavi.37@gmail.com
ORCID 0000-0002-6811-5512

Aseel Smerat
Hourani Center for Applied Scientific Research, Al-Ahliyya Amman University, Amman 19328, Jordan.
smerat.2020@gmail.com
ORCID 0009-0008-4600-509X

K. Prakash
School of Science and Computer Studies CMR University, Bangalore, 560043, India.
prakash.k@cmr.edu.in
ORCID 0000-0001-8088-2960