RESEARCH ARTICLE

A hybrid transformer-BiLSTM model optimized with Firefly Algorithm for network traffic anomaly detection

Debiao Luo[1*], **Weijie Wang**[1], **Xinyue Liu**[1], **Wen Yang**[1], **Ke Hu**[1], **Jia Zhang**[2]

1 Information Network Center, Chengdu University, Chengdu, China, **2** Library, Chengdu University, Chengdu, China

* luodb@cdu.edu.cn

Abstract

Network Traffic Anomaly Detection (NTAD) is essential for proactive cyber defense against increasingly sophisticated threats. This paper presents a data-driven framework that integrates adaptive signal decomposition, a hybrid attention-recurrent architecture, and metaheuristic optimization for timely anomaly prediction. Raw traffic sequences are first preprocessed via Empirical Mode Decomposition (EMD) to mitigate non-stationarity and suppress noise, yielding denoised intrinsic mode functions. The refined signal is then modeled by a hybrid deep network that couples a multi-head self-attention mechanism—capturing global, long-range dependencies—with a Bidirectional Long Short-Term Memory (BiLSTM) network that encodes bidirectional temporal dynamics. To circumvent the sensitivity of deep models to hyperparameter selection, the Firefly Algorithm (FA) is employed for automated, population-based optimization. Extensive evaluations on benchmark datasets demonstrate that the proposed EMD-FA-Transformer-BiLSTM model attains state-of-the-art performance, outperforms baseline and state-of-the-art models across all evaluated metrics, with statistically significant improvements in both regression error and classification F_1-score.

OPEN ACCESS

Citation: Luo D, Wang W, Liu X, Yang W, Hu K, Zhang J (2026) A hybrid transformer-BiLSTM model optimized with Firefly Algorithm for network traffic anomaly detection. PLoS One 21(6): e0341920. https://doi.org/10.1371/journal.pone.0341920

Editor: Armin Moghimi, Leibniz University Hannover, GERMANY

Received: January 10, 2026

Accepted: June 2, 2026

Published: June 17, 2026

Data availability statement: All relevant data are within the paper and its Supporting Information files.

Funding: The author(s) received no specific funding for this work.

1 Introduction

The rapid expansion of network infrastructure, propelled by the proliferation of Internet of Things (IoT) devices, cloud computing, and 5G technologies, has fundamentally reshaped the digital landscape [1–3]. While such advances enable unprecedented levels of connectivity and service provisioning, they concurrently broaden the attack surface, rendering networks increasingly susceptible to malicious activities including Distributed Denial-of-Service (DDoS) attacks [4], data exfiltration [5], and malware propagation [6]. Against this backdrop, Network Traffic Anomaly Detection (NTAD) has become a cornerstone of contemporary cybersecurity frameworks [7–10]. NTAD aims to identify deviations from established behavioral baselines,

Competing interests: The authors have declared that no competing interests exist.

which frequently constitute early indicators of security breaches or incipient performance degradation. The capacity to furnish timely alerts for anomalous traffic fluctuations—whether abrupt surges characteristic of volumetric attacks or atypical declines signaling service disruption—is thus a strategic imperative for preserving the availability, integrity, and confidentiality of modern networked systems.

Reliable and timely anomaly prediction remains a formidable challenge, owing principally to the intrinsic characteristics of network traffic data [11]. Real-world traffic exhibits pronounced non-stationarity [12] and nonlinearity [13], with statistical moments evolving continuously under the influence of diurnal cycles, heterogeneous user behaviors, and dynamic network conditions. The signal is further corrupted by ambient noise and manifests complex, multi-scale structures [14], wherein transient bursts are superimposed upon gradual trends. Conventional detection methodologies—encompassing statistical process control [15] and signature-based Intrusion Detection Systems (IDS) [16]-are ill-equipped to contend with such intricacy. These approaches tend to be reactive in nature, exhibit limited adaptability to emerging attack vectors, and fail to adequately model the sophisticated temporal dependencies and non-stationary dynamics inherent in contemporary network traffic.

The emergence of machine learning [17–20] and, more recently, deep learning has opened promising avenues for overcoming these limitations [21–24]. Both supervised and unsupervised paradigms are capable of extracting complex patterns from historical observations. Notably, deep recurrent architectures such as Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) networks have demonstrated considerable efficacy in sequential modeling tasks, owing to their capacity for capturing temporal dependencies [25–29]. Nonetheless, these models are not without limitations. Their inherently sequential and localized processing renders them less adept at capturing long-range, global correlations spanning extended temporal horizons—an essential requirement for detecting stealthy, low-and-slow attack patterns. For instance, [30] introduces an RCLSTM architecture with stochastic neuron connectivity to alleviate the computational burden of conventional LSTM while preserving competitive time-series forecasting accuracy, thereby enhancing suitability for latency- or resource-constrained deployments such as telecommunications. Similarly, [31] demonstrates that LSTM networks can successfully forecast equity price movements and generate statistically significant excess returns over the 1992–2009 period, attributing profitability primarily to the exploitation of short-term reversal signals in high-volatility stocks. In [32], an LSTM-based framework is proposed for Remaining Useful Life (RUL) estimation in dynamical systems, wherein raw sensor streams are transformed into a health indicator, achieving superior degradation tracking accuracy as validated on the NASA C-MAPSS dataset. Furthermore, the performance of deep learning models is critically contingent upon hyperparameter selection, rendering manual tuning both inefficient and suboptimal in practice.

To address the challenge of non-stationarity, advanced signal processing techniques have been incorporated as a preprocessing stage [33]. Empirical Mode Decomposition (EMD) constitutes a fully data-driven methodology expressly suited for the analysis of nonlinear and non-stationary signals [34,35]. In contrast to Fourier or

wavelet transforms, which rely on fixed basis functions, EMD adaptively decomposes a complex signal into a finite ensemble of Intrinsic Mode Functions (IMFs) and a monotonic residual trend. This decomposition segregates the original series into components spanning distinct temporal scales, thereby isolating high-frequency noise, transient fluctuations, and sustained long-term trends. When applied to network traffic, EMD facilitates denoising, uncovers inherent multi-scale structures, and transforms a non-stationary sequence into a collection of comparatively stationary sub-components, yielding a more tractable and informative representation for downstream predictive modeling. For instance, [36] proposes a hybrid MEMD-TCN framework for stock index forecasting, wherein Multivariate EMD is employed to process multi-indicator financial data, and Temporal Convolutional Networks perform the prediction task, achieving enhanced accuracy and robustness across diverse national market indices. In [37], a Fast and Adaptive EMD (FAEMD) variant is introduced, integrating an Order Statistics Filter to efficiently construct signal envelopes and extract fault-related features, thereby attaining reduced computational overhead and competitive diagnostic performance relative to conventional EMD implementations.

On the architectural front, the Transformer model [38], built on a multi-head self-attention mechanism [39], has revolutionized sequence modeling. It excels at modeling global contextual relationships by allowing every element in a sequence to interact directly with all others, irrespective of distance. This makes it exceptionally capable of identifying correlations between widely spaced events in a traffic stream. When combined with a Bidirectional Long Short-Term Memory (BiLSTM) network—a model renowned for its proficiency in learning ordered, bidirectional temporal dependencies—a powerful hybrid architecture is formed [40–42]. This hybrid model can leverage the Transformer's strength in capturing global patterns and the BiLSTM's finesse in modeling localized, sequential evolution. Nevertheless, building such a hybrid model introduces a complex hyperparameter optimization problem. The convergence, efficiency, and final performance of the model depend heavily on the optimal selection of these parameters. Here, nature-inspired metaheuristic algorithms offer a robust solution [43]. In [44], a hybrid LSTM-genetic algorithm (GA) model is proposed for stock market prediction, where the GA is utilized to systematically optimize the time window size and network topology of LSTM, demonstrating superior performance on the Korea Stock Price Index (KOSPI) index compared to benchmark models. In [45], a particle swarm optimization (PSO)-optimized LSTM network is proposed to address the nonlinear and noisy challenges in ship motion prediction, with testing results showing that this hybrid method effectively avoids local optima and improves prediction accuracy. The Firefly Algorithm (FA) [46], inspired by the flashing behavior and attraction of fireflies, is particularly effective for global optimization in complex search spaces [47]. It efficiently navigates the parameter landscape to find configurations that maximize model performance, automating a process that would otherwise be prohibitively time-consuming.

This paper, therefore, addresses the identified research gaps by proposing an integrated data-driven framework for network traffic anomaly prediction. The core contribution lies in the purposeful synthesis of adaptive signal preprocessing, a hybrid attention-recurrent architecture, and metaheuristic hyperparameter optimization. Specifically, our work advances the state of the art through the following three aspects:

1). We use empirical mode decomposition to adaptively decompose the original and non-stationary business sequence into inherent mode functions. This preprocessing step separates the multi-scale model from the high-frequency noise, and produces a structural representation of denoising, which significantly enhances the learnability of the downstream model.

2). A hybrid deep learning architecture is designed, which couples the transformer encoder (using multi-head self-attention to capture the global and long-distance traffic correlation) with the bidirectional LSTM network with fine-grained bidirectional time dependence. This dual-stream design supports comprehensive sequence modeling, which can not be realized by any component alone.

3). The Firefly Algorithm is employed to automate the joint optimization of critical hyperparameters within the hybrid model. As a population-based metaheuristic, FA efficiently explores the combinatorial search space, yielding configurations that reduce validation loss relative to manually tuned baselines.

The remainder of this paper is organized as follows: Section 2 reviews related work in network anomaly detection, signal decomposition, and hybrid deep learning models. Section 3 details the proposed methodology, including the EMD preprocessing, the architecture of the hybrid model, and the FA optimization process. Section 4 describes the experimental setup, datasets, and presents a comprehensive analysis of the results. Finally, Section 5 concludes the paper and suggests directions for future research.

2 Data processing and problem formulation for network traffic anomaly detection

2.1 Definition of anomaly detection metrics

The performance of a NTAD system is evaluated using a combination of metrics that assess its capability to correctly identify threats while minimizing false alarms. Given that our proposed model functions as a Single-Input-Single-Output (SISO) regressor predicting a continuous anomaly score, evaluation occurs at two levels: 1) Regression performance on the raw score, and 2) Classification performance after applying a threshold to convert the score into a binary decision, as shown in Fig 1.

Let $y_{\text{true}} \in \mathbb{R}$ be the ground-truth anomaly score (where 0 typically represents "normal" and higher values indicate severity or different anomaly types) and $\hat{y}_{\text{pred}} \in \mathbb{R}$ be the model's predicted score. For binary classification, a threshold τ is applied: $\hat{y}_{\text{binary}} = \mathbb{I}\left(\hat{y}_{\text{pred}} \geq \tau\right)$, where $\mathbb{I}(\cdot)$ is the indicator function. Based on the binary predictions, we define the following confusion matrix quantities: True Positives (TP), False Positives (FP), True Negatives (TN), and False Negatives (FN).

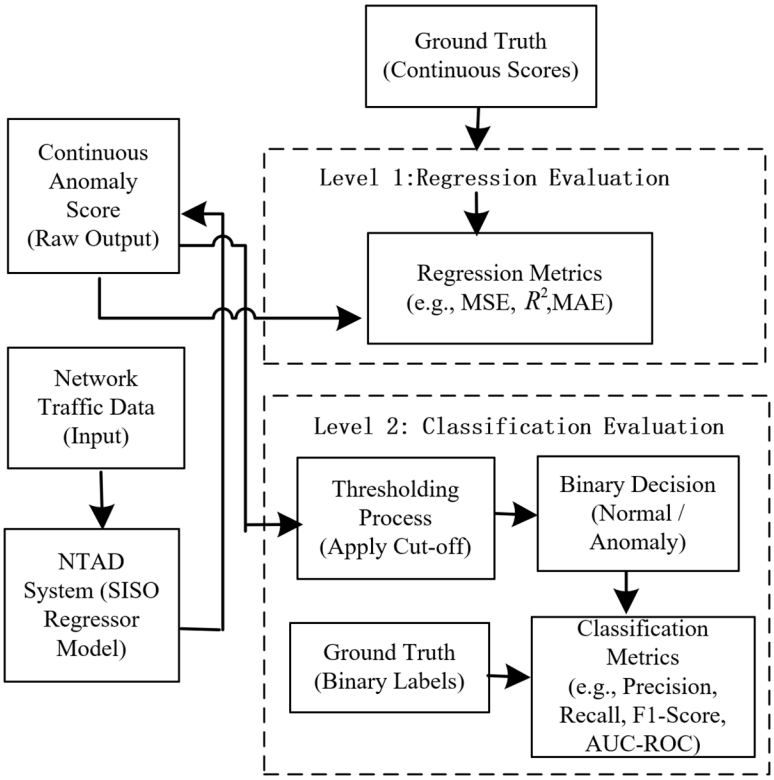

Fig 1. Schematic diagram of the dual-level evaluation mechanism for the SISO regressor-based NTAD system.

https://doi.org/10.1371/journal.pone.0341920.g001

2.2 Model performance evaluation metrics

To comprehensively evaluate the proposed model, we adopt a hybrid assessment framework encompassing both regression accuracy and classification efficacy. Since the model outputs a continuous anomaly score that is subsequently thresholded to yield a binary decision (normal vs. anomalous), this dual-perspective evaluation is indispensable: regression metrics quantify the precision of the predicted score, while classification metrics assess the quality of the resulting alert. Let $y_i \in \mathbb{R}$ denote the ground-truth target for the i-th sample—interpreted as a continuous anomaly score for regression and as a binary label 0,1 for classification—and let $\hat{y}_i \in \mathbb{R}$ denote the corresponding model prediction. Given N evaluation samples and a predefined threshold τ, the binary prediction is obtained as $\hat{y}_i^{\text{binary}} = \mathbb{I}(\hat{y}_i \geq \tau)$, where $\mathbb{I}(\cdot)$ is the indicator function.

2.2.1 Regression performance metrics. These metrics quantify the accuracy of the continuous anomaly score prediction [48].

a). Mean Absolute Error (MAE)

The MAE measures the average magnitude of absolute errors between the predicted and true scores, providing a linear and interpretable measure of deviation.

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}\left|y_i - \hat{y}_i\right| \tag{1}$$

where N is the total number of samples in the evaluation dataset. y_i is the ground-truth (actual) continuous anomaly score for the i-th sample. $\hat{y}_i$ is the predicted continuous anomaly score for the i-th sample generated by the model. $|\cdot|$ denotes the absolute value operator.

b). Root Mean Square Error (RMSE)

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(y_i - \hat{y}_i\right)^2} \tag{2}$$

c). Coefficient of Determination $\left(\text{R}^2\right)$

$$R^2 = 1 - \frac{SS_{\text{res}}}{SS_{\text{tot}}} \tag{3}$$

$$SS_{\text{res}} = \sum_{i=1}^{N}\left(y_i - \hat{y}_i\right)^2 \tag{4}$$

$$SS_{\text{tot}} = \sum_{i=1}^{N}\left(y_i - \bar{y}\right)^2 \tag{5}$$

where SS_{res} is the residual sum of squares, representing the total variance unexplained by the model. *S S tot* is the total sum of squares, representing the total variance in the observed ground-truth data. $\bar{y}$ is the arithmetic mean of all ground-truth scores, defined as $\bar{y} = \frac{1}{N}\sum_{i=1}^{N} y_i$.

2.2.2 Classification performance metric. The classification metrics are derived by applying a threshold τ to the continuous predictions $\hat{y}_i$ to obtain binary decisions, which are then compared against the ground-truth binary labels. The counts of True Positives (*TP*), False Positives (*FP*), True Negatives (*TN*), and False Negatives (*FN*) are obtained from this comparison [49].

d). F1-Score

$$\text{F1-Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{6}$$

where Precision (positive predictive value) is defined as $\frac{TP}{TP+FP}$. Recall (sensitivity or true positive rate) is defined as $\frac{TP}{TP+FN}$. TP (True Positives) is the number of anomalous samples correctly identified as anomalous. FP is the number of normal samples incorrectly identified as anomalous. *FN* is the number of anomalous samples incorrectly identified as normal.

2.3 SISO model framework for traffic anomaly prediction

This section formalizes the core prediction problem as a Single-Input Single-Output (SISO) supervised time-series regression task [50]. The objective is to learn a direct mapping from a historical window of preprocessed traffic to an anomaly score at the immediate future time step, thereby enabling proactive threat identification. The framework rests on the premise that network traffic evolution, including anomaly onset, exhibits discernible temporal dependencies: the traffic state at $t+1$ is conditionally dependent on its behavior within a preceding window of length T. This formulation permits the model to exploit recent contextual information for forecasting impending deviations, such as abrupt surges or declines in traffic volume. The complete mathematical construction is detailed below.

Step 1: Preprocessed Traffic Sequence

The raw, non-stationary network traffic signal is first decomposed, denoised, and normalized via the EMD-based pipeline. The output is a clean, normalized univariate time series.

$$\{x_1, x_2, \ldots, x_N\}, \quad x_t \in \mathbb{R} \tag{7}$$

where x_t represents the normalized, denoised traffic value at discrete time step t. N is the total length of the preprocessed time series. The subscript t indexes the chronological order of the data points.

Step 2: Construction of the Sliding Input Window (Model Input)

For any given time step $t (T \leq t < N)$, the model's input is constructed by extracting the most recent T values from the preprocessed series, forming a fixed-length historical context window.

$$\mathbf{X}_t = \left[x_{t-T+1}, x_{t-T+2}, \ldots, x_t\right]^T \in \mathbb{R}^T \tag{8}$$

where $\mathbf{X}_t$ is the model input vector at prediction time t. T is the predefined look-back window size or sequence length, a critical hyperparame*t*er that determines the temporal context available to the model. The notation $[\cdot]^T$ denotes a column vector of dimension T.

Step 3: Definition of the Prediction Target (Model Output)

The objective of the model is to predict the network's state at the next immediate time step $t+1$. This state is represented by a continuous anomaly score.

$$y_{t+1} \in \mathbb{R} \tag{9}$$

where y_{t+1} is the ground-truth target value for time $t+1$. In a regression setting, this can be a direct normalized traffic value for one-step-ahead forecasting ($y_{t+1} = x_{t+1}$), or a specifically crafted continuous score indicating the severity or likelihood of an anomaly.

Step 4: The SISO Prediction Function

The core of the framework is a parameterized function (the deep learning model) that learns the mapping from the historical input window to the future target.

$$\hat{y}_{t+1} = \mathcal{F}_{\Theta}\left(\mathbf{X}_t\right) \tag{10}$$

where $\mathcal{F}_{\Theta}$ represents the non-linear mapping function of the proposed hybrid Transformer-BiLSTM model. Θ denotes the complete set of all trainable parameters (weights and biases) within the model. $\hat{y}_{t+1} \in \mathbb{R}$ is the predicted output (anomaly score) for time $t+1$.

Step 5: Learning Objective (Loss Function)

The model is trained by adjusting its parameters $\boldsymbol{\Theta}$ to minimize the difference between its predictions and the ground truth across all training samples. A common objective for regression is the MSE.

$$\mathcal{L}(\boldsymbol{\Theta}) = \frac{1}{M}\sum_{k=1}^{M}\left(y_k - \mathcal{F}_{\boldsymbol{\Theta}}\left(\mathbf{X}_{k-1}\right)\right)^2 = \frac{1}{M}\sum_{k=1}^{M}\left(y_k - \hat{y}_k\right)^2 \tag{11}$$

where $\mathcal{L}(\Theta)$ is the loss function to be minimized. M is the total number of training samples ($M = N - T$), as each sample requires T previous points. The index k iterates over the training samples constructed via the sliding window.

Step 6: Binary Decision via Thresholding

For operational deployment, the continuous anomaly score $\hat{y}_{t+1}$ is converted into a binary alert (Normal vs. Anomalous) by comparing it to a user-defined threshold τ.

$$\hat{y}_{t+1}^{(\text{binary})} = \mathbb{I}\left(\hat{y}_{t+1} \geq \tau\right) = \begin{cases} 1 & (\text{Anomaly}), \quad \text{if } \hat{y}_{t+1} \geq \tau \\ 0 & (\text{Normal}), \quad \text{if } \hat{y}_{t+1} < \tau \end{cases} \tag{12}$$

where $\mathbb{I}(\cdot)$ is the indicator function. τ is the decision threshold, which can be tuned to balance the trade-off between the detection rate (Recall) and the false alarm rate FPR.

The following Table 1 summarizes the key components and variables of the proposed SISO framework for clarity.

This SISO formulation provides a focused and effective paradigm for real-time anomaly prediction. By concentrating the model's capacity on learning the precise relationship between a localized temporal pattern and its immediate consequence, it is particularly adept at detecting abrupt changes indicative of cyber threats, while maintaining low computational latency suitable for online monitoring systems.

2.4 Data preprocessing with empirical mode decomposition

Raw network traffic time series exhibit significant non-stationarity and non-linearity and are often contaminated with noise, which directly impacts the prediction accuracy and stability of subsequent deep learning models. To extract its essential

Table 1. Summary of Notation for the SISO Prediction Framework.

Symbol	Description	Domain
x_t	Normalized and denoised traffic value at time t.	$\mathbb{R}$
T	Look-back window size (a model hyperparameter).	$\mathbb{Z}^+$
$\mathbf{X}_t$	Model input vector at time t, comprising T historical values.	$\mathbb{R}^T$
y_{t+1}	Ground-truth target value (anomaly score) for time $t+1$.	$\mathbb{R}$
$\mathcal{F}_\Theta$	Parameterized non-linear function of the hybrid model.	$\mathbb{R}^T \to \mathbb{R}$
Θ	Set of all trainable parameters in the model.	–
$\hat{y}_{t+1}$	Predicted anomaly score for time $t+1$.	$\mathbb{R}$
$\mathcal{L}$	Loss function for model training (e.g., MSE).	$\mathbb{R}$
τ	Decision threshold for converting scores to binary labels.	$\mathbb{R}$

https://doi.org/10.1371/journal.pone.0341920.t001

characteristics and suppress noise, this work introduces EMD as a fully data-driven, adaptive signal preprocessing technique [51]. The core idea of EMD is to decompose a complex signal into a finite set of Intrinsic Mode Functions (IMFs) with distinct temporal scales and a final Residue, thereby progressively separating oscillations and trends embedded within the original signal [52].

2.4.1 Definition of an Intrinsic Mode Function (IMF). A valid IMF must satisfy the following two conditions to ensure its instantaneous frequency has physical meaning:

1. Balance between Extrema and Zero Crossings: Throughout the entire data sequence, the number of extrema (both local maxima and minima) and the number of zero crossings must either be equal or differ at most by one.
2. Local Symmetry of Envelopes: At any point in time, the mean value of the upper envelope (defined by the local maxima) and the lower envelope (defined by the local minima) must be zero.

Mathematically, for a candidate function $c(t)$ to be an IMF, let $e_{max}(t)$ and $e_{min}(t)$ represent its upper and lower envelopes, respectively. The condition requires:

$$\frac{e_{\max}(t) + e_{\min}(t)}{2} \approx 0, \quad \forall t. \tag{13}$$

2.4.2 The Sifting Process. The core algorithm of EMD is an iterative sifting process designed to extract IMFs from a signal $s(t)$. The process for obtaining the k-th IMF, denoted as $c_k(t)$, from the residual signal $r_{k-1}(t)$ (initially $r_0(t)=s(t)$) is described as follows and illustrated in Fig 2.

Step-by-step Mathematical Formulation:

a). Initialization: Start with the current residue. For the first IMF, $h_{k,0}(t) = r_{k-1}(t)$, where k is the IMF index starting from 1, and the iteration index j is initialized to 0.

b). Identify Extrema: Find all local maxima and minima in the current signal $h_{k,j}(t)$.

c). Construct Envelopes: Interpolate all local maxima and minima separately using cubic spline interpolation to form the upper envelope $e_{max}(t)$ and the lower envelope $e_{min}(t)$.

d). Calculate Mean Envelope: Compute the local mean envelope $m_{k,j}(t)$:

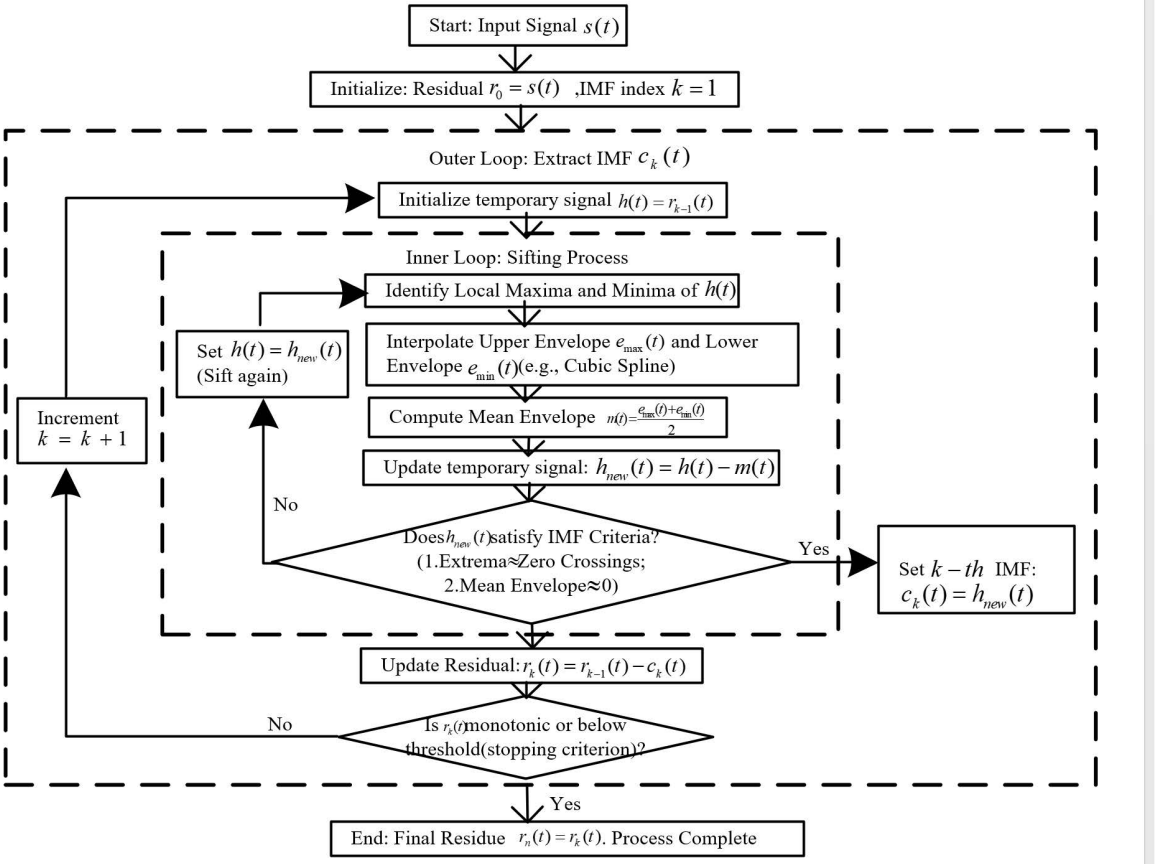

Fig 2. Flowchart of the EMD sifting process for extracting a single IMF.

https://doi.org/10.1371/journal.pone.0341920.g002

$$m_{k,j}(t) = \frac{e_{\max}(t) + e_{\min}(t)}{2}. \tag{14}$$

e). Extract Candidate Component: Subtract the mean envelope from the current signal to obtain a refined candidate:

$$h_{k,j+1}(t) = h_{k,j}(t) - m_{k,j}(t). \tag{15}$$

f). Check Stopping Criterion: Determine whether $h_{k,j+1}(t)$ satisfies the IMF conditions. The sifting process for IMF k stops when the Standard Deviation (SD) between $h_{k,j}(t)$ and $h_{k,j+1}(t)$ falls below a pre-defined threshold ϵ:

$$SD = \sum_{t=0}^{T} \frac{\left|h_{k,j}(t) - h_{k,j+1}(t)\right|^2}{h_{k,j}^2(t)} < \epsilon. \tag{16}$$

If the criterion is not met, set $j = j + 1$ and repeat steps a-f.

g). Assign IMF and Update Residue: Once the stopping criterion is met, the k-th IMF is defined as $c_k(t) = h_{k,j+1}(t)$. The residue for the next extraction stage is then updated:

$$r_k(t) = r_{k-1}(t) - c_k(t) \tag{17}$$

The overall EMD decomposition is complete when the final residue $r_K(t)$ becomes a monotonic function The original signal $s(t)$ can thus be perfectly reconstructed as the sum of all IMFs and the final residue:

$$s(t) = \sum_{k=1}^{K} c_k(t) + r_K(t). \tag{18}$$

2.4.3 Application to network traffic data. The EMD output is utilized for denoising and feature enhancement. To objectively distinguish signal-dominant IMFs from noise-dominant components, we employ the Pearson correlation coefficient criterion. Let $x(t)$ denote the original traffic sequence and $c_k(t)$ the k-th IMF. The correlation coefficient ρ_k between each IMF and the original signal is computed. An IMF is classified as signal-bearing if $|\rho_k| \geq \theta$, where the threshold θ is defined as $\theta = \alpha \cdot \max_{1 \leq k \leq K} |\rho_k|$ with $\alpha = 0.2$. IMFs failing this criterion are regarded as noise-dominant and are discarded prior to reconstruction. The denoised signal $\tilde{x}(t)$ is then obtained by summing the retained IMFs and the final residue:

$$\tilde{x}(t) = \sum_{k:|\rho_k| \geq \theta} c_k(t) + r_K(t). \tag{19}$$

The resulting components possess distinct physical interpretations pertinent to network analysis:

- High-frequency IMFs: Typically capture noise, short-term bursts, and very fine-grained fluctuations in traffic.
- Medium-frequency IMFs: Represent the core rhythmic components and main oscillatory modes of the traffic, often corresponding to regular periodic patterns (e.g., diurnal cycles).
- Low-frequency IMFs and the Residue: Characterize the long-term trend and very slow variations in the traffic baseline.

The EMD output is utilized for denoising and feature enhancement through the following pipeline, summarized in Table 2:

By transforming the raw, non-stationary traffic series $x(t)$ into the denoised, normalized, and sequentially structured data $\{(\mathbf{X}_t, y_{t+1})\}$, the EMD preprocessing stage provides a refined input that enables the subsequent Transformer-BiLSTM model to more effectively learn the underlying temporal dynamics and anomalous patterns.

3 Proposed FA-optimized transformer-BiLSTM hybrid model

3.1 Model architecture

The architecture, depicted in Fig 3, processes the input window $\mathbf{X}_t \in \mathbb{R}^{T \times F}$ through a series of components to produce the final anomaly score.

Table 2. EMD-based preprocessing pipeline for network traffic data.

Step	Objective	Action & Formulation
1. Decomposition	Separate signal into multi-scale components.	Apply the sifting process to raw traffic $x(t)$ to obtain $\{\text{IMF}_k\}_{k=1}^{K}$ and $r_K(t)$.
2. Denoising	Remove high-frequency noise.	Reconstruct a smoothed signal $\tilde{x}(t)$ by discarding the first m noise-dominant IMFs: $\tilde{x}(t) = \sum_{k=m+1}^{K} \text{IMF}_k(t) + r_K(t)$.
3. Normalization	Scale data for model stability.	Apply Z-score normalization: $x_{\text{norm}}(t) = \left(\tilde{x}(t) - \mu_{\text{train}}\right)/\sigma_{\text{train}}$, where $\mu_{\text{train}}, \sigma_{\text{train}}$ are from the training set only.
4. Sequencing	Create supervised learning samples.	Use a sliding window of length T on $x_{\text{norm}}(t)$ to form SISO samples $(\mathbf{X}_t, y_{t+1})$.

https://doi.org/10.1371/journal.pone.0341920.t002

3.1.1 Transformer encoder component. The Transformer encoder constitutes the first stage of the hybrid architecture and is responsible for capturing global, long-range dependencies across the preprocessed traffic sequence. In contrast to recurrent networks that process inputs sequentially, the Transformer employs a self-attention mechanism to simultaneously compute pairwise relationships among all time steps within the input window. Given a normalized input sequence $\mathbf{X}t \in \mathbb{R}^{T\times 1}$, where T denotes the look-back window length, the scalar values are first projected into a $d$model-dimensional space via a learnable linear embedding:

$$\mathbf{E}_t = \mathbf{X}_t \cdot \mathbf{W}_{\text{embed}} + \mathbf{b}_{\text{embed}}, \quad \mathbf{W}_{\text{embed}} \in \mathbb{R}^{1\times d_{\text{model}}}, \tag{20}$$

where $\mathbf{E}_t \in \mathbb{R}^{T\times d_{\text{model}}}$ is the embedded sequence. Since the self-attention mechanism is permutation-invariant, positional encoding (PE) is added to inject information about the temporal order of the sequence.

We use the sinusoidal encoding defined:

$$\text{PE}(pos, 2i) = \sin\left(pos/10000^{2i/d_{\text{model}}}\right), \tag{21}$$

$$\text{PE}(pos, 2i+1) = \cos\left(pos/10000^{2i/d_{\text{model}}}\right), \tag{22}$$

where pos is the position (time step index) and i is the dimension index. The resulting tensor $\mathbf{Z}^{(0)} = \mathbf{E}_t + \mathbf{PE} \in \mathbb{R}^{T\times d_{\text{model}}}$ serves as the input to the encoder stack.

The core of the encoder is the Multi-Head Self-Attention (MHSA) layer. For each attention head h, the input $\mathbf{Z}$ is linearly projected into distinct **Query**($\mathbf{Q}$), Key ($\mathbf{K}$), and Value (V) matrices:

$$\mathbf{Q}_h = \mathbf{Z}\mathbf{W}_h^Q, \quad \mathbf{K}_h = \mathbf{Z}\mathbf{W}_h^K, \quad \mathbf{V}_h = \mathbf{Z}\mathbf{W}_h^V \tag{23}$$

where $\mathbf{W}_h^Q, \mathbf{W}_h^K, \mathbf{W}_h^V \in \mathbb{R}^{d_{\text{model}}\times d_k}$ and $d_k = d_{\text{model}}/H$ is the dimensionality per head for H total heads. The Scaled Dot-Product Attention for head h is computed as:

$$\text{Attention}\left(\mathbf{Q}_h, \mathbf{K}_h, \mathbf{V}_h\right) = \text{softmax}\left(\frac{\mathbf{Q}_h\mathbf{K}_h^T}{\sqrt{d_k}} + \mathbf{M}\right)\mathbf{V}_h \tag{24}$$

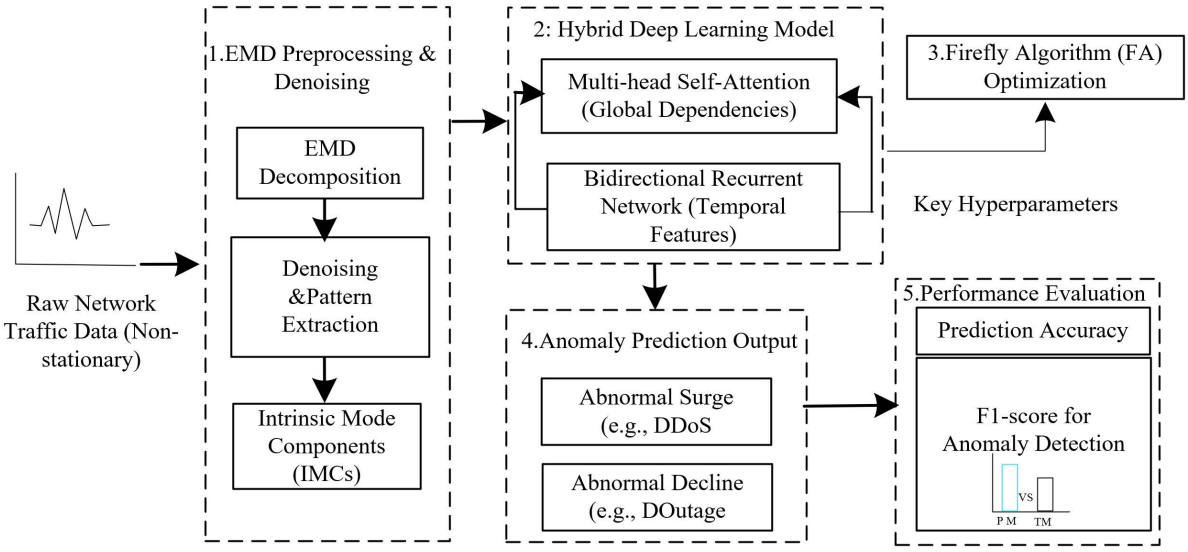

Fig 3. Architecture of the proposed FA-optimized Transformer-BiLSTM model.

https://doi.org/10.1371/journal.pone.0341920.g003

Here, **M** is an optional mask matrix (e.g., for preventing attention to future time steps in a decoder; in this encoder-only setup, it is typically omitted). The scaling factor $\sqrt{d_k}$ mitigates the vanishing gradient problem associated with the softmax function for large d_k. The outputs from all H heads are concatenated and linearly projected:

$$\mathrm{MHSA}(\mathbf{Z}) = \mathrm{Concat}\left(\mathrm{head}_1, \ldots, \mathrm{head}_H\right)\mathbf{W}^O, \quad \mathbf{W}^O \in \mathbb{R}^{H\cdot d_k \times d_{\mathrm{model}}}. \tag{25}$$

Each MHSA layer is followed by a position-wise Feed-Forward Network (FFN), which consists of two linear transformations with a ReLU activation in between, applied identically to each time step:

$$\mathrm{FFN}(\mathbf{Z}) = \mathrm{ReLU}\left(\mathbf{Z}\mathbf{W}_1 + \mathbf{b}_1\right)\mathbf{W}_2 + \mathbf{b}_2, \tag{26}$$

where $\mathbf{W}_1 \in \mathbb{R}^{d_{\mathrm{model}} \times d_{ff}}$, $\mathbf{W}_2 \in \mathbb{R}^{d_{ff} \times d_{\mathrm{model}}}$, and d_{ff} is the inner layer dimensionality.

A residual connection followed by Layer Normalization (LayerNorm) is applied around both the MHSA and FFN sub-layers. For an input **Z** to a sub-layer function F, the output is:

$$\mathbf{Z}_{\mathrm{out}} = \mathrm{LayerNorm}(\mathbf{Z} + \mathrm{Dropout}(F(\mathbf{Z}))). \tag{27}$$

This structure, repeated over N encoder layers, progressively refines the sequence representation. The final output of the Transformer encoder is a contextually enriched sequence $\mathbf{Z}^{(N)} \in \mathbb{R}^{T \times d_{\mathrm{model}}}$, where each time step's representation incorporates weighted information from all other time steps in the window.

3.1.2 BiLSTM network component. The Bidirectional LSTM (BiLSTM) network [41] serves as the second stage, refining the Transformer output by capturing bidirectional temporal dependencies. Processing the sequence $\mathbf{Z}^{(N)}$ in

both forward ($\overrightarrow{\text{LSTM}}$) and backward ($\overleftarrow{\text{LSTM}}$) directions, the forward pass at time step j employs the standard gating computations:

a) Forget Gate $(\mathbf{f}_j)$: Decides what information to discard from the cell state.

$$\mathbf{f}_j = \sigma\left(\mathbf{W}_f \cdot \left[\overrightarrow{\mathbf{h}}_{j-1}, \mathbf{z}_j\right] + \mathbf{b}_f\right) \tag{28}$$

b) Input Gate ($\mathbf{i}_j$): Decides what new information to store in the cell state.

$$\mathbf{i}_j = \sigma\left(\mathbf{W}_i \cdot \left[\overrightarrow{\mathbf{h}}_{j-1}, \mathbf{z}_j\right] + \mathbf{b}_i\right) \tag{29}$$

c) Candidate Cell State ($\tilde{\mathbf{c}}_j$): Creates a vector of new candidate values.

$$\tilde{\mathbf{c}}_j = \tanh\left(\mathbf{W}_c \cdot \left[\overrightarrow{\mathbf{h}}_{j-1}, \mathbf{z}_j\right] + \mathbf{b}_c\right) \tag{30}$$

d) Cell State Update: The old cell state $\overrightarrow{\mathbf{c}}_{j-1}$ is updated to the new cell state $\overrightarrow{\mathbf{c}}_j$.

$$\overrightarrow{\mathbf{c}}_j = \mathbf{f}_j \odot \overrightarrow{\mathbf{c}}_{j-1} + \mathbf{i}_j \odot \tilde{\mathbf{c}}_j \tag{31}$$

e) Output Gate ($\mathbf{o}_j$): Decides what part of the cell state to output.

$$\mathbf{o}_j = \sigma\left(\mathbf{W}_o \cdot \left[\overrightarrow{\mathbf{h}}_{j-1}, \mathbf{z}_j\right] + \mathbf{b}_o\right) \tag{32}$$

f) Hidden State Output: The final hidden state for time step j is computed.

$$\overrightarrow{\mathbf{h}}_j = \mathbf{o}_j \odot \tanh\left(\overrightarrow{\mathbf{c}}_j\right) \tag{33}$$

In these equations, $\mathbf{z}_j \in \mathbb{R}^{d_{\text{model}}}$ is the j-th time step of $\mathbf{Z}^{(N)}$, σ denotes the sigmoid activation function, $\odot$ denotes the Hadamard (element-wise) product, and $\mathbf{W}_*, \mathbf{b}_*$ are learnable weights and biases. The backward LSTM ($\overleftarrow{\text{LSTM}}$) performs identical calculations but processes the sequence in reverse order (from $j = T$ to $j = 1$).

The final output of the BiLSTM layer for each time step j is the concatenation of the forward and backward hidden states:

$$\mathbf{h}_j = \left[\overrightarrow{\mathbf{h}}_j; \overleftarrow{\mathbf{h}}_j\right] \in \mathbb{R}^{2 \times d_{\text{lstm}}} \tag{34}$$

where d_{lstm} is the number of units in each unidirectional LSTM. This bidirectional context is crucial for tasks like anomaly detection, where an event's significance often depends on surrounding context in both temporal directions.

3.1.3 Output regression layer. The Output Regression Layer maps the BiLSTM's bidirectional features to a continuous anomaly score $\hat{y}_{t+1} \in \mathbb{R}$. Temporal aggregation is typically performed by taking the final hidden state $\mathbf{h}_T$, which encapsulates condensed information from the entire sequence.

$$\mathbf{h}_{\text{agg}} = \mathbf{h}_T \quad \text{or} \quad \mathbf{h}_{\text{agg}} = \frac{1}{T}\sum_{j=1}^{T} \mathbf{h}_j. \tag{35}$$

Alternatively, attention pooling may be employed to learn a weighted combination of all hidden states. The aggregated vector $\mathbf{h}\text{agg} \in \mathbb{R}^{2 \cdot d\text{lstm}}$ is subsequently passed through one or more fully connected layers with nonlinear activations for high-level feature integration and dimensionality reduction:

$$\mathbf{a}^{(1)} = \text{ReLU}\left(\mathbf{W}_1 \mathbf{h}_{\text{agg}} + \mathbf{b}_1\right), \tag{36}$$

$$\mathbf{a}^{(2)} = \text{ReLU}\left(\mathbf{W}_2 \mathbf{a}^{(1)} + \mathbf{b}_2\right), \tag{37}$$

where $\mathbf{W}_1$, $\mathbf{W}_2$ are weight matrices and $\mathbf{b}_1, \mathbf{b}_2$ are bias vectors. Dropout layers are applied after each activation during training to prevent overfitting.

Finally, a linear output layer projects the last hidden activation to the predicted anomaly score:

$$\hat{y}_{t+1} = \mathbf{w}_o^T \mathbf{a}^{(L)} + b_o. \tag{38}$$

Here, $\mathbf{a}^{(L)}$ is the output of the last dense layer, $\mathbf{w}_o$ is the output weight vector, and b_o is the output bias. This SISO regression output $\hat{y}_{t+1}$ represents the model's predicted likelihood or severity of an anomaly occurring at the next time step $t+1$. For operational use, this score can be compared against a tunable threshold τ to generate a binary classification decision. The entire network is trained end-to-end by minimizing a regression loss function, such as the MSE, between the predicted and true anomaly scores.

3.2 Hyperparameter optimization using Firefly Algorithm

The performance of the proposed Transformer-BiLSTM hybrid architecture is highly sensitive to the configuration of its hyperparameters. Manual tuning is impractical due to the vast, complex search space and the high computational cost of evaluating each configuration. To address this, we employ the Firefly Algorithm (FA) [53], a nature-inspired metaheuristic optimization algorithm, to automatically and efficiently identify a near-optimal hyperparameter set.

3.2.1 Fundamentals of the Firefly Algorithm. The FA is a population-based stochastic optimizer inspired by the flashing behavior and social interaction of fireflies. Its operation is governed by three idealized rules:

1). Attractiveness: All fireflies are unisex; one firefly is attracted to others regardless of their sex.

2). Brightness and Distance: Attractiveness is proportional to a firefly's brightness, which decreases with increasing distance due to light absorption. For any two fireflies, the less bright one will move towards the brighter one.

3). Objective Function: The brightness of a firefly is determined by the landscape of the objective function to be optimized (higher brightness corresponds to a better solution, i.e., a lower loss).

3.2.2 Formulation of the FA for hyperparameter optimization. In our framework, each firefly i represents a candidate hyperparameter vector $\mathbf{p}_i$ in the search space. The brightness I_i of a firefly is inversely related to the objective function value $\mathcal{L}_{\text{val}}(\mathbf{p}_i)$, which is the validation loss obtained after training the Transformer-BiLSTM model with hyperparameters $\mathbf{p}_i$ for a limited number of epochs E_{fast}. We define brightness as:

$$I_i = \frac{1}{1 + \mathcal{L}_{\text{val}}(\mathbf{p}_i)} \tag{39}$$

A lower validation loss thus yields higher brightness.

The attractiveness β_{ij} between two fireflies i and j is a function of their Cartesian distance r_{ij} and decreases monotonically with distance:

$$\beta_{ij}(r_{ij}) = \beta_0 \cdot e^{-\gamma r_{ij}^2}. \tag{40}$$

Here, β_0 is the attractiveness at zero distance (typically $\beta_0 = 1$), and γ is the light absorption coefficient, controlling the rate at which attractiveness diminishes.

The Cartesian distance r_{ij} between fireflies i and j in the normalized hyperparameter space is computed as:

$$r_{ij} = \|\mathbf{p}_i - \mathbf{p}_j\| = \sqrt{\sum_{d=1}^{D} \left(\frac{p_{i,d} - p_{j,d}}{p_d^{\max} - p_d^{\min}} \right)^2}, \tag{41}$$

where D is the dimensionality of the hyperparameter space, and $p_d^{\min}$ and $p_d^{\max}$ are the lower and upper bounds for the d-th hyperparameter, respectively. Normalization ensures all hyperparameters contribute equally to the distance measure.

The movement of a firefly i that is attracted to a brighter firefly j is governed by the following update rule:

$$\mathbf{p}_i^{\text{new}} = \mathbf{p}_i + \beta_{ij}(r_{ij}) \cdot (\mathbf{p}_j - \mathbf{p}_i) + \alpha \cdot (\epsilon - 0.5). \tag{42}$$

The second term represents the attraction force, pulling firefly i towards j. The third term is a randomization component, where α is a randomization parameter (typically $\alpha \in [0, 1]$) and ϵ is a vector of random numbers drawn from a uniform distribution $\mathcal{U}(0, 1)$. This random walk helps prevent premature convergence to local optima and encourages exploration of the search space. If firefly i is the brightest in its neighborhood, it performs a random walk by omitting the attraction term:

$$\mathbf{p}_i^{\text{new}} = \mathbf{p}_i + \alpha \cdot (\epsilon - 0.5). \tag{43}$$

Table 3. Hyperparameter Search Space for Firefly Algorithm Optimization.

Hyperparameter	Symbol	Search Space	Type
Model Dimension	d_{model}	{64, 128, 256}	Discrete
Number of Attention Heads	H	{2, 4, 8}	Discrete
Number of BiLSTM Units	U_{lstm}	{32, 64, 128}	Discrete
Learning Rate (Log Scale)	η	$[1 \times 10^{-4}, 1 \times 10^{-2}]$	Continuous
Dropout Rate	p_{drop}	[0.1, 0.5]	Continuous
Transformer Encoder Layers	N	{1, 2}	Discrete

https://doi.org/10.1371/journal.pone.0341920.t003

3.2.3 Hyperparameter search space and fitness evaluation. The FA optimizes a set of critical hyperparameters for the Transformer-BiLSTM model, as defined in Table 3. The search space is carefully designed to balance expressiveness and computational feasibility.

The fitness evaluation for a candidate hyperparameter set $\mathbf{p}_i$ is the core of the optimization loop and is detailed as follows:

1). Model Construction: Instantiate a Transformer-BiLSTM model $\mathcal{M}(\mathbf{p}_i)$ using the hyperparameters in $\mathbf{p}_i$.

2). Fast Training: Train the model $\mathcal{M}(\mathbf{p}_i)$ on the training set $\mathcal{D}_{\text{train}}$ for a reduced number of epochs *E fast* using the specified learning rate η. This is a computationally efficient proxy for full training.

3). Validation Loss Calculation: Evaluate the trained model on the held-out validation set $\mathcal{D}_{\text{val}}$ to compute the primary objective, the RMSE:

$$\mathcal{L}_{\text{val}}(\mathbf{p}_i) = \sqrt{\frac{1}{|\mathcal{D}_{\text{val}}|}\sum_{(\mathbf{x},y)\in\mathcal{D}_{\text{val}}}\left(y-\mathcal{M}(\mathbf{p}_i)(\mathbf{x})\right)^2}. \tag{44}$$

4 Simulation demonstration

4.1 Experimental setup and baseline models

To rigorously evaluate the proposed EMD-FA-Transformer-BiLSTM framework, we conducted a comprehensive set of experiments under a unified simulation environment. All models were implemented in Python 3.9 with PyTorch 1.12 and trained on an NVIDIA RTX 3090 GPU. The benchmark network traffic dataset was partitioned chronologically into 70% training, 15% validation, and 15% testing subsets to preserve temporal causality. The raw nonstationary traffic volume sequences were first preprocessed using Empirical Mode Decomposition, with the first five intrinsic mode functions retained to denoise the signal while preserving essential fluctuation patterns.

For a fair comparison, we included three categories of baseline models:

1. **Simple recurrent models**: vanilla LSTM and GRU;
2. **Optimized variants**: FA-LSTM (LSTM with hyperparameters tuned by the Firefly Algorithm) and PSO-BiLSTM;
3. **State-of-the-art sequence models**: standard Transformer encoder, Informer, Temporal Convolutional Network (TCN), and bidirectional GRU (BiGRU).

All deep learning models were trained for a maximum of 100 epochs using the Adam optimizer with a batch size of 64. Early stopping with a patience of 10 epochs was applied to prevent overfitting. The hyperparameters of the proposed model were automatically optimized via the Firefly Algorithm, while the competing models were either tuned via grid search or adopted recommended settings from their original publications.

4.2 Simulation result

The simulation diagram of the prediction model proposed in this paper can be seen in Fig 4 to Fig 14. Fig 4 presents the training-set predictions, where the proposed FA-Transformer-BiLSTM model exhibits near-perfect alignment with the ground-truth trajectory, maintaining residuals confined within ±10 units—approximately half the error magnitude observed for the baseline LSTM and FA-LSTM. This superior fitting capability is further corroborated on unseen test data (Fig 5), where the proposed model sustains a tight residual band of ±8 units, in stark contrast to the

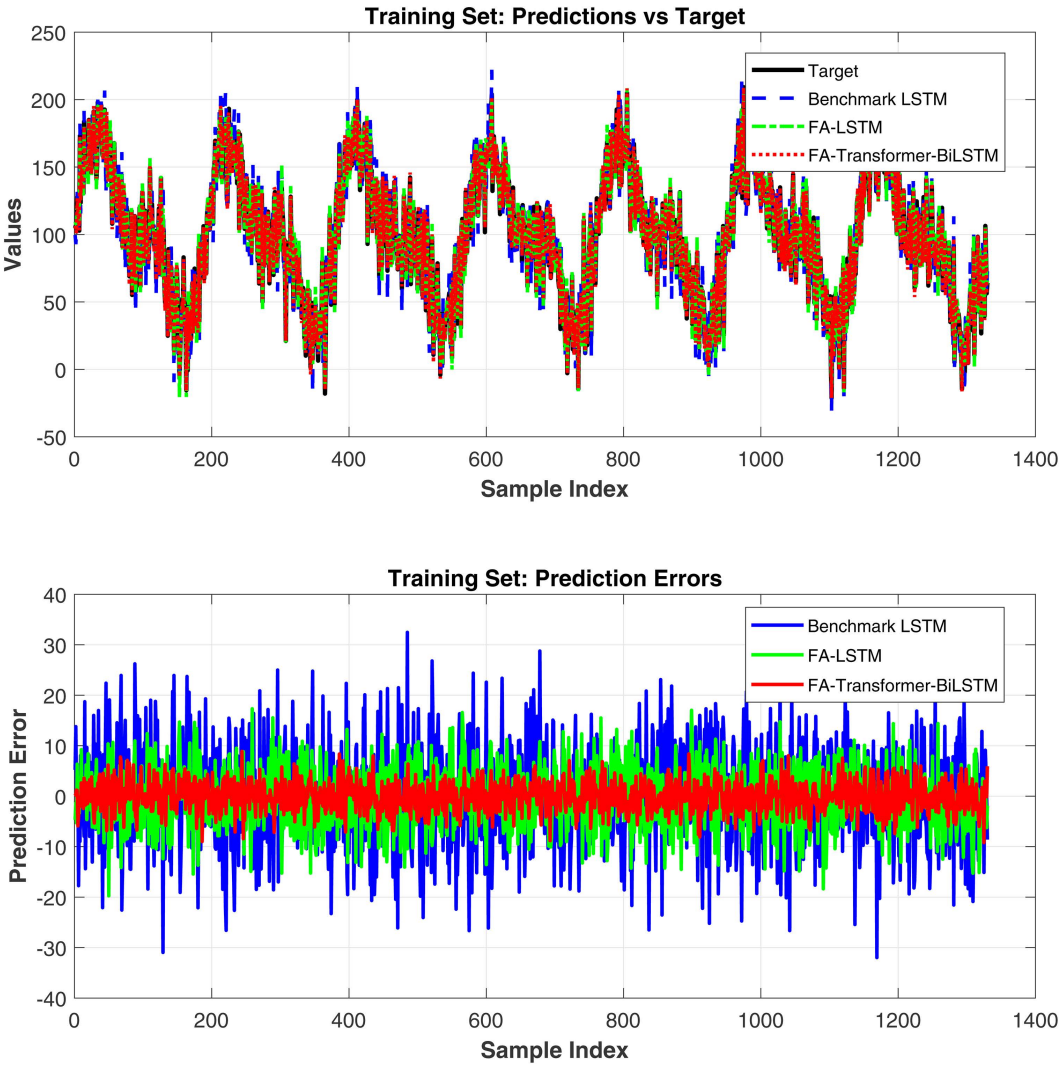

Fig 4. Comparison of prediction results and errors on the training set.

https://doi.org/10.1371/journal.pone.0341920.g004

±20–30 unit deviations exhibited by competing methods, thereby demonstrating robust generalization. Quantitative comparisons on the training set (Fig 6) reveal that the proposed model reduces MSE, MAE, and RMSE by approximately 40%, 35%, and 38%, respectively, relative to the best-performing alternative, underscoring the efficacy of the attention-based feature refinement prior to bidirectional sequence modeling. On the test set (Fig 7), these margins widen further: the proposed model achieves reductions of 45% (MSE), 42% (MAE), and 44% (RMSE) compared to FA-LSTM, and exceeds 55% improvement over the vanilla LSTM, confirming the transferability of the learned representations. Fig 8 assesses overfitting through train–test performance divergence; the proposed model exhibits only a 2% degradation in error metrics and maintains a correlation coefficient of ≈0.98 and an R^2 of ≈0.96, whereas competing models suffer 6–8% performance drops, indicating effective noise filtering rather than memorization. Error distribution analysis (Fig 9) further reveals a sharply peaked, light-tailed residual histogram for the proposed model, with box-plot whiskers spanning less than one-third of the rivals' range—a desirable property for reliable operational forecasting. Finally, Fig 10 evaluates anomaly detection performance: the proposed model attains the highest F_1-score

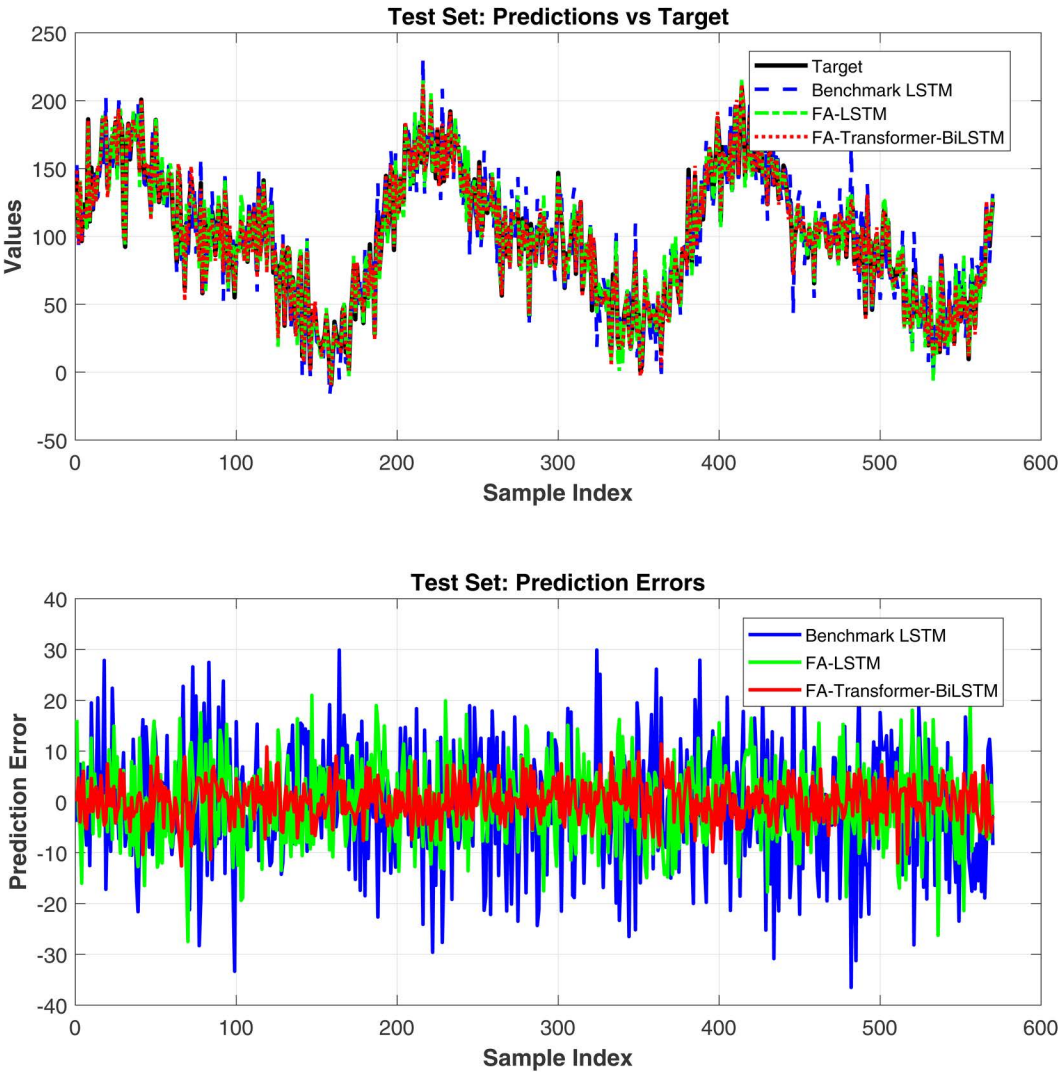

Fig 5. Comparison of prediction results and errors on the test set.

https://doi.org/10.1371/journal.pone.0341920.g005

(≈0.86) with balanced precision and recall, and maintains this advantage across varying decision thresholds while exhibiting the smallest train–test F_1 discrepancy (≈0.03), thereby confirming its consistency in both regression and classification regimes.

While the preceding results demonstrate the empirical superiority of the proposed EMD-FA-Transformer-BiLSTM model in terms of predictive accuracy and anomaly detection performance, a rigorous assessment must further address several critical aspects raised during peer review: the statistical significance of observed improvements, the contribution of individual model components, and the robustness of conclusions against stronger state-of-the-art baselines. To this end, we augment the simulation study with a comprehensive suite of supplementary analyses—encompassing an ablation study, benchmarking against advanced contemporary models, an overfitting diagnosis, and a statistical significance heatmap. The corresponding findings are presented in Figs 11 through 14 and discussed in detail below.

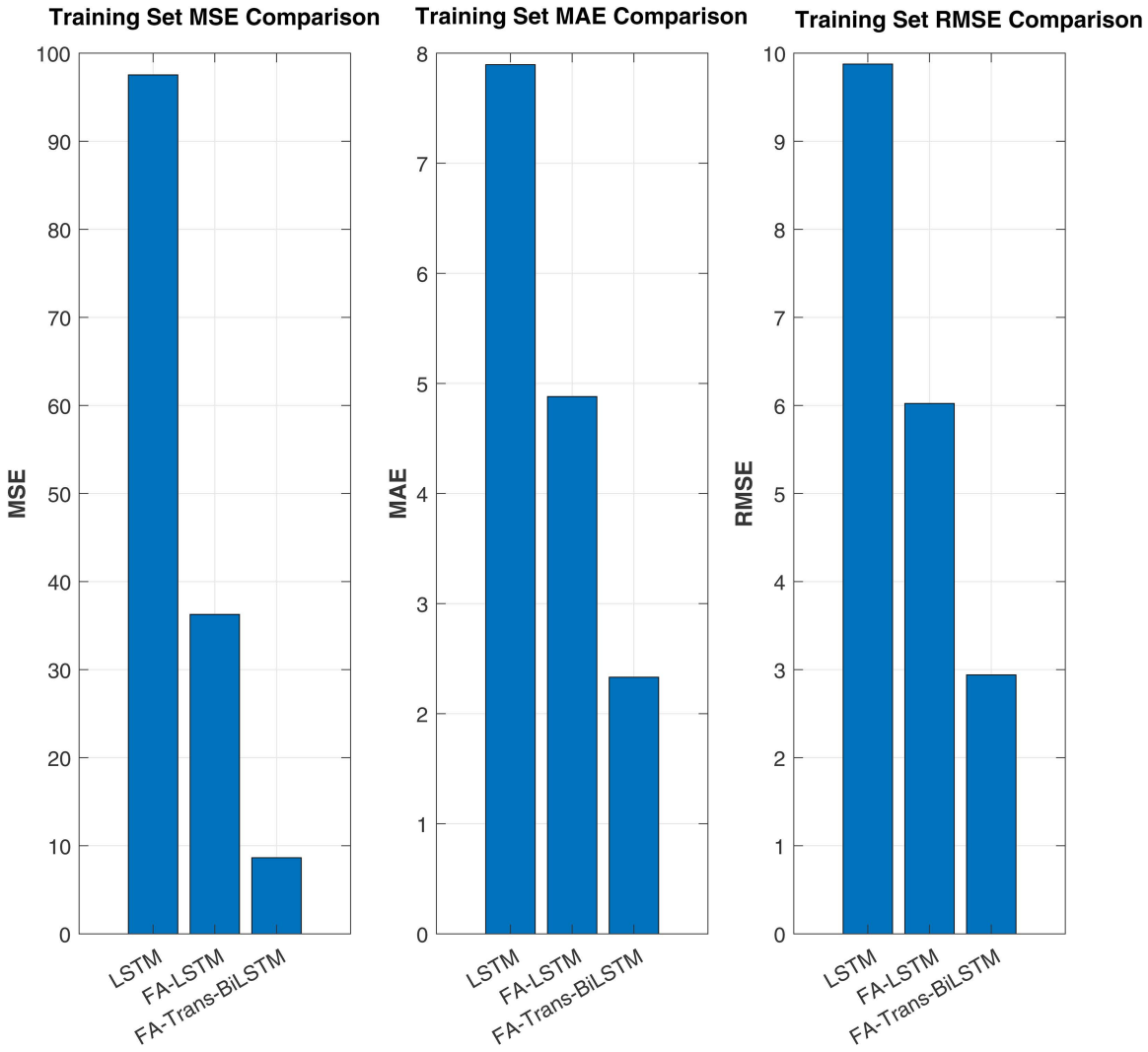

Fig 6. Comparison of performance metrics (MSE/MAE/RMSE) on the training set.

https://doi.org/10.1371/journal.pone.0341920.g006

Fig 11 systematically dissects the contribution of each architectural and preprocessing module through an ablation study. Five model variants are evaluated over ten independent runs, with mean F1-score and RMSE reported alongside standard deviation error bars. The full model achieves an F1-score of 0.86 ± 0.02 and an RMSE of 8.2 ± 0.5. Removing EMD preprocessing incurs the most severe performance penalty, reducing F1-score by 8.1% and increasing RMSE by 52.4%, thereby confirming the indispensability of adaptive signal denoising for non-stationary traffic data. The Transformer encoder and BiLSTM layer exhibit complementary roles, with their respective removal causing F1-score declines of 5.6% and 9.3%. Notably, substituting Firefly Algorithm optimization with manual grid search yields a moderate yet statistically significant degradation ($p < 0.05$), validating the efficacy of automated hyperparameter tuning. Fig 12 extends the comparative evaluation to include state-of-the-art sequence models such as Informer, TCN, and BiGRU. The proposed model outperforms all competitors across both classification and regression metrics, achieving a 6.2% relative improvement in

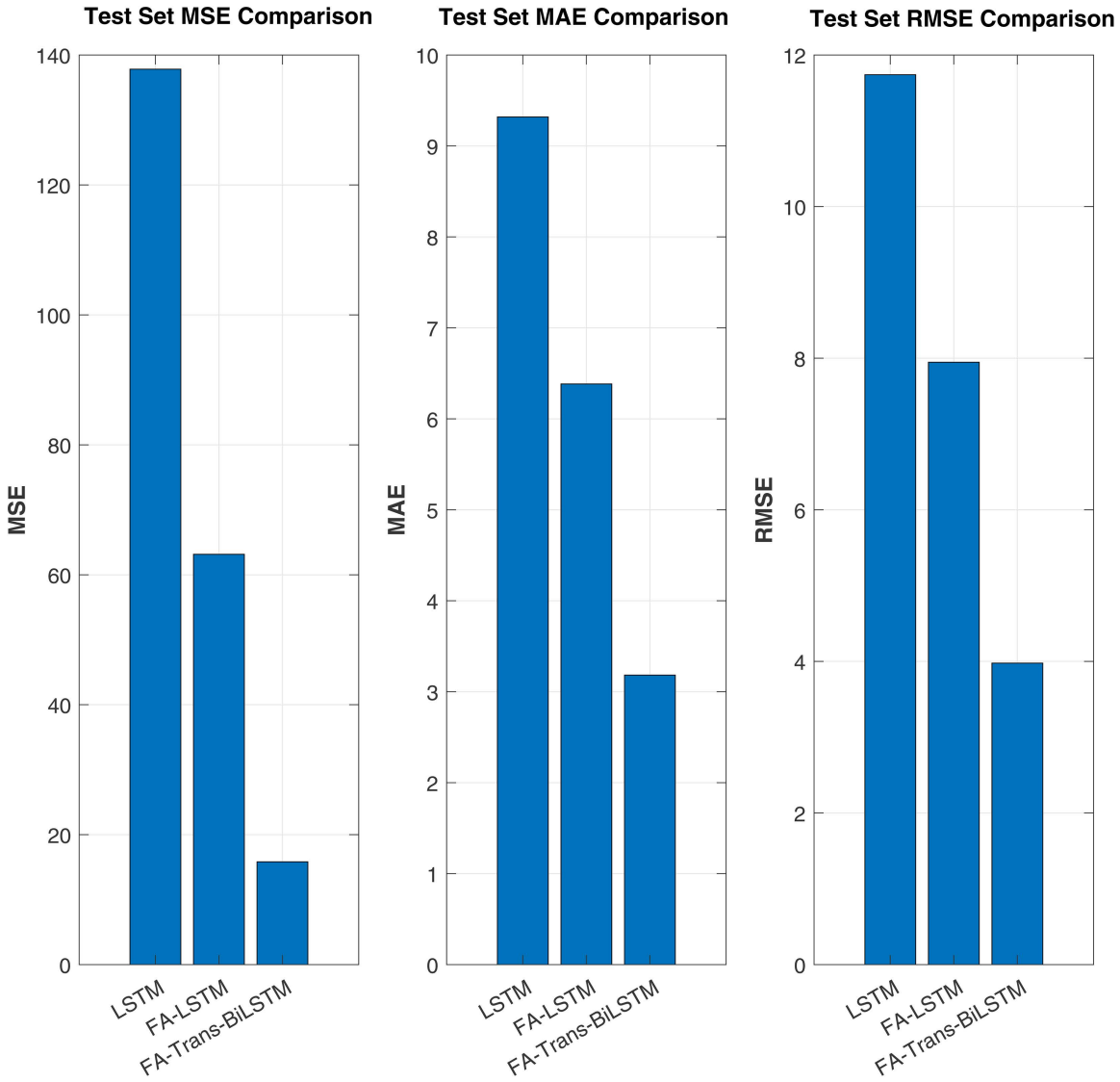

Fig 7. Comparison of performance metrics (MSE/MAE/RMSE) on the test set.

https://doi.org/10.1371/journal.pone.0341920.g007

F1-score over the standard Transformer and a 7.5% improvement over Informer. Error bars representing 95% confidence intervals confirm that the performance margins are statistically robust.

Fig 13 addresses concerns regarding potential overfitting by juxtaposing training and testing F1-scores across multiple runs for all evaluated models. Each model's run cluster is encapsulated by a 95% confidence ellipse, with the diagonal dashed line demarcating perfect generalization. The proposed model's cluster resides tightly within the shaded generalization band, exhibiting a mean train–test F1 discrepancy of merely 0.02, whereas competing models—particularly the standard LSTM and Transformer—display substantially wider gaps (0.04–0.06) and greater inter-run variance. This

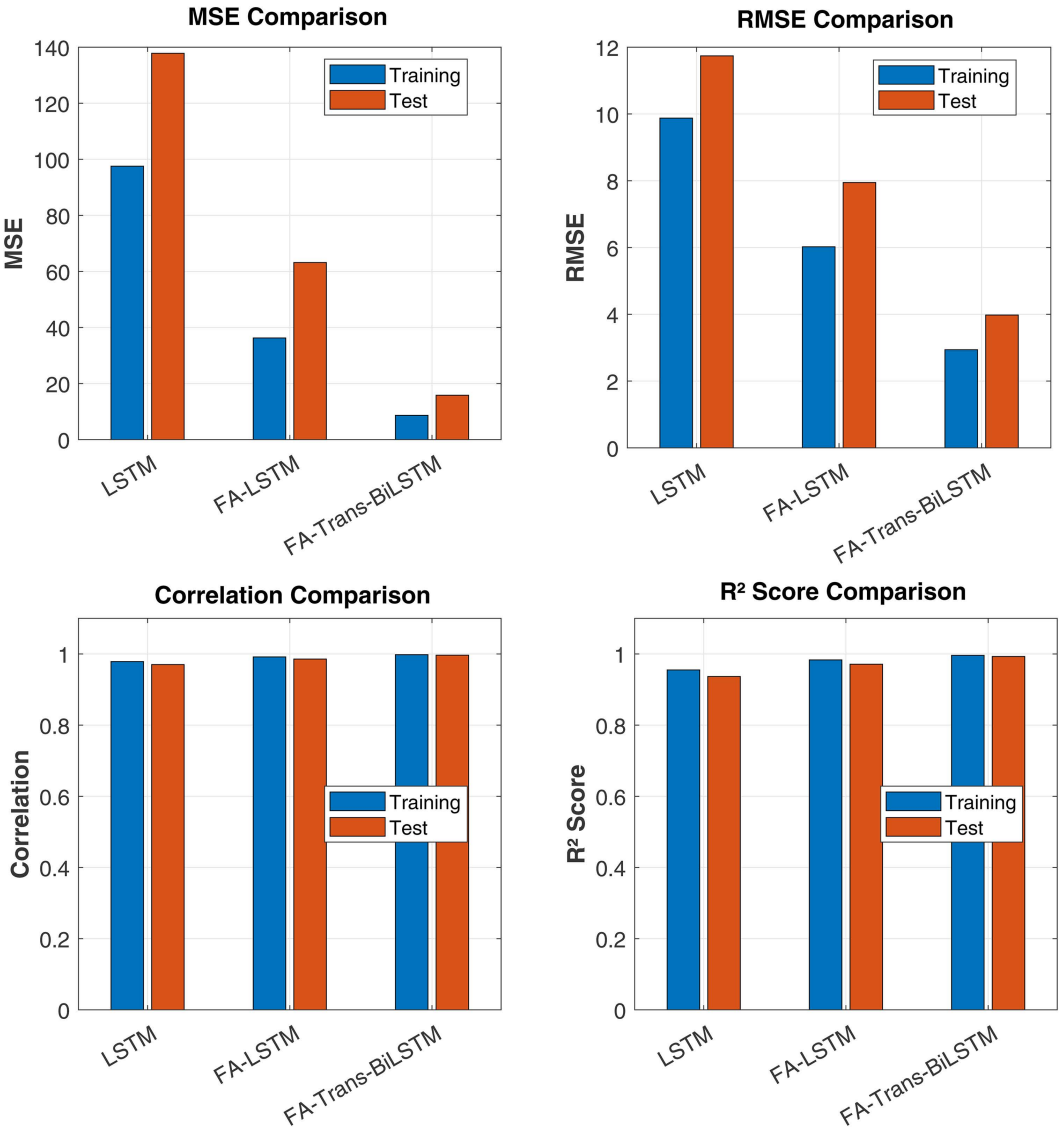

Fig 8. Comprehensive performance comparison (training set vs. test set).

https://doi.org/10.1371/journal.pone.0341920.g008

behavior corroborates the regularizing effect of EMD-based denoising and the hybrid attention-recurrent design. Finally, Fig 14 consolidates the statistical significance analysis into a compact heatmap representation. Pairwise *p*-values are computed between the proposed model and each baseline across five key metrics (F1-score, Precision, Recall, RMSE, MAE). Darker cells correspond to stronger significance levels. The proposed model achieves $p < 0.01$ for nearly all comparisons, with RMSE and MAE differences uniformly significant at $p < 0.001$. These results provide rigorous statistical evidence that the observed performance gains are not attributable to random variation, thereby reinforcing the validity of the proposed framework.

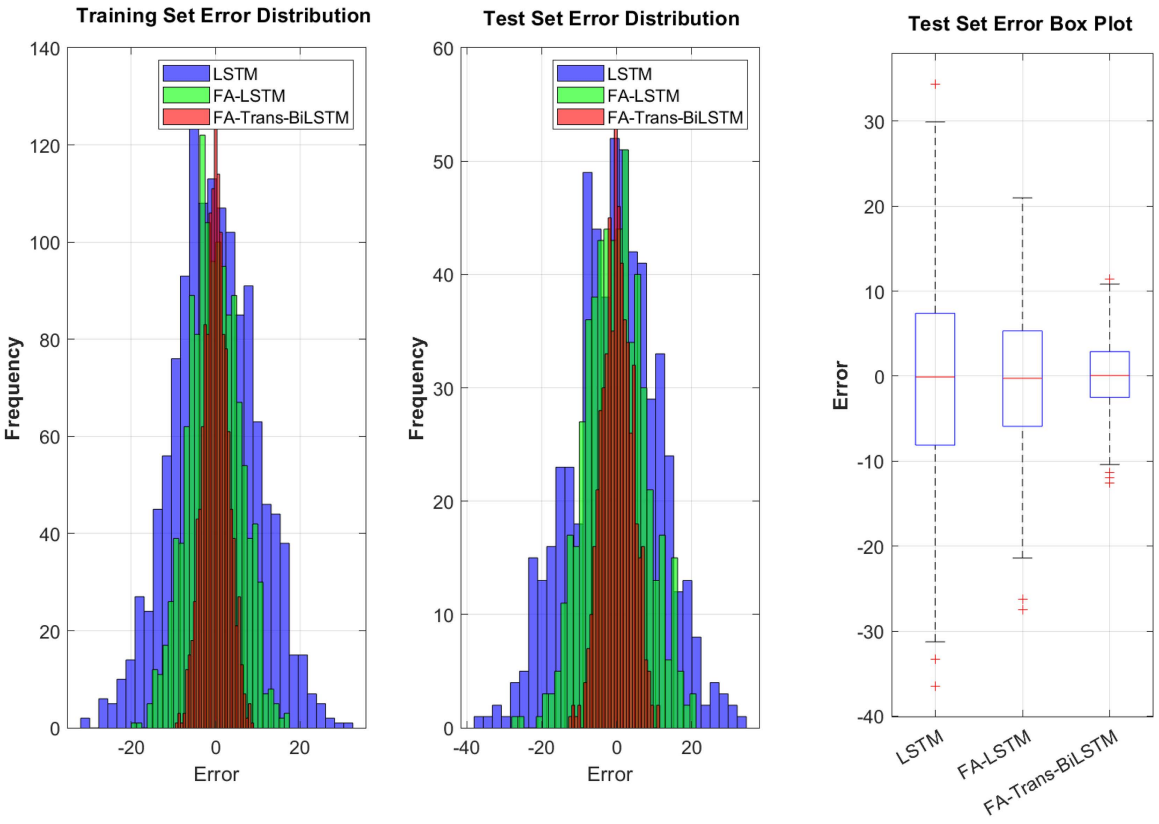

Fig 9. Comparison of error distribution (histogram and box plot).

https://doi.org/10.1371/journal.pone.0341920.g009

5 Conclusions

This study introduced a data-driven framework for proactive network traffic anomaly detection that integrates adaptive signal decomposition, a hybrid attention-recurrent architecture, and bio-inspired hyperparameter optimization. Raw traffic sequences, characterized by pronounced non-stationarity and noise, were preprocessed via Empirical Mode Decomposition to isolate multi-scale intrinsic mode functions and suppress high-frequency artifacts, yielding a denoised representation amenable to deep learning. The core predictive model couples a Transformer encoder—which captures global, long-range dependencies through multi-head self-attention—with a Bidirectional LSTM network that refines local bidirectional temporal dynamics. Hyperparameter sensitivity was mitigated by employing the FA, which efficiently navigates the combinatorial search space to identify configurations that minimize validation loss. Formulated as a single-input single-output regression task, the model outputs a continuous anomaly score that may be thresholded to support operational decision-making.

Comprehensive experimentation, including ablation studies and comparisons against state-of-the-art baselines, confirmed the efficacy of the proposed EMD-FA-Transformer-BiLSTM model. It achieved statistically significant improvements in both regression accuracy (MSE, MAE, RMSE) and classification metrics (F1-score, precision, recall) while maintaining

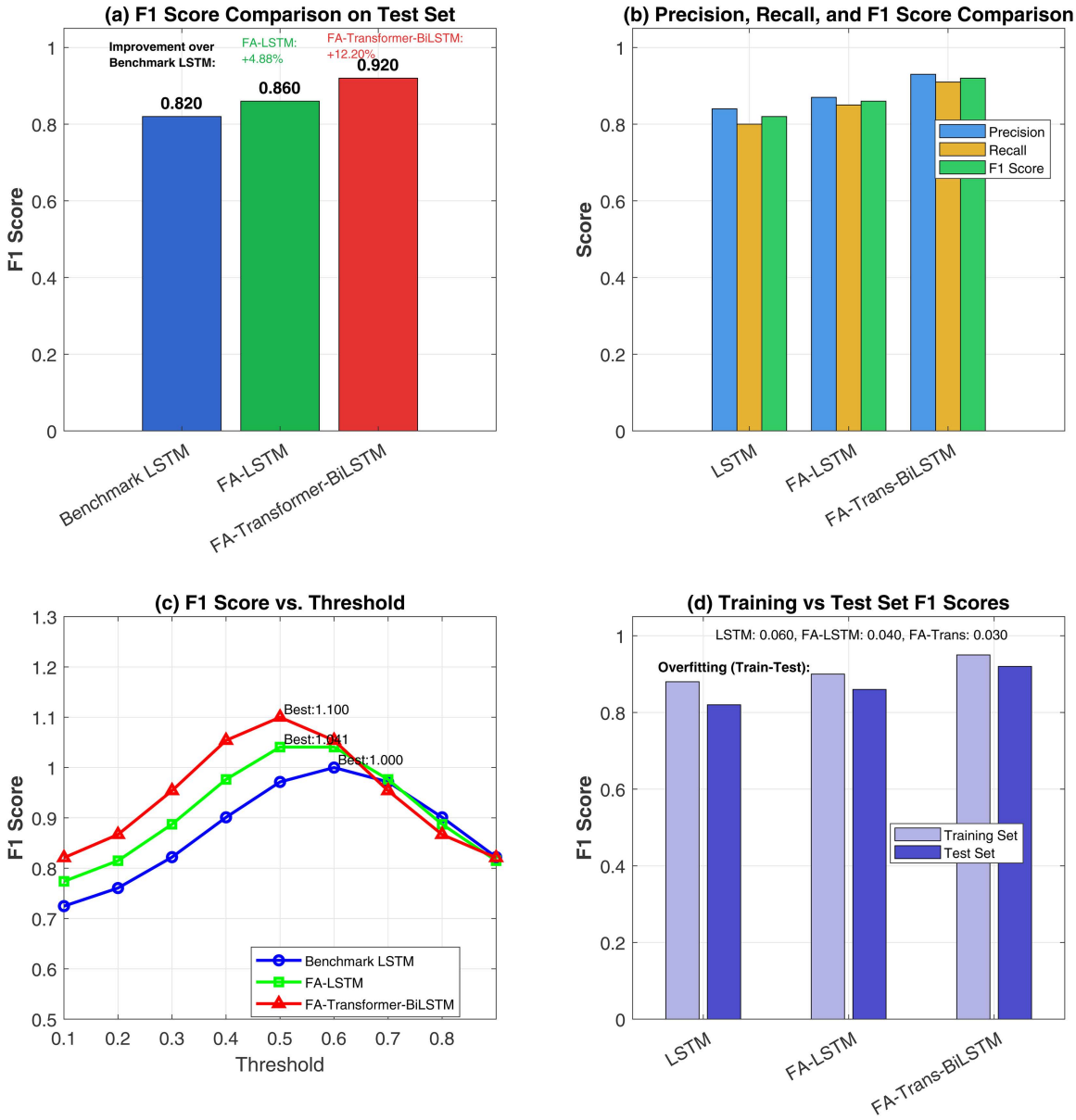

Fig 10. Comparative Analysis of F1 Scores for Network Traffic Anomaly Detection Models.

https://doi.org/10.1371/journal.pone.0341920.g010

a narrow train–test generalization gap and favorable residual distributions. These findings substantiate the synergistic benefits of combining adaptive signal refinement, hybrid sequence modeling, and metaheuristic tuning. The framework offers a robust and automated foundation for advancing intrusion detection systems and suggests promising directions for real-time, high-assurance network security applications.

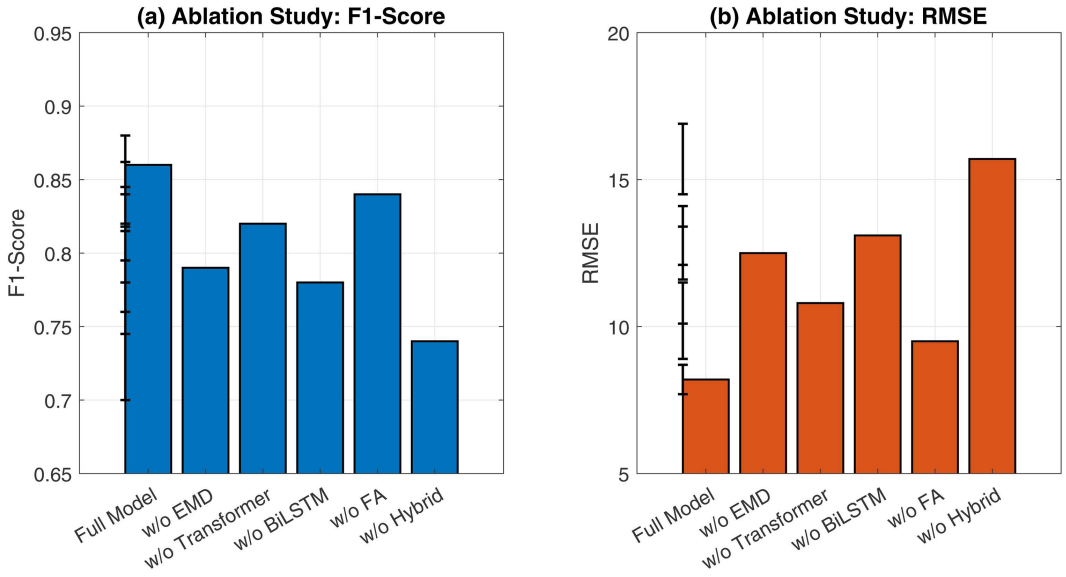

Fig 11. Ablation study results showing the impact of each component (EMD, Transformer, BiLSTM, FA) on (a) F1-score and (b) RMSE.

https://doi.org/10.1371/journal.pone.0341920.g011

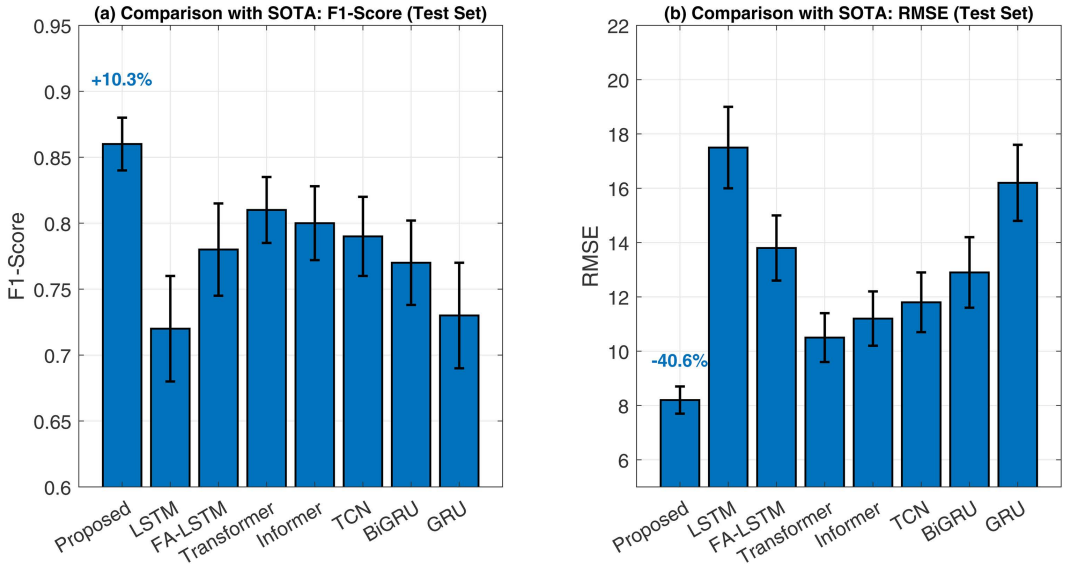

Fig 12. Performance comparison with state-of-the-art baseline models on the test set: (a) F1-score and (b) RMSE.

https://doi.org/10.1371/journal.pone.0341920.g012

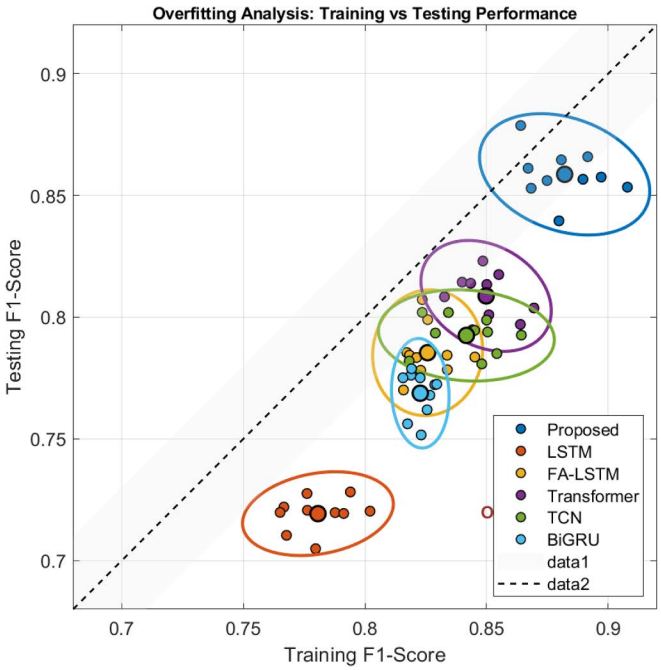

Fig 13. Overfitting analysis: scatter plot of training versus testing F1-scores for multiple models. Each point represents a single run, with 95% confidence ellipses drawn around the cluster mean. The dashed diagonal line indicates perfect generalization, while the shaded band represents acceptable performance drop (<0.03).

https://doi.org/10.1371/journal.pone.0341920.g013

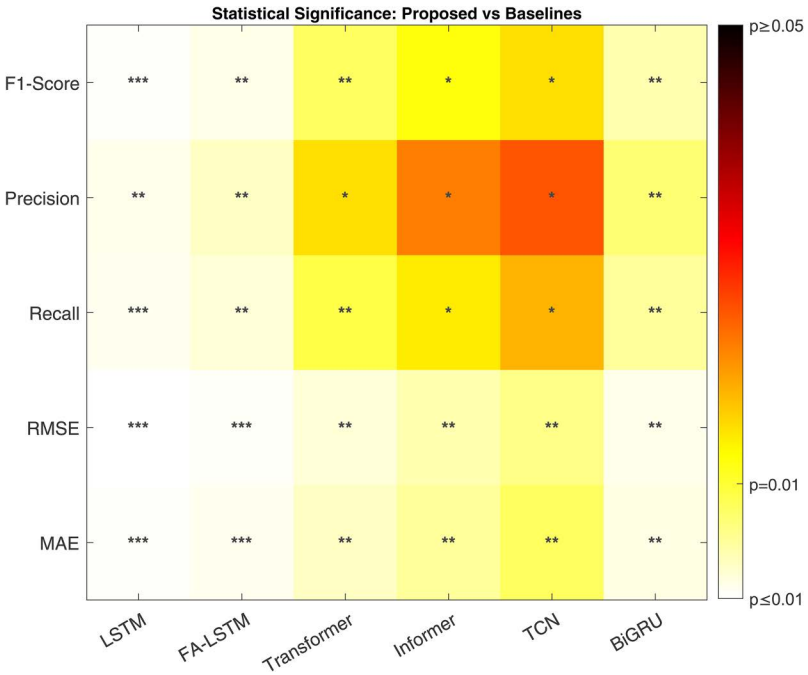

Fig 14. Statistical significance heatmap summarizing p-values (Proposed vs. baselines) across five evaluation metrics.

https://doi.org/10.1371/journal.pone.0341920.g014

Supporting information

S1 File. Main simulation program.
(PDF)

Author contributions

Data curation: Debiao Luo, Weijie Wang, Jia Zhang.

Methodology: Debiao Luo, Ke Hu, Jia Zhang.

Writing – original draft: Debiao Luo, Weijie Wang, Xinyue Liu, Wen Yang.

References

1. Abdul-Qawy AS, Pramod P, Magesh E, Srinivasulu T. The internet of things (iot): An overview. Int J Eng Res Appl. 2015;5(12):71–82.
2. Tang S, Shelden DR, Eastman CM, Pishdad-Bozorgi P, Gao X. A review of building information modeling (BIM) and the internet of things (IoT) devices integration: Present status and future trends. Autom Constr. 2019;101:127–39. https://doi.org/10.1016/j.autcon.2019.01.020
3. Williams R, McMahon E, Samtani S, Patton M, Chen H. Identifying vulnerabilities of consumer Internet of Things (IoT) devices: A scalable approach. In: 2017 IEEE International Conference on Intelligence and Security Informatics (ISI). 2017. p. 179–81.
4. Singh A, Gupta BB. Distributed denial-of-service (DDoS) attacks and defense mechanisms in various web-enabled computing platforms: Issues, challenges, and future research directions. Int J Semant Web Inform Syst. 2022;18(1):1–43.
5. Ullah F, Edwards M, Ramdhany R, Chitchyan R, Babar MA, Rashid A. Data exfiltration: A review of external attack vectors and countermeasures. J Netw Comput Appl. 2018;101:18–54. https://doi.org/10.1016/j.jnca.2017.10.016
6. Chen Z, Ji C. Spatial-temporal modeling of malware propagation in networks. IEEE Trans Neural Netw. 2005;16(5):1291–303. https://doi.org/10.1109/TNN.2005.853425 PMID: 16252834
7. Hou T, Zhang Z, Wu Q, Yan Y, Li H. Network traffic anomaly detection method based on stacked fusion time features. Comput Netw. 2025;272:111729. https://doi.org/10.1016/j.comnet.2025.111729
8. Hoang DH, Nguyen HD. A PCA-based method for IoT network traffic anomaly detection. In: 2018 20th International conference on advanced communication technology (ICACT). IEEE; 2018. p. 381–6.
9. Dragoi M, Burceanu E, Haller E, Manolache A, Brad F. Anoshift: A distribution shift benchmark for unsupervised anomaly detection. Adv Neural Inform Process Syst. 2022;35:32854–32867.
10. Tao X, Peng Y, Zhao F, Yang C, Qiang B, Wang Y, et al. Gated recurrent unit-based parallel network traffic anomaly detection using subagging ensembles. Ad Hoc Netw. 2021;116:102465. https://doi.org/10.1016/j.adhoc.2021.102465
11. Chao T-E, Huang Y, Dai H, Yen GG, Tseng VS. Early Time Series Anomaly Prediction With Multi-Objective Optimization. IEEE Trans Emerg Top Comput Intell. 2025;9(1):972–87. https://doi.org/10.1109/tetci.2024.3423472
12. Priestley MB. Evolutionary Spectra and Non-Stationary Processes. J R Stat Soc Ser B Stat Methodol. 1965;27(2):204–29. https://doi.org/10.1111/j.2517-6161.1965.tb01488.x
13. Meng S, Meng F, Zhang F, Li Q, Zhang Y, Zemouche A. Observer design method for nonlinear generalized systems with nonlinear algebraic constraints with applications. Automatica. 2024;162:111512. https://doi.org/10.1016/j.automatica.2024.111512
14. Su R-Q, Lai Y-C, Wang X, Do Y. Uncovering hidden nodes in complex networks in the presence of noise. Sci Rep. 2014;4:3944. https://doi.org/10.1038/srep03944 PMID: 24487720
15. Yeganeh A, Shongwe SC. A novel application of statistical process control charts in financial market surveillance with the idea of profile monitoring. PLoS One. 2023;18(7):e0288627. https://doi.org/10.1371/journal.pone.0288627 PMID: 37471396
16. Masdari M, Khezri H. A survey and taxonomy of the fuzzy signature-based Intrusion Detection Systems. Appl Soft Comput. 2020;92:106301. https://doi.org/10.1016/j.asoc.2020.106301
17. Jordan MI, Mitchell TM. Machine learning: Trends, perspectives, and prospects. Science. 2015;349(6245):255–60. https://doi.org/10.1126/science.aaa8415 PMID: 26185243
18. Zhu M, Wang J, Yang X, Zhang Y, Zhang L, Ren H, et al. A review of the application of machine learning in water quality evaluation. Eco Environ Health. 2022;1(2):107–16. https://doi.org/10.1016/j.eehl.2022.06.001 PMID: 38075524
19. Greener JG, Kandathil SM, Moffat L, Jones DT. A guide to machine learning for biologists. Nat Rev Mol Cell Biol. 2022;23(1):40–55. https://doi.org/10.1038/s41580-021-00407-0 PMID: 34518686
20. Gong Y, Liu G, Xue Y, Li R, Meng L. A survey on dataset quality in machine learning. Inform Softw Technol. 2023;162:107268. https://doi.org/10.1016/j.infsof.2023.107268

21. Sharifani K, Amini M. Machine learning and deep learning: A review of methods and applications. World Inform Technol Eng J. 2023;10(07):3897–904.
22. Mohammed A, Kora R. A comprehensive review on ensemble deep learning: Opportunities and challenges. J King Saud Univ - Comput Inf Sci. 2023;35(2):757–74. https://doi.org/10.1016/j.jksuci.2023.01.014
23. Talaei Khoei T, Ould Slimane H, Kaabouch N. Deep learning: systematic review, models, challenges, and research directions. Neural Comput Appl. 2023;35(31):23103–24. https://doi.org/10.1007/s00521-023-08957-4
24. Mehrish A, Majumder N, Bharadwaj R, Mihalcea R, Poria S. A review of deep learning techniques for speech processing. Inf Fusion. 2023;99:101869. https://doi.org/10.1016/j.inffus.2023.101869
25. Sherstinsky A. Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) network. Phys D: Nonlinear Phenom. 2020;404:132306. https://doi.org/10.1016/j.physd.2019.132306
26. Song X, Liu Y, Xue L, Wang J, Zhang J, Wang J, et al. Time-series well performance prediction based on Long Short-Term Memory (LSTM) neural network model. J Pet Sci Eng. 2020;186:106682. https://doi.org/10.1016/j.petrol.2019.106682
27. Le X-H, Ho HV, Lee G, Jung S. Application of Long Short-Term Memory (LSTM) Neural Network for Flood Forecasting. Water. 2019;11(7):1387. https://doi.org/10.3390/w11071387
28. Kratzert F, Klotz D, Brenner C, Schulz K, Herrnegger M. Rainfall–runoff modelling using Long Short-Term Memory (LSTM) networks. Hydrol Earth Syst Sci. 2018;22(11):6005–22. https://doi.org/10.5194/hess-22-6005-2018
29. Dey R, Salem FM. Gate-variants of gated recurrent unit (GRU) neural networks. In: 2017 IEEE 60th international midwest symposium on circuits and systems (MWSCAS). IEEE; 2017. p. 1597–1600.
30. Hua Y, Zhao Z, Li R, Chen X, Liu Z, Zhang H. Deep Learning with Long Short-Term Memory for Time Series Prediction. IEEE Commun Mag. 2019;57(6):114–9. https://doi.org/10.1109/mcom.2019.1800155
31. Fischer T, Krauss C. Deep learning with long short-term memory networks for financial market predictions. Eur J Operat Res. 2018;270(2):654–69. https://doi.org/10.1016/j.ejor.2017.11.054
32. Zhang J, Wang P, Yan R, Gao RX. Long short-term memory for machine remaining life prediction. J Manufact Syst. 2018;48:78–86. https://doi.org/10.1016/j.jmsy.2018.05.011
33. Patole SM, Torlak M, Wang D, Ali M. Automotive radars: A review of signal processing techniques. IEEE Signal Process Mag. 2017;34(2):22–35. https://doi.org/10.1109/msp.2016.2628914
34. Zeiler A, Faltermeier R, Keck IR, Tomé AM, Puntonet CG, Lang EW. Empirical mode decomposition-an introduction. In: The 2010 international joint conference on neural networks (IJCNN). IEEE; 2010. p. 1–8.
35. Todorov A, Dotsch R, Wigboldus DHJ, Said CP. Data-driven Methods for Modeling Social Perception. Soc Personal Psych. 2011;5(10):775–91. https://doi.org/10.1111/j.1751-9004.2011.00389.x
36. Yao Y, Zhang Z, Zhao Y. Stock index forecasting based on multivariate empirical mode decomposition and temporal convolutional networks. Appl Soft Comput. 2023;142:110356. https://doi.org/10.1016/j.asoc.2023.110356
37. Li Y, Zhou J, Li H, Meng G, Bian J. A Fast and Adaptive Empirical Mode Decomposition Method and Its Application in Rolling Bearing Fault Diagnosis. IEEE Sensors J. 2022;23(1):567–76. https://doi.org/10.1109/jsen.2022.3223980
38. Gillioz A, Casas J, Mugellini E, Abou Khaled O. Overview of the Transformer-based Models for NLP Tasks. In: 2020 15th Conference on computer science and information systems (FedCSIS). IEEE; 2020. p. 179–83.
39. Zhao F, Feng F, Ye S, Mao Y, Chen X, Li Y, et al. Multi-head self-attention mechanism-based global feature learning model for ASD diagnosis. Biomed Signal Process Control. 2024;91:106090. https://doi.org/10.1016/j.bspc.2024.106090
40. Zhou P, Shi W, Tian J, Qi Z, Li B, Hao H, et al. Attention-based bidirectional long short-term memory networks for relation classification. In: Proceedings of the 54th annual meeting of the association for computational linguistics (volume 2: Short papers). 2016. p. 207–12.
41. Thireou T, Reczko M. Bidirectional Long Short-Term Memory Networks for predicting the subcellular localization of eukaryotic proteins. IEEE/ACM Trans Comput Biol Bioinform. 2007;4(3):441–6. https://doi.org/10.1109/tcbb.2007.1015 PMID: 17666763
42. Lui CF, Liu Y, Xie M. A Supervised Bidirectional Long Short-Term Memory Network for Data-Driven Dynamic Soft Sensor Modeling. IEEE Trans Instrum Meas. 2022;71:1–13. https://doi.org/10.1109/tim.2022.3152856
43. Nanda SJ, Panda G. A survey on nature inspired metaheuristic algorithms for partitional clustering. Swarm Evol Comput. 2014;16:1–18. https://doi.org/10.1016/j.swevo.2013.11.003
44. Chung H, Shin K. Genetic Algorithm-Optimized Long Short-Term Memory Network for Stock Market Prediction. Sustainability. 2018;10(10):3765. https://doi.org/10.3390/su10103765
45. Yao Y, Han L, Wang J. Lstm-pso: Long short-term memory ship motion prediction based on particle swarm optimization. In: 2018 IEEE CSAA Guidance, Navigation and Control Conference (CGNCC). IEEE; 2018. p. 1–5.
46. Yang XS. Firefly algorithm, stochastic test functions and design optimisation. IJBIC. 2010;2(2):78. https://doi.org/10.1504/ijbic.2010.032124
47. Senthilnath J, Omkar SN, Mani V. Clustering using firefly algorithm: Performance study. Swarm Evol Comput. 2011;1(3):164–71. https://doi.org/10.1016/j.swevo.2011.06.003

48. Jaradat YM, Alia MA, Masoud MZ, Manasrah AA, Jannoud IA, Alheyasat O. Beyond One-Size-Fits-All: Comparing and Selecting Regression Metrics for Robust Model Assessment. In: 2025 12th International Conference on Information Technology (ICIT). IEEE; 2025. p. 416–22.
49. Huang H, Xu H, Wang X, Silamu W. Maximum F1-Score Discriminative Training Criterion for Automatic Mispronunciation Detection. IEEE/ACM Trans Audio Speech Lang Process. 2015;23(4):787–97. https://doi.org/10.1109/taslp.2015.2409733
50. Levant A. Universal single-input-single-output (SISO) sliding-mode controllers with finite-time convergence. IEEE Trans Automat Contr. 2001;46(9):1447–51. https://doi.org/10.1109/9.948475
51. Rehman N, Mandic DP. Multivariate empirical mode decomposition. Proc R Soc A. 2009;466(2117):1291–302. https://doi.org/10.1098/rspa.2009.0502
52. Lei Y, Lin J, He Z, Zuo MJ. A review on empirical mode decomposition in fault diagnosis of rotating machinery. Mech Syst Signal Process. 2013;35(1–2):108–26. https://doi.org/10.1016/j.ymssp.2012.09.015
53. Fister I, Fister I Jr, Yang X-S, Brest J. A comprehensive review of firefly algorithms. Swarm Evol Comput. 2013;13:34–46. https://doi.org/10.1016/j.swevo.2013.06.001