Eng Int Syst (2026) 5: 665–681
© 2026 CRL Publishing Ltd

Engineering
Intelligent Systems

Statistical Analysis and Advanced Predictive Techniques to Optimize Detection of Network Traffic Anomaly and Improve Network Security and Threat Identification

Zhe Zhang[a] **and Xiangjiao Liu**[b]*

School of Artificial Intelligence and Software Engineering, Nanyang Normal University, Nanyang 473061, Henan, China

Traffic patterns in today's network environment vary considerably, and traditional anomaly detection methods were not designed to address the various issues relating to the high dimensionality of structural representations of network topologies, temporal dependencies, and real-time response, leading to decreased accuracy and consistency in identifying anomalies. The research presented in this paper proposes a novel method of detecting anomalies within network traffic utilizing statistical analysis optimized by means of advanced predictive techniques. Using the structured features of cleaned-up network traffic datasets, the study creates a statistical feature analysis approach in order to develop a baseline of normal network traffic behaviour; this provides insight into the statistical nature of your organisation's network traffic patterns. In addition, the paper utilizes an LSTM and a multi-level predictive model to develop a methodology to predict the future behaviour of the network traffic. Finally, by leveraging the residual-based anomaly scoring techniques, experiments demonstrate the ability to support accurate and timely identification of network traffic anomalies. The experiments were conducted on many different types of network traffic datasets and resulted in an anomaly detection accuracy of 96.8%, an F1 score of 95.9%, a false positive rate of 0.02, a false negative rate of 0.11, and an average residual prediction convergence rate of 0.0125 during training. The median time required for the detection of anomalies end-to-end was 0.20 seconds. The results presented in this paper verify that the proposed methodology has the potential for very precise and low-latency detection of network traffic anomalies and will provide technical support for the identification of network security and network threat risks.

Keywords: Network traffic anomaly detection; advanced prediction techniques; statistical analysis; network security; threat identification

1. INTRODUCTION

The rapidly changing nature of the network landscape has created an unprecedented level of network traffic that is both dynamic and sequentially timed (i.e., both temporal and physical). The complexity of network traffic structures is characterized by high-dimensional structures and multi-source heterogeneity with varying degrees of complexity within these interactions. This translates into an increasing need for an anomaly-detection mechanism that can protect against risk in any type of networked environment. The ability to accurately identify and promptly respond to deviations from the normal operating state has become a critical requirement because, increasingly, anomalous activity is able to be concealed within complex systems [1–2]. More

*Corresponding author. [a]Email: 27555859@qq.com, [b]liuxiangjiao042@163.com

effective and efficient anomaly detection techniques will play a vital role in decreasing the potential impact of attacks, improving the resilience of systems and improving the capability of proactive defence mechanisms [3–5].

Anomaly detection has many challenges because of the current technology, which cannot adequately detect anomalies. The traditional statistical approaches used for anomaly detection have many limitations when characterizing the flow features that change over time, when facing distribution drift, and when trying to model the many interactions among the variables contained within large (i.e., high dimensional) interaction structures. For example, in order to maintain stable thresholds and reliable baselines within dynamic network environments, it is imperative to have the means to monitor the thresholds and baselines of the data used for detection. The traditional methods of feature creation through handcrafted features and fixed-window models for the creation of the long-term window to monitor the temporal correlations of the data produce limited results because there are substantial nonlinear dependencies for the data contained within the long-term windows of the high-dimensional sequences [6–7]. Moreover, the normal modes of operation of large-scale systems change over time, which introduces the problem of losing the distribution stability of the training data; in addition, detection techniques based on the use of static statistics may not provide stable results in terms of temporal consistency [8]. Furthermore, the number of items identified as anomalous is always much smaller than the number of normal items, resulting in the development of a heavily imbalanced dataset and causing the detection models to learn the biased data [9–10]. In some cases, the attack pattern exhibits only a small perturbation of signal strength or, in many cases, the attack patterns are masked by background noise, making it very difficult for an analyst to differentiate between the aberrant and the normal features [11–12].Cross-scenario deployment faces challenges due to domain distribution differences, limiting the model's generalization ability [13]. Prediction techniques still face problems such as large fluctuations in input sequences, strong local mutations, and the intertwining of short-term patterns and long-term dependencies when processing network traffic, resulting in a lack of stable structure in prediction residuals in real-time detection scenarios. Collectively, these problems weaken the effectiveness of traditional statistical methods and prediction techniques when applied alone, making it difficult to meet the requirements of real-time performance and accuracy simultaneously.

Recent research has attempted to alleviate these difficulties by means of deep representation learning, graph structure modeling, generative modeling, and temporal prediction. Although reconstruction-based methods using autoencoders improve anomaly discrimination by learning low-dimensional manifolds of normal patterns, reconstruction residuals in noisy environments are prone to unstable convergence [14]. Dependency modeling based on graph neural networks is suitable for indicating inter-system associations, but its performance degrades with dynamic topology and node semantic changes [15]. Distribution fitting based on generative adversarial networks can learn complex data distributions, but the training process is susceptible to pattern collapse [16–17]. Although long-sequence modeling based on the Transformer architecture enhances global dependency extraction, its high computational cost limits its applicability in real-time scenarios [18]. Distribution learning based on diffusion models improves the distribution discrimination of anomaly scores; however, the iterative process in the inference phase affects detection latency [19]. Overall, existing methods still require a trade-off between real-time performance, stability, and resource consumption in network traffic scenarios, failing to address the core issues of insufficient statistical stability and unstable prediction residual structure. These are the main technical issues currently faced by the system.

To overcome the challenges posed by weak anomaly feature separation, unstable convergence of residuals, and of the time lapse between the initial indication of an anomaly and its detection, this paper proposes a method that integrates statistical and prediction techniques for the analysis of these problems. The method uses a multi-stage encoding scheme that enables the extraction of semantic feature information from normal patterns at various times. By applying structural consistency constraints to the multi-stage encoding of normal patterns, the method is able to stabilize the representation of these normal patterns so that the residuals produce a well-defined structure that is easily identifiable as a clear clustering structure. Additionally, the paper utilizes a module that helps to identify the temporal relationships of normal sequences by fitting the normal sequence to a time-series using residual prediction, as well as scoring the offset of the anomaly segments in the multi-stage residual representation. The combination of the statistical analysis method and the prediction techniques yields a combination of static and dynamic features with respect to anomalies. Through joint training, the method can optimize the feature encoding, structural constraints, and residual prediction functions, thereby developing a low-latency and interpretable mechanism for the scoring of anomalies. The enhanced robustness, real-time performance, and detection capability of the proposed method will enable security providers to more easily detect dynamic threats in complex environments.

2. DESIGN OF AN ANOMALY DETECTION METHOD BASED ON STATISTICAL-PREDICTIVE CO-OPTIMIZATION

2.1 Network Traffic Data Processing and Feature Construction

To create effective interpretable input for future base statistics modeling or time sequence predictive analysis, this piece of the data pipeline systematically breaks down and builds system features from the raw data associated with the network traffic. The raw data is imported from the raw traffic sources into this part of the pipeline. Here, the pipeline checks the integrity of the data and synchronizes the raw data based on the time sent/received. Similar to packets in a network, these flow records are stored based on the direction of the packets,

using the Flow Definition for determining the flow direction. The Flow Definition states that a flow record comprises a combination of the five-tuple related to the packet source and target IP addresses, sources and targets for port numbers, and protocol and time of aggregation for a session-level exchange. This reconstructed flow record is consistent with CICFlowMeter's method of constructing a flow record by collecting the individual packet records at the packet level, generating statistics based on the assembly of these records by packet length, sequence of arrival, and by the way they are aggregated over time. Flag counts and directional ratios are extracted from the protocol layer to characterize session behavior. To transform packet-level information into model-friendly time-series input, fixed-window sliding aggregation is used, mapping each time window t to a d-dimensional feature vector X_t, defined as:

$$X_t = [f_{1,t}, f_{2,t}, \ldots, f_{d,t}]^T \tag{1}$$

where, $f_{j,t}$ represents the aggregated statistic calculated for the j-th feature within time window t, such as mean, variance, or count. The vector dimension d is determined by pre-selection of features.

Feature preprocessing includes normalization and robustness operations to suppress the impact of distribution drift on subsequent modeling. Continuous features are standardized using Z-score, which is calculated as follows:

$$\tilde{x} = \frac{x - \mu}{\sigma} \tag{2}$$

where, x denotes the original value of the feature; μ represents the mean of that feature observed on the training data, and σ the standard deviation. Prior to applying equation (2), the features that exhibit long-tailed distributions undergo a logarithmic transformation to enable a better normal approximation to be generated. Also, the robustness against outliers can be increased by restricting any outliers found to a specific interval limit using a quantile-based truncation technique thus reducing the amount of extreme data which contaminates estimates for standardized parameters.

The selection of feature dimensions involves two stages of screening. The first stage ranks possible features based on their representation capabilities and their potential costs in terms of real-time computation. Candidate features are ranked according to their importance scores calculated by means of a tree-based methodology, and are used to determine mutual information between them [20–21]. The second stage removes repeatedly-used lagged descriptors from consideration based on their temporal correlation test, and retains only a small subset of those that provide significant value as predictors within all the time periods examined[22]. Additional methods are required to compensate for class imbalance in the construction of training sets in order to reduce model bias during training while retaining the actual structure of the time-distributed real traffic distributions. This is done using an oversampling strategy for temporally correlated samples and a loss reweighting strategy based on sample weights.

Statistical feature analysis indicates the pattern of the amount of vehicle traffic in an area during a specific period of time. This enables prediction models to have a structured form for the mapping of time periods to enable predictions to be made and achieve convergence to the steady state form of the model. The normal traffic pattern during a certain time period will exhibit measurable and statistically regular patterns of behaviour. By providing this statistical behaviour to the LSTM model, this allows the predictions produced by the LSTM to depend on the steady state behaviour of the traffic flow while also learning to recognize the dynamic nature of changes, thereby improving the model's ability to predict "good" trends while being less susceptible to new noise.

To facilitate the subsequent design of anomaly scoring based on residuals, the predicted target sequence y_t and the model input sequence $X_{t-w+1:t}$ need to be generated simultaneously during the feature construction stage. The window length w is determined in the validation set by means of sliding window cross-validation. The prediction residual (signed residual) is defined as:

$$r_t = y_t - \hat{y}_t \tag{3}$$

where, y_t represents the true target value, and $\hat{y}_t$ represents the output estimate of the prediction model at time t. The residual sequence is statistically analyzed through a sliding window to estimate the local scale of the residual distribution, which is used as a standardized benchmark for subsequent anomaly scoring.

When generating target and input sequences, the synergistic relationship between statistical features and prediction models is clearly established: statistical features characterize the steady-state baseline, and prediction models learn dynamic trends. After the two are integrated, the prediction residuals form a separable structure around the steady-state baseline, which reduces the impact of noise and improves the explanatory power of the residuals, making the residual space more suitable for the detection of anomaly offsets.

A feature-grouping strategy is used to make features more interpretable by separating them into three distinct groups: (1) time series statistics; (2) protocol behaviour; and (3) interaction relationships. When developing models and applying them to on-line analyses for event detection, each feature's contribution from both training and serving the model is stored separately so that alarm interpretations can be derived from every input available. This process involves constructing, normalizing, and grouping the features. Figure 1 illustrates how these three processes relate to each other. Selected representative fields associated with traffic activity from the current study are listed in Table 1, including their types and ranges of values.

Figure 1 illustrates the complete modeling process of network traffic from raw packet input to anomaly score output. First, the underlying packets are transformed into bidirectional flow records through session reassembly and flow construction. Subsequently, statistical features such as packet length, arrival time, protocol behavior, and directionality ratio are extracted. After time-window sliding aggregation and normalization, a stable statistical baseline model is established. The statistical features and baseline information are fed into the deep prediction model in a collaborative manner to show the temporal changing trend of traffic. The prediction residual between the model output

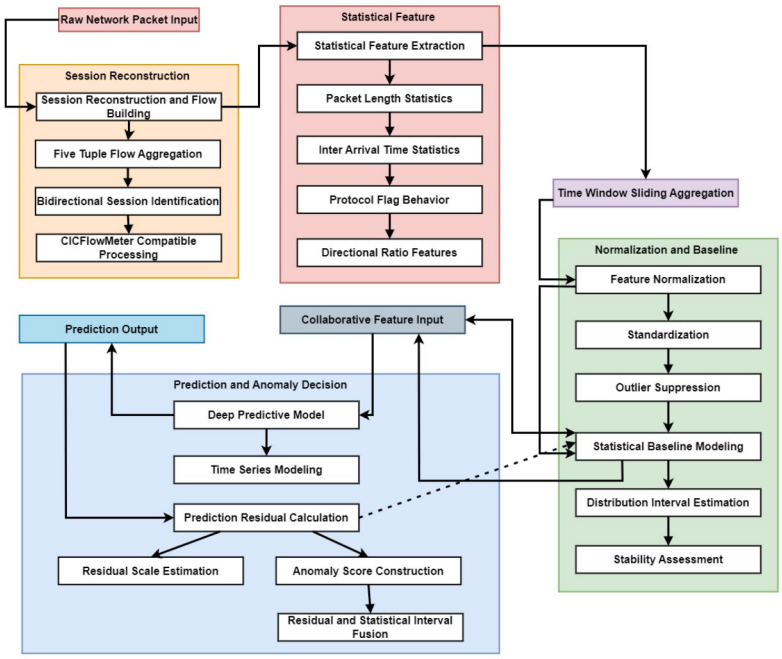


Figure 1 Framework of statistical-predictive collaborative feature processing.

Table 1 Comparison of dataset structure features.

Field Name	Field Type	Typical Range	Description
PacketLengthMean	Continuous	0–1500 (bytes)	Mean packet length within a single flow, measuring the distribution of packet sizes
PacketLengthStd	Continuous	0–750 (bytes)	Standard deviation of packet lengths, reflecting size variation within the flow
InterArrivalMean	Continuous	0–10 (seconds)	Average inter-arrival time of packets, characterizing traffic rate features
FlowDuration	Continuous	0–3600 (seconds)	Duration of the flow, indicating session length
TCPFlagsCount	Discrete	0–10000 (count)	Cumulative count of various TCP((Transmission Control Protocol)) flags, representing protocol behavior
SrcPortEntropy	Continuous	0–8 (bits)	Entropy of source ports, reflecting the complexity of port usage distribution

and the actual observation is further used to construct the anomaly score. At the same time, the residual distribution participates in the adaptive update of the statistical baseline through the feedback path, thereby introducing a prediction-driven dynamic adjustment mechanism while maintaining statistical robustness, and achieving continuous perception and accurate judgment of network traffic anomalies.

Table 1 lists the six representative feature fields selected for the study, their typical numerical ranges, and functional descriptions.

The features listed in Table 1 are selected based on the flow-level feature extraction conventions of CICFlowMeter and UNSW-NB15. The numerical ranges of the fields are derived from empirical statistics of training subsets of public datasets and summaries of common network protocol constraints. The selection aims to cover three types of information: traffic intensity, persistence, and behavioral patterns, providing multi-dimensional support for statistical baseline modeling and predictive model input.

2.2 Establishment of Statistical Baselines for Normal Behavior

In this phase of the study, a robust baseline is constructed for normal flow in order to define “normal" using quantitative statistical characterization and to provide testable thresholds and discriminants for the subsequent determination of prediction residuals. The process begins with univariate distribution estimation, obtaining the empirical probability density function for each feature through kernel density estimation (KDE) to depict the shape of the nonparametric distribution. A sliding window with a smoothing factor is used to track the density changes over time [23–25]. For multivariate relationships, robust covariance estimation and Mahalanobis distance are used to construct a joint baseline measure to reflect the correlation and co-variability among features. To accommodate the time-varying nature of flow, the baseline uses an exponentially weighted moving average (EWMA) to smooth the statistics over time, thus forming

Table 2 Summary of statistical characteristics of normal behavior.

Statistical Feature	Estimation Method	Reference Interval in Training Phase	Stability Score
Mean μ	Sample mean (training set)	$[\mu_{\text{low}}, \mu_{\text{high}}]$	0–1 (robustness score)
Variance σ^2	Bias-corrected sample variance	$[\sigma^2_{\text{low}}, \sigma^2_{\text{high}}]$	0–1
Kurtosis	Second-order moment of kernel density potential distribution	$[k_{\text{low}}, k_{\text{high}}]$	0–1
Skewness	Standardized third central moment	$[s_{\text{low}}, s_{\text{high}}]$	0–1
Mahalanobis Distance Threshold	Chi-square critical value $\chi^2_{d,\alpha}$	—	0–1
Burstiness Index	Ratio metric based on peak rate	$[b_{\text{low}}, b_{\text{high}}]$	0–1

dynamic control limits [26]. This process uses the training set as a reference window to estimate the initial density and covariance, and then uses a sliding update rule in the online stage to maintain the temporal consistency of the baseline.

The statistical baseline does more than determine the deviation: it is also used to generate a stable normal behavior structure in the time dimension so that the subsequent prediction module has a fixed reference point in the input stage. Through the kernel density distribution and the time series mean and variance of the EWMA output, the steady-state form of normal flow is quantified and expressed, and a constrained stability band is formed in the feature space, thereby reducing the disturbance of high-frequency noise and periodic fluctuations on the input structure.

The univariate probability density obtained by kernel density estimation $\hat{p}_j(x)$is calculated as follows:

$$\hat{p}_j(x) = \frac{1}{nh_j} \sum_{i=1}^{n} K\left(\frac{x - x_{i,j}}{h_j}\right) \tag{4}$$

where, $\hat{p}_j(x)$ represents the empirical density estimate of the j-th feature; n is the number of samples used for estimation; $x_{i,j}$ represents the observation of the i-th sample on the j-th feature; h_j is the bandwidth parameter of the feature; and $K(\cdot)$ represents the kernel function. Kernel density estimation is used to derive the confidence interval boundaries and deviation probabilities of the feature to assist in threshold setting. The joint baseline metric uses Mahalanobis distance to measure the offset of the flow vector relative to the normal distribution [27]. It is defined as:

$$D_M(X_t) = \sqrt{(X_t - \mu)^T \Sigma^{-1} (X_t - \mu)} \tag{5}$$

where, X_t represents the eigenvector of time window t, μ is the eigen mean vector estimated during training, Σ represents the covariance matrix estimated during training, and Σ^{-1} is the inverse of the covariance matrix. The square of the Mahalanobis distance follows a chi-square distribution under the assumption of multidimensional normality; therefore, the threshold is set to $\chi^2_{d,\alpha}$ (the chi-square quantile with degrees of freedom of dimension d and a significance level of α) to determine the significance of joint deviation; expressed by the equation:

$$\mathbb{I}\{D^2_M(X_t) > \chi^2_{d,\alpha}\} \tag{6}$$

where, $D^2_M(X_t)$ represents the square value of equation (5), and $\chi^2_{d,\alpha}$ represents the critical value of the chi-square distribution at the significance level α, indicating whether a joint deviation alarm is triggered. To improve the robustness of covariance estimation under insufficient sample size or noise, a covariance estimator with shrinkage is used to provide a stable Σ estimate during the training period, and μ and Σ are updated slightly in an exponentially weighted manner during the online phase to maintain temporal consistency [28].

The baseline for traffic statistics needs to remain smooth over time and not destabilize too quickly in the face of sudden fluctuations. Therefore, EWMA is used to smooth key statistics, defined as follows:

$$s_t = \lambda x_t + (1 - \lambda) s_{t-1} \tag{7}$$

where, s_t represents the smoothing statistic at time t, x_t represents the original statistical value observed at that time, and the parameter $\lambda \in (0, 1$ is represented as the smoothing factor of EWMA, controlling the sensitivity to new observations. The mean and variance estimates of the EWMA output are used as the dynamic midline and bandwidth of the control chart, thereby reducing the impact of diurnal periodic fluctuations on threshold determination.

To facilitate the quantification of baseline characteristics and provide interpretable statistical descriptions for subsequent models, a set of robust statistical indicators was calculated for each feature during the training period, presented in a table for subsequent judgment and visualization. Table 2 shows the commonly used statistical features, estimation methods, training period observation intervals, and stability scores in the study.

The training period reference intervals in Table 2 are jointly determined by the empirical quantiles and kernel density estimates of the training set. The stability score is back-mapped to the 0–1 interval using the temporal variance of the training period statistic; the closer the value is to 1, the higher the temporal stability of the statistical feature during the training period. The Mahalanobis distance threshold lists the chi-square limit parameters used for joint deviation determination, and the burst index is a normalized ratio that measures the surge in traffic within a short time window.

During the training phase, the initial parameters are estimated using normal subsets of CIC-IDS2017 and UNSW-NB15. During the online phase, the EWMA update rule is used to maintain the temporal consistency of the parameters. The density structure, mean curve, variance curve, and joint offset limit output by statistical modeling are used as background constraint inputs for the prediction model, so that the residuals have an interpretable steady-state reference during the decision phase, forming a statistical-prediction collaborative anomaly-detection link.

2.3 Advanced Prediction Model Construction and Training Strategies

Periodic fluctuations of network traffic are represented by short-term dynamic changes that occur over time. Network traffic also demonstrates spatial and temporal interactions with multi-dimensional characteristics (statistical attributes) that are characteristic of statistical patterns of interaction over time. To enable the prediction of a persistent set of patterns of traffic, the current study developed a deep learning model, known as a deep prediction model, using statistical baseline parameters for predictive input data. The deep prediction model obtains temporal dependencies for traffic prediction while preserving the steady-state structure characteristic of normal traffic behaviour and predicting those patterns into the future [29]. The primary input to the prediction model was a joint feature vector of the traffic time series and the statistical baseline, which generated predictions of future traffic from the temporal relationships established using parameterized temporal mappings. The prediction accuracy of the deep prediction model was improved while training using supervised learning through minimization of prediction errors, thereby providing consistent inputs to the next component, an anomaly scoring mechanism, to support the scoring of anomalies [30–31].

Within the prediction framework, the long-term dependency modeling function of LSTM maintains a causal coupling with the steady-state characterization of the statistical baseline. The temporal unit extracts structural trends across windows, while the statistical baseline constrains short-term fluctuations on the input side, ensuring the model maintains a stable internal memory structure in the face of noise disturbances. The prediction residuals are shaped by both long-term trend shifts and short-term steady-state deviations, resulting in a concentrated distribution of the residual sequence in the normal range and a shifted structure away from the steady-state center in the abnormal range. Without statistical steady-state constraints, the residuals would exhibit irregular oscillations due to noise; without the temporal prediction module, statistical threshold judgments would become unstable and drift over time. After their fusion, the residuals form a concentrated normal band around the normal baseline, while the shifts in abnormal flows exhibit clear abrupt changes in the residuals.

When defining the prediction process, the input vector x_t represents the joint representation concatenated from time series features and statistical baseline features, and the hidden state vector h_t describes the internal memory structure of the model at time t. The state update process of the model is determined by equation (8):

$$h_t = f_\theta(x_{t2}, h_{t-1}) \tag{8}$$

where, x_{t2} represents the multi-source feature input, h_{t-1} represents the network memory state at the previous time step, and $f_\theta(\cdot)$ represents the nonlinear mapping function controlled by the parameter set θ.

When generating future flow estimates, the output of the predicted value $\hat{y}_{t+1}$ is represented as follows:

$$\hat{y}_{t+1} = W_o h_t + b_o \tag{9}$$

where, $\widehat{y}_{t+1}$ represents the model's prediction of the flow rate at the next time step, W_o represents the output layer weight matrix, and b_o represents the output layer bias vector.

During model training, parameter updates are supervised by constructing a loss function, and the mean squared error is used to represent the prediction fit quality. The loss function is defined as:

$$\mathcal{L} = \frac{1}{N}\sum_{t=1}^{N}(y_t - \hat{y}_t)^2 \tag{10}$$

where, y_t represents the actual flow observation at time t, $\widehat{y}_t$ prediction model estimates the flow at time t, and N represents the number of valid samples in the time series.

The output of the prediction model after training is used to construct the residual sequence. The residuals measure the instantaneous fluctuation of the prediction error and indirectly indicate whether the traffic flow deviates from normal behavior patterns. The residuals used in this process are absolute residuals. Their definition is as follows:

$$r_t = |y_t - \hat{y}_t| \tag{11}$$

where, r_t represents the absolute residual strength at time t.

The statistical baseline maintains a steady state while training, producing predicted values that will fluctuate around the baseline as it is created. The predicted values also establish the LSTM's ability to ensure inter-port reliability over all time frames, thereby creating sufficient conditions for the formation of separate clusters of residuals within normal values. Any irregular flow will destabilize both the temporal correlation and the statistical properties of the residuals, resulting in the considerable shifting of the residuals as they leave the steady state. Together, the predictive module and the statistical baseline help clarify the differences between the normal and abnormal distributions of residuals, thereby creating a strong platform for the detection of anomalies.

The comparison trend between predicted performance and actual flow is used to confirm whether the model achieves a stable fit within the normal behavior range. To illustrate the dynamic relationship between the predicted curve and the actual flow, this section visualizes the model's predicted sequence and the actual sequence. Figure 2 shows the small residual fluctuations within the normal range and the significantly increased deviation within the abnormal range.

In Figure 2(a), the actual flow rate at time points 1–5 is between 98 req/s and 115 req/s, while the predicted flow rate is between 97 req/s and 109 req/s. The two are relatively

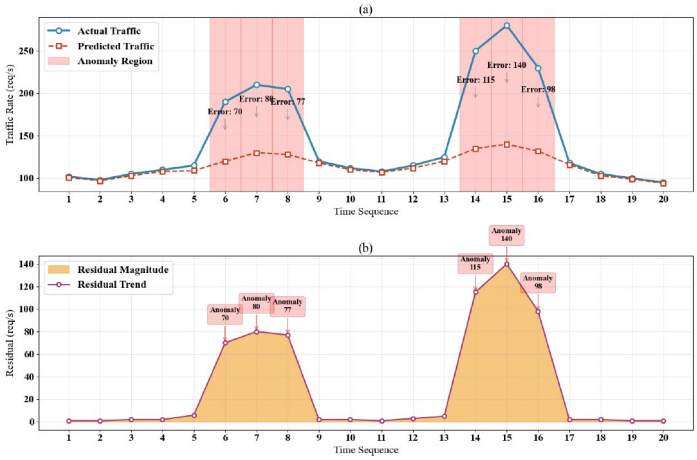


Figure 2 Comparison of model prediction and actual flow. Figure 2(a). Comparison of actual flow and predicted flow. Figure 2(b). Analysis of the variation of predicted residuals over time.

close, indicating that the model's time series modeling based on the statistical baseline can effectively fit stable fluctuations. In time points 6–8, the actual flow rate suddenly increases to 190–210 req/s, while the predicted value is only 120–130 req/s, and the residual jumps to 70–80 req/s. This is because the sudden flow rate exceeds the normal range established during the training phase, causing the historical patterns relied upon by the LSTM to become invalid. In Figure 2(b), the residual sequence remains between 1 and 5 during normal periods, and suddenly increases to 70 req/s to 140 req/s after entering the abnormal period. At time point 15, the actual flow rate was 280 req/s (residual peak 140 req/s), due to the fact that the abnormal flow rate creates a temporal deviation (time between events) as well as a geographic statistical distribution shift. Both of these deviations increase the residual score prediction error. This results in an abnormal flow creating a significantly different/separable signal in the prediction residual. As shown in Figure 2, the prediction residuals are produced by an abusive flow as a result of their disrupting of temporal correlation, and statically stabilizing the prediction residual and making it easier for the prediction model to predict correct future event patterns.

2.4 Adaptive Anomaly Detection Strategy Based on Residual-Statistical Fusion

Anomaly detection is centered on predicted residuals. The final alarm decision process is formed by scaling the statistical baselines and fusing multi-source scores, aiming to maintain high detection sensitivity while reducing false alarm rates and ensuring controllable online response latency. The residual sequence is locally scaled to generate normalized residual terms. Normalization uses the local mean and standard deviation of the training-period residual distribution for standardization. This is defined as:

$$\tilde{r}_t = \frac{r_t - \mu_r(t)}{\sigma_r(t)} \tag{12}$$

where, r_t represents the absolute residual at time t, $\mu_r(t)$ is the empirical mean estimate of the residual within the sliding window, and $\sigma_r(t)$ represents the empirical standard deviation estimate of the residual within the same sliding window. It should be noted that the standardization in equation (12) is different from the global normalization for the input features in equation (2). In this section, the normalization applies to the prediction residual, and uses the local mean and standard deviation of the residual sequence during the training period for dynamic standardization to reflect the transient intensity of the residual deviation, which is used for subsequent anomaly scoring calculation. The sliding window uses exponential weighting to maintain temporal adaptability; its update form is given in equation (15). The Mahalanobis distance metric is used to measure the deviation of joint features to indicate the collaborative shift of multidimensional features. The Mahalanobis distance is scaled after logarithmic transformation, denoted as m_t.

Normalized residuals and scaled Mahalanobis distances are fused into a composite outlier score using a weighted rule, which is a weighted linear sum.

$$S_t = \tilde{\alpha} \cdot \tilde{r}_t + \beta \cdot \frac{\ln(1+m_t)}{\ln(1+m_{ref})} \tag{13}$$

where, S_t represents the composite anomaly score at time t; $\tilde{\alpha}$ and β represents the normalized score weight coefficient and satisfies $\tilde{\alpha}+\beta = 1$; m_{ref} represents the baseline Mahalanobis distance value during the training period to avoid the denominator being zero and to achieve dimensionless processing. The weight coefficient is determined by a grid search on the validation set to balance the contribution of residuals and covariances to the final decision.

The alarm determination employs a two-stage mechanism of adaptive threshold T_t and short-term peak detection. Short-term peak detection identifies significant spikes in the score sequence so as to improve response to sudden attacks. The two-stage threshold determination uses an EWMA update strategy to maintain a dynamic threshold. The EWMA threshold is defined as follows:

$$T_t = \gamma T_{t-1} + (1-\gamma) q_\rho(S_{t-L+1t}) \tag{14}$$

where, T_t represents the current threshold, $\gamma \in (0, 1)$ is the threshold decay factor, and $q_\rho(S_{t-L+1t})$ represents the ρ quantile calculated on a historical score sequence of length L, used to reflect the upward baseline of recent scores. The threshold determination trigger condition is that a high-confidence alarm is generated when $S_t > T_t$ and the short-term peak detection flag is true, and a low-confidence alarm is generated when $S_t > T_t$ is satisfied alone for subsequent correlation analysis. The short-term peak detection is based on the threshold determination of the sliding residual growth rate $g_t = (r_t - r_{t-1})/r_{t-1}$ to improve the sensitivity to rate-type attacks.

Online deployments utilize an incremental computation strategy for low-latency operation to reduce the computational cost per score [32]. The mean and variance of the residuals are updated using an exponentially weighted incremental method, expressed as:

$$\mu_r(t) = \lambda\mu_r(t-1) + (1-\lambda)r_t, \sigma_r^2(t) = \lambda\sigma_r^2(t-1) + (1-\lambda)(r_t - \mu_r(t))^2$$

where, λ represents the smoothing factor. The update equation avoids having to recalculate the entire window, thus maintaining real-time performance in high-throughput scenarios. The online update of the covariance matrix adopts an incremental method with shrinkage to maintain the numerical stability of the inverse covariance calculation and reduce computational overhead.

The alarm output process uses high-confidence alarms as input for automated response triggering, and includes an interpretable-evidence set to support source tracing and manual review. The evidence set includes the component contribution values of the current score (from the score contributions of the two components defined in equation (13), namely the normalized residual contribution and the scaled Mahalanobis distance contribution), historical trajectory segments of the trigger threshold T_t, and short-term peak flag bits. Alarm records are output to security information and event management (SIEM) in the form of structured logs for subsequent situational correlation. To reduce the false alarm rate, a short-term suppression window is included in the alarm stream. Repeated low-confidence alarms from the same source and destination within the suppression window are batch aggregated and reported to aggregate statistics to reduce the maintenance burden and improve alarm quality. Together, these mechanisms form a stable detection closed loop under the synergy of statistical baseline and prediction residual, so that the anomaly judgment achieves the design goal between accuracy and real-time performance.

3. EXPERIMENTAL DESIGN AND PERFORMANCE EVALUATION SCHEME

3.1 Introduction to Dataset and Experimental Environment

This study used the publicly available intrusion detection datasets CIC-IDS2017 and UNSW-NB15 as data sources for the experiments. CIC-IDS2017 provides raw pcap (Packet Capture Library) capture files for multiple days and flow-based feature CSVs. Flow features were exported from pcap using the CICFlowMeter tool, covering various real-world attack scenarios and normal business interactions. Official annotations include timestamps, quintuples, byte/packet statistics, and several derived statistics. The raw traffic data from UNSW-NB15 was generated using the IXIA PerfectStorm simulator and captured using tcpdump. It was then extracted using traffic analysis tools and labeled with attack categories and normal tags by experts. The data provides multi-dimensional feature fields to support classification and time-series studies. The raw files of the above datasets were directly obtained according to the official format, and their time synchronization and integrity were verified in the original time-series order during the initial processing stage of this study to ensure consistency in subsequent time-series modeling.

The same preprocessing and stream-level reconstruction process was applied to both datasets to ensure methodological consistency. For the pcap in CIC-IDS2017, CICFlowMeter was used to export bidirectional session-based flow records while retaining official fields. For the pcap in UNSW-NB15, the same flow reconstruction rules were used to generate comparable flow-level views. After session reconstruction, the generated flow records underwent field cleaning, removing entries with missing or anomalous markers and performing unified time zone and clock drift correction on the timestamps. Flow-level records were mapped to time-series input units using a fixed-window sliding aggregation strategy. The sliding window length and step size were determined on the validation set and uniformly applied to all datasets. The feature vector output by sliding aggregation included fields such as packet length statistics, arrival interval statistics, port distribution entropy, and protocol flag statistics. These fields directly corresponded to the officially exported fields of both datasets to ensure the consistency of feature semantics.

To tackle the issue of uneven labels, a method involving oversampling that keeps time order and adjusting loss weights was used to reduce bias during model training. That is to say, the oversampling step copies unusual segments locally while adding small noise changes to keep the time structure intact and preserve the original sequence order. The loss weights are set opposite to the frequency with which each class appears during training to change the extent to which each contributes to the gradient. To ensure that the training and testing steps can be repeated, all data splitting and preparation work is done using a fixed random number seed, which is explained in the experiment set-up notes.

The setup for running the experiments comprised computer hardware with a single-node CPU processor, such as Intel Xeon with at least 12 cores, having 128 GB of memory, and an NVIDIA Tesla V100 graphics card with 16 GB memory or something similar. Storage used fast NVMe SSDs to handle reading and writing large streaming files efficiently. On the software side, the operating system was Ubuntu 20.04, using Python 3.8 along with PyTorch 1.10 and Scikit-learn 1.x for building and training models. For pulling out traffic features, CICFlowMeter was utilized according to its official release

Table 3 Experimental Training and Testing Parameter Settings.

Setting Item	Parameter Value	Description	Remarks
Data Split	70% training, 15% validation, 15% test	Traffic sample split ratio	Ensures independence between training and testing data
Batch Size	128	Number of samples fed into the model per iteration	Affects training convergence speed and stability
Learning Rate	0.001	Initial learning rate of the model optimizer	Adam optimizer used
Number of Epochs	50	Number of full passes through the training data	Ensures sufficient model fitting
Anomaly Categories	Scanning, DoS, Injection, Lateral Movement, Hybrid	Used to test detection performance across attack types	Differentiates attack types for accuracy evaluation
Online Detection Mechanism	Asynchronous input	Real-time traffic processing mode	—
Model Input Dimension	32	Length of the feature vector	—
Optimizer	Adam	Parameter update method	Used together with the learning rate

version. The Pandas and Matplotlib tools handled data work and created the charts. The random seed stayed fixed at 42 throughout. During training, data was read in batches using streaming windows to decrease the extent to which the input/output operations affected training time.

3.2 Construction of Training and Detection Process

The data used in the experiment were partitioned into three data sets: training, validating, and testing, based on when they occurred. The first data set made up about 70% of the total amount of data, the validating data set made up about 15%, and the last set contained 15%. To maintain consistency between the amount of 'abnormal' and 'normal' traffic over time, it was necessary for all three sets of data to maintain their original ratios. For the training of the models, a batch-based approach was taken that allowed the modelling to take advantage of the temporary, structural, traffic features of each batch, which were encapsulated in the multi-dimensional window of time. The length of the windows and the size of the overlaps were calculated prior to processing the validating data set to enable the modelling team to capture all of the early-time features that were dependent on the time that they were recorded. The Adam adaptive optimizer was used to train the models, with a learning rate of 0.001 and a batch size of 128, for a total of 50 iterations (training rounds). After each round of training, the validating data set was used to calculate both the loss and the F1 score in order to monitor the model's performance. An early stopping mechanism was put in place to help prevent the overfitting of models.

The model was trained on user-defined scanning, DoS attacks, and injection as anomaly categories, as well as on hybrid attacks, in order to assess its applicability to multiple types of abnormal behaviour. To classify based on frequency counts, weights were assigned to the different categories through learning to avoid overfitting due to low data availability for the remaining categories during training. In addition to training, statistical baseline features were provided as input to the DNN along with temporal features, so that both could work together to provide a long-term prediction of the types and amounts of traffic. The differences between predictions and the statistical models (prediction residuals) were established as a reference in order to evaluate the performance of the DNN during training, and were also used to evaluate the predictions against the actual traffic conditions when the DNN was deployed for real-time anomaly detection. The online detection process used an asynchronous mechanism to take in real-time traffic and create feature vector sets from sliding windows and input them into the prediction model. Anomaly scores were produced from the prediction errors using a two-tier assessment process for continuous, low-latency anomaly detection while the DNN was operating in an online detection environment. All aspects of the training and detection stages were conducted within a consistent hardware environment to support stability and repeatability in batch processing, model inference, and asynchronous data stream processing operation.

To facilitate the identification of the impact of various settings on performance in subsequent experimental analysis, Table 3 lists key parameters such as training batch size, number of iterations, learning rate, anomaly type, and online detection mechanism.

Table 3 shows the core settings during the training process. The training batch size of 128 ensures the stability of gradient updates, the number of iterations is set to 50 so that the model can converge under different abnormal scenarios, the learning rate of 0.001 improves convergence efficiency while maintaining training stability, the input feature dimension of 32 combines statistical and temporal features to meet the requirements of the prediction model, the asynchronous online detection mechanism enables the system to maintain low latency processing capability in real traffic environments, and the selection of Adam optimizer helps to adaptively

Table 4 Comparison of experimental methods and variable configurations.

Model Name	Uses Statistical Baseline	Input Dimension	Number of Epochs	Key Parameters
Traditional Statistical Threshold Method	No	20	-	Fixed threshold range
SVM	No	50	100	Kernel: RBF (Radial Basis Function)
Random Forest (RF)	No	50	100	Number of trees: 200
Baseline-Only Model	Yes	30	–	KDE bandwidth; EWMA λ; Mahalanobis threshold
LSTM–Deep Prediction (No Baseline)	No	80	50	Learning rate: 0.001, Batch size: 128

adjust parameter updates, accelerate training convergence speed, and improve the robustness of anomaly detection.

3.3 Experimental Variables and Comparison Methods

To systematically evaluate the performance of the proposed statistical-prediction collaborative method (i.e., a deep prediction model fusing statistical baselines) in network traffic anomaly detection, multiple sets of comparative methods and variable controls were set up in the experiments. The comparative methods covered traditional statistical thresholding, Support Vector Machine (SVM), Random Forest (RF), statistical baseline-only models, and a version of the deep prediction model without fusing statistical baselines. The various methodologies in the experimentation phase were controlled for the equality and consistency of results by training and testing on the same data partitions, which allowed for a valid comparison between the methods. The experimental variables were varied within the experiments; these included types of prediction models, types of feature selection methods, and types of mechanisms for constructing thresholds. The types of prediction model types included LSTM, GRU (Gate Recurrent Unit), multi-layer Transform, as well as other types of prediction models. The inputs to the training were produced using statistical-based and time series-based features to produce predictions of future traffic. The selection of features was designed in several ways to ascertain the effect of different sets of features on increased detection accuracy and reduced false alarm rates; specifically, full-structured features, time series features, and statistical features only were used for each predictive model type among all combinations tested. The threshold differences were evaluated to determine how stable these types of threshold methods were when facing fluctuations in vehicular traffic patterns; fixed, dynamic, and residual-statistical fusions were tested as potential solutions for diurnal traffic fluctuations, to determine the consistency of decisions when using this method.

For each of the methods selected for comparison, the experiment was performed on the same hardware platform; all experiments had the same setup for batch processing, model predictions, and the ability to perform real-time data stream processing. Online anomaly detection was achieved using sliding-window-generated feature vectors as input into the model. Anomalies were given an anomalous score based on their prediction residual and statistical baseline. All experiments produced complete performance metric sets for all comparisons of the different methods and all combinations of variables. The common metrics produced include accuracy, recall, F1 score, false positives and false negatives. Table 4 summarizes the method settings and parameters for the different methods being compared.

Table 4 shows how the input feature size, training number of training rounds, and the main training parameter settings for each of the approaches used during the experiment were established. Utilizing these methods, the performance metrics were obtained for analysis and comparison.

4. RESULTS ANALYSIS

4.1 Anomaly Detection Performance Index Analysis

This experiment was conducted to determine how the integration of statistical and predictive methods affects the detection of anomalies in network. Via the experiment, systematic comparisons were made of various anomaly-detection approaches using a single combined dataset to determine their level of performance. The experiment determined the accuracy, recall, and F1 Scores of four detection techniques: traditional statistical thresholding, Support Vector Machines, Random Forest, and a deep learning model that integrates statistical predictions with residuals. All methods also received an identical amount of traffic during their training and testing procedures. The experiment included various types of anomalous traffic (scanning, DoS attacks injection attack, and lateral movement attacks), and provided a complete overview of the results for each of these attacks' respective detection methods. The full results of this experiment are shown in Figure 3.

As seen in Figure 3, the deep prediction model that included statistical baselines achieved an accuracy of 96.8%,

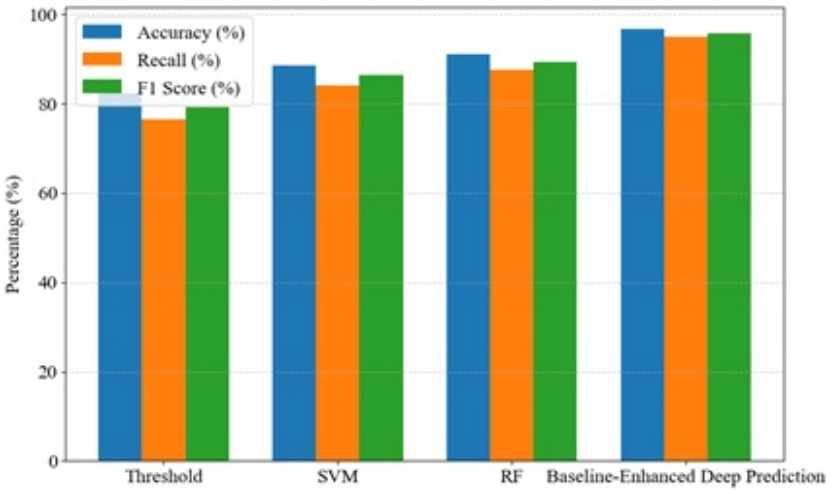


Figure 3 Comparison of anomaly detection performance.

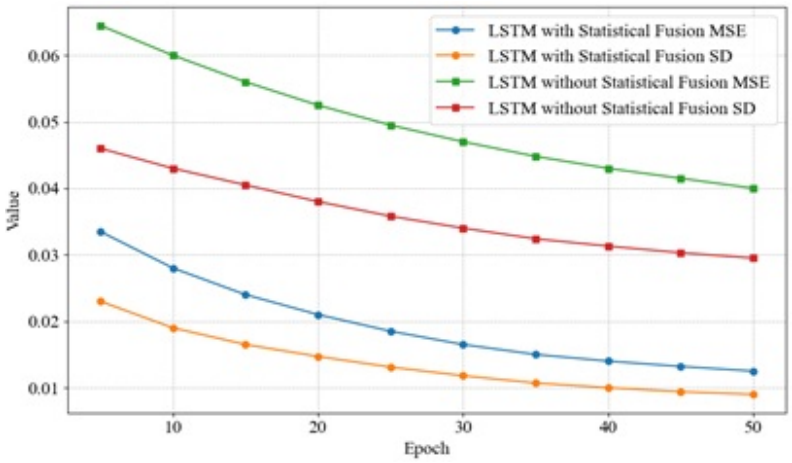


Figure 4 Predicted residual convergence and fitting ability.

a recall of 95.1%, and an F1 score of 95.9%. These scores indicate a 14.5% increase in accuracy and a 16.6% increase in the F1 score when compared to results obtained by traditional statistical thresholding approaches. Neither the SVM nor the random forest models that were based on either historical data features or local statistical features were able to detect abnormal fluctuations and were inferior to the deep prediction model proposed in this paper. The deep prediction model distinguishes between abnormal traffic and normal traffic because it integrates statistical baseline constraints with time-series learning through trend analysis, resulting in the detection of a greater number of residual outliers, better detection accuracy, and improved F1 scores. Furthermore, the data presented here indicates that the combination of predictive statistical collaboration maintains high accuracy in the detection of abnormal traffic patterns, thereby ensuring reliable error detection for multiple types of network communications.

4.2 Analysis of Residual Convergence and Fitting Ability of Prediction Model

In this experiment, the fitting ability and convergence characteristics of the deep prediction model for normal and abnormal traffic were studied by monitoring the residual changes during the training process. LSTM was used to perform time-series prediction on various types of network traffic sequences. In each training round, the residuals between the predicted and actual values were recorded, and the mean squared error and residual standard deviation were calculated to indicate prediction consistency. The performance differences between models using and those not using statistical baseline features, were compared during training, and the impact of different training window lengths on the residual convergence speed was analyzed. Traffic data over continuous time periods were collected, and the changes in residuals with training rounds were visualized to observe the model's fitting performance and outlier identification ability at different stages. The results are shown in Figure 4.

Figure 4 shows that the mean square error (MSE) of the LSTM-statistical fusion model decreased continuously from 0.0335 after 5 training epochs to 0.0125 after 50 epochs, while the MSE of the non-fusion model decreased only from 0.0645 to 0.0400. This indicates that statistical-based features create a robust basis of normal behaviour on which to build a prediction model that can quickly learn to differentiate between normal variability and abnormal deviation early in the modelling process, thus enabling rapid model convergence, and suppressing overfitting. The

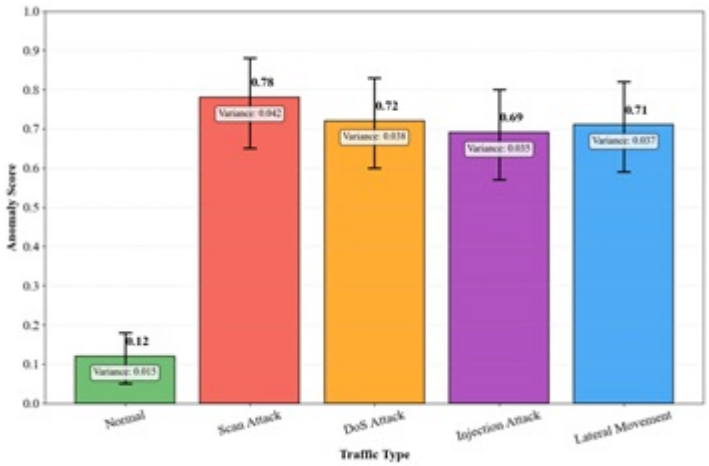


Figure 5 Comparison of abnormal score distribution.

standard deviation (SD) of the errors associated with the fusion prediction model is lower than that of the non-fusion prediction model, which again suggests that the inclusion of statistical features in the prediction process allows greater compression (less dispersive) as well as providing a better fit to normal traffic patterns. The results illustrate the synergy between statistical methods and indicate that deep time series modelling improves prediction consistency and increases the generalizability of the model.

4.3 Discrimination Analysis of Abnormal Score Distribution

To test whether or not the residual-driven system was able to effectively distinguish between normal and abnormal or anomalous traffic patterns, an experiment was performed whereby the processed traffic data was used to produce scores of anomalies that could be used as indicators of abnormal behaviour on the network, and a statistical analysis was performed on the resulting distribution of the anomaly scores for different categories of traffic samples. The statistical analysis involved comparing the distributions of the scores of normal traffic and each of the types of attack traffic to determine how they differed in terms of the mean, variance, and maximum and minimum value ranges of each type of traffic. In addition to this statistic analysis, the experiment also tracked the change in the anomaly scores associated with each of the various attacks when residual amplification occurred, and showed how well the model was able to discriminate between anomalies in multiple scenarios. The results are presented in Figure 5.

Figure 5 demonstrates that there are small fluctuations in the residuals when using the average anomaly score of 0.12 from normal traffic data (Variance = 0.015). Compared with normal traffic, scanning attack anomalies produce an average anomaly score of 0.78, while denial-of-service attack scores are an average of 0.72, injection attack scores are 0.67, lateral shift scores are 0.69, and so forth. In each case, these increase the residuals significantly compared to normal traffic, thus yielding higher average values for anomaly scoring. Fluctuations in the average scores for the types of attacks compared with normal traffic show that more attacks were observed over different time periods, producing an increased amount of residual feedback to the model's prediction that an anomaly will occur. Generally speaking, the higher variance is the result of the cumulative impact created by the residual scores of each type of attack compared with those in the statistical baseline when it was predicted that an anomaly would occur. The results also show that the anomaly scoring methodology can distinguish the characteristics of different forms of anomalies as they relate to their respective traffic sources. Thus, this provides an accurate means of detecting an anomaly within various types of network traffic. Moreover, it can be used a basis for developing new models that can improve the identification of network anomalies.

4.4 Optimization Analysis of False Alarm Rate and False Negative Alarm Rate

The experiment was conducted to assess the efficacy of the deep prediction model using the combined statistical baseline for identifying both false positive and false negative results in a realistic network environment. After training the model, multiple test runs were performed. For this testing process, anomalies detected on both "Normal" and "Attack" traffic streams were compared to determine the FPR (false positive rates) and FNR (false negative rates) based on differences in threshold settings. With the residual-driven scoring method combined with the statistical interval fusion judgment, the study measured the improvement in the system's ability to detect anomalies, indicated by low false alarm rates and low false negative rates across different traffic patterns. The findings of this testing are depicted in Figure 6.

As indicated in Figure 6, with an increase in the threshold, there is a gradual decrease in both the false alarm rate (from 0.12 to 0.02) and false negative rate (from 0.22 to 0.11). This suggests that, in comparison to low thresholds, higher thresholds result in fewer false judgements regarding normal traffic. This is mainly because the residual was restricted to be within the normal fluctuation range of data so that the statistical interval fusion judgement mechanism ignores

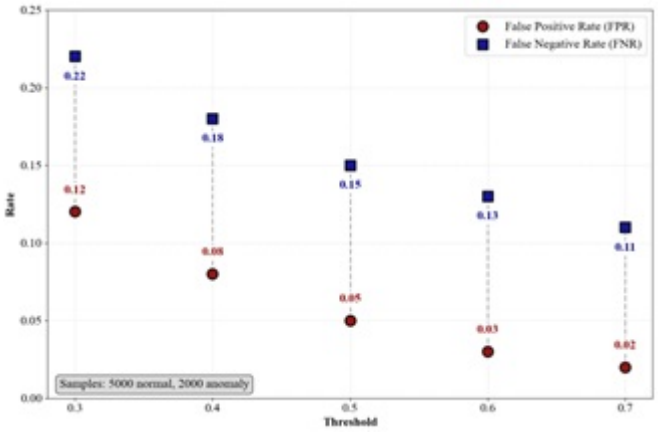


Figure 6 Comparison of false alarm rate and false negative rate.

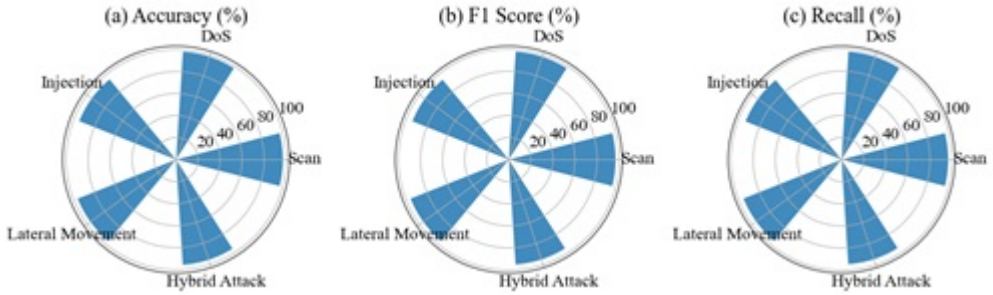


Figure 7 Detection performance for different attack types. Figure 7(a). Accuracy under different attack types. Figure 7(b). F1 Scores under different attack types. Figure 7(c). Recall rates under different attack types.

small errors or deviations. The decline in the number of false negatives indicates the extent to which the predictive model is sensitive to abnormal patterns This means that the predictive model retains its ability to find most of the abnormal behaviours despite increased thresholds. There are firm indications that the optimization of the threshold and statistical/predictive fusion mechanism leads to significant control over false alarms while concurrently maintaining low levels of false negatives.

4.5 Detection Performance Analysis under Different Attack Types

During this experiment, detection tests were conducted for different forms of attack traffic that the networask w exposed to (Scanning, DoS (Denial of Service), Injection, Lateral Movement, Hybrids). The experiment was designed to compare the performance of each type of attack within the model through its prediction residual and anomaly scores. The residual-driven anomaly scoring system produced detection results that indicated how accurately, proficiently and recall-wise each type of attack was detected by the different detection models. By controlling the input features and training iterations used in the experiment, it was verified that changes in the model's identified response levels between different attack scenarios indicate the method's ability to adapt to different types of threats. The results are illustrated in Figure 7.

Figure 7a displays the accuracy: 97.1% for Scanning Attacks, 98.3% for DoS, 94.7% for Injection, 96.2% for Lateral Movement, and 95.8% for Mixed Attacks. The accuracy scores indicate that the Predictive Model can identify true deviations from the expected normal traffic patterns, with a high level of accuracy, even in the presence of background noise or anomalous traffic. The model's excellent performance was indicated by the high level of accuracy produced during instances of significant traffic mutation, with Scanning Attacks and DoS.Figure 7b shows the F1 score was 96.8% for Scanning, 98.0% for DoS, 94.0% for Injection, 95.8% for Lateral Movement, and 95.2% for Mixed. The results showed that the model was able to achieve optimal combined performance between Accuracy and Recall, regardless of the type of Attack. The high performance of the model is a result of the residual scoring approach and Dynamic Thresholding that enables high-performance response to exceptional Weak Signal Anomalies, without producing a high number of False Positive Alarms. These capabilities contribute to the overall stability of the Predictive Model in handling various Attacks at the same time.Figure 7(c), displays Recall Rates which are 96.5% for scanning, 97.8% for DoS, 93.4% for injection, 95.4% lateral movement and 94.7% hybrid attacks. These metrics clearly indicate that the anomaly-detection model has high performance. This is due to the synergistic effect of multidimensional feature joint analysis and time series predictions of both time and event-based information which increase the model's sensitivity to attacks that are difficult to detect and result in only small disturbances at the source of the events (time). The results of this study indicate that this method provides excellent and balanced performance onattacks of semantic complexity at all levels.

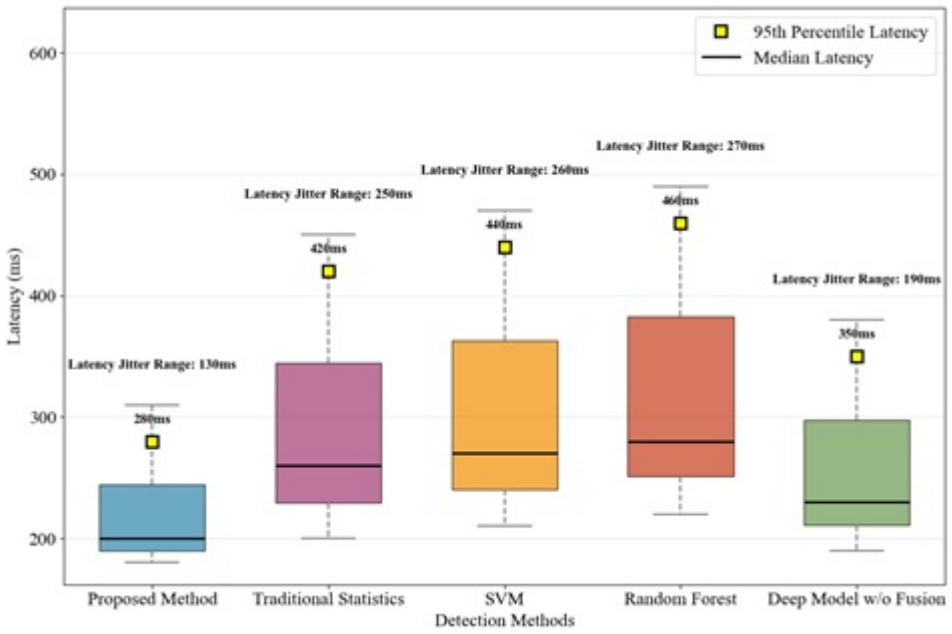


Figure 8 System detection delay distribution.

4.6 Real-time Detection Latency and System Performance Analysis

To determine how the system behaves in real time within a given real-world setting, The proposed approach was tested against baseline algorithms such as statistical thresholding, SVM, random forests and a non-statistical fusion deep-learning model. All tests were conducted on one set of network traffic, and the evaluation focused on the end-to-end latency of all tested algorithms. In addition to measuring average latency, the study also derived the various distribution statistics for the multiple rounds of testing by calculating the minimum, maximum, mean and 95th percentiles of latencies. The comparison allowed the study to examine how some of these algorithms performed at high utilizations and under abnormal circumstances, as shown in Figure 8.

Figure 8 shows that compared to other comparison methods, this method achieves a median delay of 200 milliseconds.The median latency jitter, which is the average time the model can respond within a range of time from when it was first invoked to when the user's request is fully processed, was only 130 milliseconds with the proposed method. This result indicates that the efficient feature extraction capabilities of the collaborative nature of the statistical-prediction model, combined with the residual-driven scoring strategy used in the method, have reduced redundant calculations at the model inference stage. The adaptive threshold and the asynchronous data input method help to mitigate the effects on the model's responsiveness to sudden surges of network traffic.

4.7 Analysis of the Improvement of Comprehensive Threat Identification Capabilities

To verify the comprehensive identification capability of the proposed method in complex threat scenarios, experiments were conducted in a multi-type attack traffic combination environment, including concurrent attacks such as scanning, DoS, injection, and lateral movement. The accuracy, recall, and F1 score of the proposed method were compared with those of traditional statistical thresholding, SVM, random forest, and a deep model without statistical fusion under different attack types. Also, the stability and sensitivity of each method in multi-attack scenarios were analyzed. The results show that the accuracy and F1 score of the proposed method remain within a high range under combined threat conditions, demonstrating the crucial role of the statistical-prediction synergy mechanism in complex scenarios. The experimental results are shown in Figure 9.

As shown in Figure 9(a), the results achieved using the methods proposed in this paper are: scanning, 97.1%; Denial of Service, 98.3%; injection, 94.7%; lateral movement, 96.2%; mixed, 95.8%. The superior overall performance of the proposed methods as compared to the other methods is due to the use of statistical baselines that create a stable characterization of the normal behaviour range. Also, the deep prediction module allows for the amplification of sudden anomalies in time series while providing a stable way to maintain high discrimination accuracy, under both statistical and dynamic constraints.

Figure 9(b) shows the changes in recall. The recall performance of various attacks is consistent with the accuracy trend. Among them, the recall of the scan detection method in this paper is 96.5%, the recall of DoS detection is 97.8%, the recall of injection detection is 93.4%, the recall of lateral movement detection is 95.4%, and the recall of hybrid attack is 94.7%. These results are produced by the joint deviation of anomalous behavior in density distribution and sequence morphology. The statistical baseline blocks the noise fluctuations in the low drift region, and LSTM generates a sensitive response to anomalous segments in sequence prediction, so that real attack events remain stable and prominent in the scoring space.

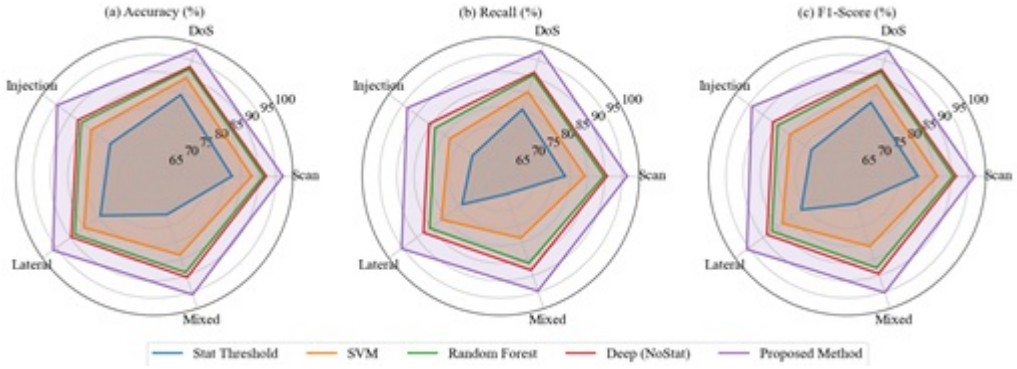


Figure 9 Overall threat identification performance comparison. Figure 9(a). Accuracy comparison of methods under different attack types. Figure 9(b). Recall comparison of methods under different attack types. Figure 9(c). F1 Score comparison of methods under different attack types.

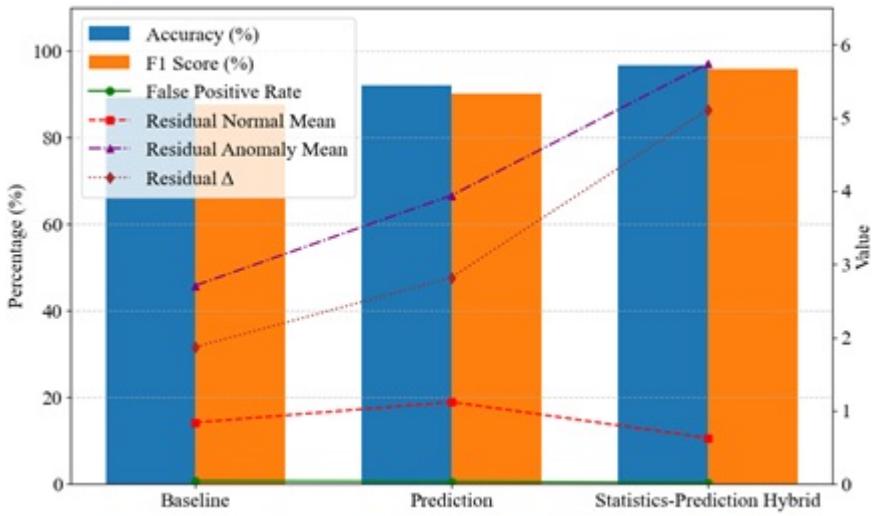


Figure 10 Comparison of ablation performance of statistical-predictive synergistic mechanisms.

Figure 9(c) shows the F1 scores for various types of attacks: 96.8% for scanning, 98.0% for DoS, 94.0% for injection, 95.8% for lateral movement, and 95.2% for mixed attacks. This data demonstrates that the method maintains a stable balance between precision and recall. The synergistic mechanism of statistical features and depth prediction residuals suppress any noise disturbances, ensuring the stable separation of attack fragments, thus achieving a consistent F1 advantage when facing various threat types, and resulting in a significant improvement in performance across scenarios.

4.8 Analysis of the Effectiveness of the Synergistic Mechanism in Ablation Experiments

To verify the synergistic effect of statistical structure modeling and time-series prediction modules, a control model containing only statistical baselines, a deep prediction model (without statistical baseline fusion) and an experimental model fusing both, were constructed. The accuracy, F1 score, false alarm rate, and residual discrimination of different structures under the same traffic conditions were compared to reveal the interactive effects of steady-state statistical constraints and time-series dependency modeling in the residual space, and to examine the mechanism by which the synergistic strategy contributes to the formation of anomaly boundaries. The experiment ran the three models on multiple batches of mixed normal and anomaly traffic, and performance metrics were recorded. The results for performance distribution and residual discrimination are shown in Figure 10.

In Figure 10, the control model that had only a statistical baseline achieved an accuracy of 89.4 % and an F1 score of 87.6 %, with a false alarm rate of 0.041. This is mainly because the statistical baseline relies solely on distribution offsets to establish thresholds, lacking the dynamic fitting of time-series trends. Consequently, sudden but statistically unrecorded outliers are not adequately represented in terms of offset. The deep prediction model achieved an accuracy of 92.1 % and an F1 score of 90.3 %. The residual discrimination overlapped significantly between normal and outlier intervals. This is because the prediction model lacks the constraint of a steady-state statistical center, causing input distribution fluctuations to amplify short-term noise, and resulting in discrete expansion of the residuals within the normal range. The experimental model achieved 96.8 % accuracy, 95.9 % F1 score, and 0.02 false alarm rate. The model's residuals had a unique pattern due to the limited range of the predicted input because of the center point of the steady state residence, which, when predicted, leads to significant residual deviations from the historic averages

in the time-series prediction, the model was more sensitive to deviations from the forecasted trend.The experimental results confirm that the collaborative method can effectively reduce the variability of the residuals while simultaneously amplifying or extending the outlier offsets, resulting in better detection ability, even within a complex traffic environment.

5. CONCLUSION

The collaborative detection model proposed in this study utilizes statistical characteristics and temporal dependencies of network data to detect anomalies associated with a network's traffic. By applying a combination of multi-scale feature modeling, the statistical constraints of normal network activity as a baseline for analysis, and the use of deep learning techniques to find the residuals from the predicted data, the study developed a robust mechanism for the detection of such anomalies. Experimental results indicate that the model is successful in training on a wide variety of traffic pattern categories, demonstrating consistently good performance. The mean squared error (MSE) of the training prediction model decreased from 0.0335 to 0.0125 over the course of the training period, demonstrating that the use of this collaborative model provides advantages in fitting complex traffic patterns. Accuracy of detection is at 96.8% and F1 score is at 95.9%, indicating that the system's use of both the residual scoring system as well as the dynamic thresholds can effectively distinguish normal from abnormal data. The median end-to-end latency for real-time evaluation of the collaborative detection model is at 200 milliseconds. The investigation demonstrates that there are significant synergies between statistical baselines and temporal mappings of the prediction model that enable the successful separation of anomalous offsets within a unified residual space. Fusion designs provide increased robustness in the detection process and enable the model to maintain performance across a range of threat scenarios, thus demonstrating their utility in protecting critical network infrastructure against attacks.

REFERENCES

1. Ndibe, OS (2025). AI-driven forensic systems for real-time anomaly detection and threat mitigation in cybersecurity infrastructures. *International Journal of Research Publication and Reviews, 6(*5), 389–411.
2. Edozie, E., Shuaibu, AN, Sadiq, BO, & John, UK (2025). Artificial intelligence advances in anomaly detection for telecom networks. *Artificial Intelligence Review, 58*, 100.
3. Bingjie,Y., Huifeng, Y., Chenjun, S., Zhi, Z. &Jinghang, F. (2021). Research on Network Security Collaborative Defense Technology Based on Swarm Intelligence and Big Data Network Security. *International Journal of Engineering Intelligent Systems*, 29(6), 379-386.
4. Sivakumar, J., Salman, NR, Salman, FR, Salimova, HR, & Ghimire, E. (2025). AI-driven cyber threat detection: Enhancing security through intelligent engineering systems. *Journal of Information Systems Engineering and Management, 10*(19s), 790–798.
5. Ke, H. (2025). Algorithm for Detection and Defense of Neural Network Technology Based on Neural Network and Multimedia. *International Journal of Engineering Intelligent Systems*, 33(4), 377-384.
6. Huang, H., Wang, P., Pei, J., Wang, J., Alexanian, S., &Niyato, D. (2025). Deep learning advancements in anomaly detection: A comprehensive survey. *IEEE Internet of Things Journal, 12*(21), 44318–44342.
7. Zheng, Y., Koh, HY, Jin, M., Chi, L., Phan, KT, Pan, S., ... & Xiang, W. (2024). Correlation-aware spatial–temporal graph learning for multivariate time-series anomaly detection. *IEEE Transactions on Neural Networks and Learning Systems, 35*(9), 11802–11816.
8. Lu, T., Wang, L., & Zhao, X. (2023). Review of anomaly detection algorithms for data streams. *Applied Sciences, 13*(10), 6353.
9. Fosiæ, I., Žagar, D., Grgiæ, K., & Križanoviæ, V. (2023). Anomaly detection in NetFlow network traffic using supervised machine learning algorithms. *Journal of Industrial Information Integration, 33*, 100466.
10. Kumari, S., Prabha, C., Karim, A., Hassan, MM, & Azam, S. (2024). A comprehensive investigation of anomaly detection methods in deep learning and machine learning: 2019–2023. *IET Information Security, 2024*(1), 8821891.
11. Ness, S., Eswarakrishnan, V., Sridharan, H., Shinde, V., Janapareddy, NVP, &Dhanawat, V. (2025). Anomaly detection in network traffic using advanced machine learning techniques. *IEEE Access, 13*, 16133–16149.
12. Ness, S. (2025). Securing networks against adversarial Domain Name System tunneling attacks using hybrid neural networks. *IEEE Access, 13*, 46697–46709.
13. Zhan, S., Huang, L., Luo, G., Zheng, S., Gao, Z., & Chao, HC (2025). A review on federated learning architectures for privacy-preserving AI: Lightweight and secure cloud–edge–end collaboration. *Electronics, 14*(13), 2512.
14. Torabi, H., Mirtaheri, SL, & Greco, S. (2023). Practical autoencoder-based anomaly detection by using vector reconstruction error. *Cybersecurity, 6*(1), 1.
15. Bilot, T., El Madhoun, N., Al Agha, K., & Zouaoui, A. (2023). Graph neural networks for intrusion detection: A survey. *IEEE Access, 11*, 49114–49139.
16. Lent, DMB, da Silva Ruffo, VG, Carvalho, LF, Lloret, J., Rodrigues, JJ, & Proença, ML (2024). An unsupervised generative adversarial network system to detect DDoS attacks in SDN. *IEEE Access, 12*, 70690–70706.
17. Ding, H., Sun, Y., Huang, N., Shen, Z., & Cui, X. (2024). TMG-GAN: Generative adversarial networks-based imbalanced learning for network intrusion detection. *IEEE Transactions on Information Forensics and Security, 19*, 1156–1167.
18. Marino, DL, Wickramasinghe, CS, Rieger, C., & Manic, M. (2025). Self-supervised and interpretable anomaly detection using network transformers. *IEEE Transactions on Industrial Informatics, 21*(5), 4250–4261.
19. Yang, Y., Tang, X., Liu, Z., Cheng, J., Fang, H., & Zhang, C. (2025). Diff-IDS: A network intrusion detection model based on diffusion model for imbalanced data samples. *Computers, Materials & Continua, 82*(3), 4389–4408.
20. Alrayes, FS, Zakariah, M., Driss, M., &Boulila, W. (2023). Deep neural decision forest (DNDF): A novel approach for enhancing intrusion detection systems in network traffic analysis. *Sensors, 23*(20), 8362.
21. Rotibi, A. O., Saxena, N., Burnap, P., & Read, C. (2025). System-level operational cyber risks identification in industrial control systems. *Cyber-Physical Systems*, 11(3), 265–296.

22. Isah, A., Shin, H., Oh, S., Oh, S., Aliyu, I., Um, TW, & Kim, J. (2023). Digital twins temporal dependencies-based on time series using multivariate long short-term memory. *Electronics, 12*(19), 4187.
23. Ahsan, M., Khusna, H., Wibawati, & Lee, MH (2023). Support vector data description with kernel density estimation (SVDD-KDE) control chart for network intrusion monitoring. *Scientific Reports, 13*, 19149.
24. Wawrowski, Ł., Białas, A., Kajzer, A., Kozłowski, A., Kurianowicz, R., Sikora, M., ... & Michalak, M. (2023). Anomaly detection module for network traffic monitoring in public institutions. *Sensors, 23*(6), 2974.
25. Frehner, R., Wu, K., Sim, A., Kim, J., & Stockinger, K. (2024). Detecting anomalies in time series using kernel density approaches. *IEEE Access, 12*, 33420–33439.
26. Landauer, M., Skopik, F., Stojanoviæ, B., Flatscher, A., & Ullrich, T. (2025). A review of time-series analysis for cyber security analytics: From intrusion detection to attack prediction. *International Journal of Information Security, 24*, 3.
27. Senthilraja, P., Palaniappan, K., Duraipandi, B., & Balasubramanian, UM (2024). Dynamic behavioral profiling for anomaly detection in software-defined IoT networks: A machine learning approach. *Peer-to-Peer Networking and Applications, 17*, 2450–2469.
28. Lakshmi, M., Rajavikram, G., Dattatreya, V., Jyothi, BS, Patil, S., &Bhavsingh, M. (2023). Evaluating the Isolation Forest method for anomaly detection in software-defined networking security. *Journal of Electrical Systems, 19*(4), 279–297.
29. Psychogyios, K., Papadakis, A., Bourou, S., Nikolaou, N., Maniatis, A., & Zahariadis, T. (2024). Deep learning for intrusion detection systems (IDSs) in time series data. *Future Internet, 16*(3), 73.
30. Altulaihan, E., Almaiah, MA, &Aljughaiman, A. (2024). Anomaly detection IDS for detecting DoS attacks in IoT networks based on machine learning algorithms. *Sensors, 24*(2), 713.
31. Talukder, MA, Sharmin, S., Uddin, MA, Islam, MM, & Aryal, S. (2024). MLSTL-WSN: Machine learning-based intrusion detection using SMOTETomek in WSNs. *International Journal of Information Security, 23*, 2139–2158.
32. Jemili, F., Jouini, K., &Korbaa, O. (2025). Intrusion detection based on concept drift detection and online incremental learning. *International Journal of Pervasive Computing and Communications, 21*(1), 81–115.