Discover Internet of Things

RESEARCH Open Access

A feature-level ensemble machine learning approach for attack detection in IoT networks

Firoz Khan[1], B. S. Sunil Kumar[1] and Sangeeta Sangani[2*]

*Correspondence:
Sangeeta Sangani
sangeeta.sangani@manipal.edu
[1]Department of Information Science and Engineering, GM Institute of Technology, Visvesvaraya Technological University, Davangere, Belagavi, Karnataka 590018, India
[2]Department of Information Technology, Manipal Institute of Technology Bengaluru, Manipal Academy of Higher Education, Manipal 576104, Karnataka, India

Abstract

The rapid adoption of Internet-of-Things (IoT) technologies in smart cities has enabled efficient data exchange among interconnected devices, enhancing automation and real-time decision-making. However, the proliferation of IoT applications and devices has also introduced significant cybersecurity risks, including Denial-of-Service (DoS), Distributed-DoS (DDoS), phishing, and spoofing attacks. Traditional AI-based intrusion detection models often struggle to maintain high accuracy over time due to Concept Drift (CD) and Class Imbalance (CI), which hinders their ability to detect evolving threats effectively. To address these limitations, this study proposes a feature-level ensemble machine learning approach called Weight-Optimized Extreme Gradient Boosting (WO-XGB). The proposed model incorporates a dynamic weight adjustment mechanism to mitigate CD and utilizes a reweighting strategy to handle CI, while integrating k-fold Cross-Validation (CV) to enhance generalization and prevent overfitting. WO-XGB was rigorously evaluated using two benchmark IoT intrusion detection datasets, i.e., Edge-IIoTset and CICIoT2023. The model achieved better results, with 99.98% accuracy on the Edge-IIoT set and 99.81% on CICIoT2023, outperforming several state-of-the-art ML and DL models. The experimental results demonstrate that WO-XGB is not only highly accurate but also resilient to evolving attack behaviors. In conclusion, WO-XGB offers a robust and adaptable solution for intrusion detection in IoT environments, effectively addressing critical challenges posed by CD and CI.

Article Highlights

- A new model called WO-XGB is presented, which improves the detection of cyber threats in smart devices by adapting to changing attack patterns.
- The method balances uneven data, avoiding bias toward common threats while still spotting rare attacks.
- Outperforms other approaches on two major datasets, showing strong accuracy and reliability in real-world tests.

Keywords IoT security, Intrusion detection, Concept drift, Class imbalance, Machine learning, WO-XGB

1 Introduction

By facilitating the connection of physical devices with digital networks, the Internet of Things (IoT) has transformed numerous fields, including transportation, smart cities, healthcare, and electronic manufacturing. IoT devices, such as sensors, smart meters, wearables, and industrial controllers, continuously generate and transmit vast amounts of data across distributed networks [1]. These data streams are processed at both the edge layer, where real-time decisions are made, and the cloud layer, which provides storage and advanced analytics [2]. The automation of operations, decision-making processes, and efficiency in operations have all been greatly enhanced by integrating IoTs with cloud and edge computing. Nonetheless, the extensive integration of IoT has concurrently presented new security weaknesses, positioning IoT systems as potential targets for cyberattacks. Numerous threats/attacks have the capability of compromising data availability, confidentiality, and integrity of IoT applications and devices [3, 4]. Some of the threats/attacks include Distributed Denial-of-Service (DDoS) and DoS attacks, which overwhelm IoT networks by flooding IoT devices with high traffic, leading to system failures and service disruptions [5]. Also, Man-in-the-Middle (MiTM) attacks allow adversaries to interrupt and manipulate communication between IoT devices and servers [6]. Nevertheless, phishing attacks exploit social-engineering methods for tricking users to show their sensitive data, while spoofing attacks enable attackers to impersonate legitimate devices or users to gain unauthorized access [7]. These threats can originate at both the edge and cloud layers, making IoT security a critical challenge. The dynamic nature of these attacks, combined with the resource-constrained nature of IoT devices, makes real-time detection and mitigation particularly difficult.

In recent years, Artificial Intelligence (AI)-based solutions, particularly Machine-Learning (ML) and Deep-Learning (DL) algorithms, have shown better performance for detecting and preventing cyber-threats/cyber-attacks in IoT networks. These models evaluated large volumes of network traffic and system logs for identifying anomalous patterns associated with malicious activities. Traditional ML algorithms such as Support-Vector-Machines (SVM) [8], Random-Forest (RF) [9], and Gradient Boosting [9] have been widely used, along with DL architectures like Deep-Neural-Networks (DNNs) [10] and Recurrent-Neural-Networks (RNNs) [11]. These AI-driven approaches have demonstrated promising results in enhancing IoT security. However, despite their effectiveness, existing methods face two major challenges: Concept Drift (CD) and Class Imbalance (CI) [12, 13]. CD occurs when attack patterns evolve, rendering previously trained models ineffective at detecting new threats. Most existing ML-based security frameworks assume that attack characteristics remain static, leading to performance degradation when attackers modify their tactics. The inability to dynamically adapt to new attack patterns results in higher false-negative rates, leaving IoT networks vulnerable. Additionally, most datasets used for training AI models suffer from CI, where attack instances are significantly underrepresented compared to normal traffic. Standard ML models trained on imbalanced datasets tend to favor the majority class (normal traffic) while failing to accurately detect minority classes (attacks). This leads to biased predictions and reduced detection rates for rare but critical cyber threats.

Furthermore, most existing models lack adaptability to evolving threat landscapes due to their static learning mechanisms, making them ineffective in handling CD, where statistical properties of attack data change over time [13, 16]. Without dynamic re-training

or adaptive learning strategies, these models experience performance degradation as new and unseen attack patterns emerge. Additionally, many conventional approaches, including classical ML and even some DL-based solutions, require extensive manual feature engineering and are computationally intensive, making them unsuitable for deployment in resource-constrained IoT environments [17, 20]. Studies have also shown that despite achieving high accuracy in controlled conditions, these models often overfit on specific datasets and fail to generalize well across different IoT scenarios and datasets [14, 19]. Moreover, hyperparameter tuning in many existing solutions is either overlooked or conducted manually, which further limits their scalability and robustness [18, 23]. The absence of integrated strategies to simultaneously implement CI and CD in an automated and computationally efficient manner remains a critical gap in current IoT intrusion detection research.

To address these limitations, this work proposes a feature-level ensemble ML approach, called Weight-Optimized Extreme Gradient Boosting (WO-XGB), for attack detection in IoT environments. The proposed model effectively handles both CD and CI through a dynamic weight optimization mechanism and a class rebalancing strategy. By continuously adapting to evolving attack patterns and ensuring that minority attack instances are adequately represented during training, WO-XGB provides a more accurate and generalizable solution for securing IoT applications and their data in edge-cloud environments. The contribution of the work is as follows.

- This work introduces a novel WO-XGB approach that jointly addresses two critical challenges in IoT attack detection, CD and CI, which are often overlooked or handled separately in prior research. Unlike conventional ML/DL methods, WO-XGB dynamically adapts to evolving attack patterns and ensures improved detection of minority attack classes.
- The WO-XGB model incorporates a feature-level ensemble mechanism and applies a dynamic weight adjustment strategy to mitigate CD. For CI, the model uses a reweighting-based loss function that penalizes misclassification of underrepresented attack classes. The model also integrates k-fold cross-validation (CV) during training to enhance generalization and prevent overfitting.
- WO-XGB was rigorously evaluated on two recent and diverse benchmark datasets, Edge-IIoTset and CICIoT2023. It achieved 99.98% accuracy on the Edge-IIoT set and 99.81% accuracy on CICIoT2023, consistently outperforming existing state-of-the-art ML and DL models. The results validate the effectiveness of WO-XGB in detecting a wide range of IoT attack types under realistic class distributions and evolving threat scenarios.

The manuscript is structured as follows: Section II provides a literature survey, highlighting existing research in the field. Section III introduces the WO-XGB model, its architecture, and methodology. Section IV presents experimental results of the WO-XGB model and compares its performance with existing approaches. Finally, Section V concludes the study and discusses future research.

2 Literature survey

This section discusses ML and DL approaches presented for detecting attacks in IoT, edge, and cloud environments. Ferrag et al. [14] presented a dataset for identifying cyber-threats in Industrial IoT (IIoT) and IoT, called Edge-IIoTset. The dataset was constructed considering different protocols, edge/cloud settings, sensors, applications, and devices. The IoT data was collected considering more than 15 sensors, using which 14 attacks were identified and classified into DDoS/DoS, MiTM, malware, injection, and phishing attacks. The dataset consisted of multiple features, i.e., more than 1000 + features for predicting attacks. For evaluation of attack prediction, they used five ML methods, which included Deep Neural Network (DNN), K-Nearest Neighbor (KNN), Support Vector Machine (SVM), Random Forest (RF), and DT. Findings showed that DNN showed better accuracy for predicting different attacks (94.67%–15 class, 96.01%–6 class, and 99.99%–2 class). This work did not handle CI and CD. Pinto et al. [15] presented an IoT attack dataset using real-time IoT processes. This work considered 33 kinds of attacks using 105 devices. The 33 attacks were then classified into seven classes, which include Mirao, spoofing/phishing, brute force, web-based, recon, DoS, and DDoS attacks. For evaluation, five ML approaches were used, including AdaBoost (AB), Multilayer Perceptron (MLP), Logistic Regression (LR), RF, and DNN. Among all the models, RF showed better accuracy for attack prediction (99.16%–34 class, 99.43%–8 class, and 99.68%–2 class). This work did not handle CI and CD. Aburasain et al. [16] presented an Enhanced Black-Widow-Optimization with Hybrid-DL-Enabled-Intrusion-Detection (EBWO-HDLID) approach for detecting attacks that happen because of IoT devices in a smart-farming environment. In EBWO-HDLID, a Bald-Eagle-Searching (BES) approach was presented for selecting features for attack detection. For classifying attacks, the HDLID method was used. For hyperparameter tuning, the EBWO was used. Evaluations were conducted considering two datasets, i.e., Edge-IIoTset and ToN-IoT. During evaluation, the EBWO-HDLID achieved 98.81% average testing accuracy for ToN-IoT and 98.35% average testing accuracy for Edge-IIoTset. This work did not handle CI and CD.

a. Alshehri et al. [17], to handle the problem of CI during attack prediction in IIoT networks, presented Self-Attention based Deep-Convolution Neural-Network (SA-DCNN). The SA process helped in extracting features, and DCNN helped in the detection of attacks. Further, for removing redundant data from cross and intra-class samples, presented a two-step cleaning approach was presented. Further, for handling the underfitting problem, presented Mutual-Information (MI) based feature-filter approach was presented, which ranked different features, removing less important features for prediction. The SA-DCNN was evaluated using Edge-IIoTset and IoTID20 datasets, where it achieved 99.95% and 96.89% accuracy, respectively. This work did not CD problem, i.e., when attack patterns change. Khan et al. [18] presented an approach for detecting attacks in Internet of Medical Things (IoMT), which considered the CICIoT2023 dataset to build their model. In their work, they first preprocessed the dataset and used the Synthetic Minority Over-sampling Technique (SMOTE) for handling CI in the dataset. Further, for feature engineering, used the Pearson Correlation Coefficient (PCC). For the prediction and classification of attacks, used five ML approaches, which included MLP, DNN, LR, AB, and RF. Findings showed that RF achieved better accuracy (99.55%–2 class, 95.54%–8 class, and 96.32%–34 class).

This work did not CD problem. Tseng et al. [19], for attack detection, considered the CICIoT2023 dataset and presented seven DL methods, which included RNN, CNN, Long-Short Term-Memory (LSTM), DNN, and two ensemble models, CNN + LSTM and CNN + RNN. They considered all 46 features from the CICIoT2023 dataset for attack prediction. Findings show that the CNN + LSTM (99.48% accuracy-binary, 99.26% accuracy-multiclass) and DNN (99.56% accuracy-binary, 99.36% accuracy-multiclass) achieved better results for prediction when the neurons and layers were increased. Javeed et al. [20] presented a hybrid ensemble DL method that combined LSTM and Bi-Directional Gated-Recurrent Unit (Bi-GRU) for predicting attacks in smart-farming in an IoT-edge environment, where attacks were detected at the edge layer. For learning faster, they employed Truncated-Backpropagation Through-Time (TBPTT) for handling long sequences in data. Evaluations were done using Edge-IIoTset, CIC-IDS2018, and ToN-IoT dataset, where they achieved 98.32%, 99.82%, and 99.55%, respectively.

b. Elaziz et al. [21] presented an approach for detecting attacks called Convolution-Kolmogorov-Arnold Network (CKAN). In the CKAN approach, the MLP layer was replaced with the KAN layer in CNN to provide better performance. The CKAN included a feature extraction for extracting features, providing better prediction results. Evaluations were conducted considering three datasets: ToN-IoT, CICIoT2023, and NSL-KDD. Evaluations were conducted on two types of binary and multi-class, where the CKAN achieved better accuracy for ToN-IoT (99.93% for binary and 93.30% for multi), CICIoT2023 (99.22% for binary and 98.84% for multi) and NSL-KDD (98.71% for binary and 99.20 for multi). Almadhor et al. [22], used Federated Learning (FL) for training their proposed XGB model incorporated with Shapley-Additive-Explanation (SHAP) which helped in selecting features and provide feature interpretability respectively, for detecting DDoS attacks in IoT environment. Evaluations were conducted on the CICIoT2023 dataset, where 99.78% accuracy was achieved. Belachew et al. [23] presented an approach for detecting attacks in Software-Defined-Networking (SDN) IoT environment, which used ML with CV and hyperparameter tuning. The proposed model was deployed at the edge of the SDN-IoT environment. Four ML approaches were considered, i.e., Feed-Forward Neural-Network (FFNN), XGB, RF, and KNN. For selecting features, used RF feature-importance (RFFI). Evaluations were conducted considering CICIDS2017 and Edge-IIoTset, where XGB showed better performance, achieving 98.85% accuracy for binary and 99.73% accuracy for multi-class, considering CICIDS2017, and 99.99% accuracy for binary and multi-class, considering Edge-IIoTset. Chandnani et al. [24] presented an FL Multi-Layered DL approach which employed Physics-based Hyperparameter-Optimization, called FedRIME. For feature engineering used PCC, and for training the model on the server side presented an algorithm called Fed-MLDL. Evaluations were conducted on five datasets, which included CICIoT2023, CICIoT2022, IoT-2023, Edge-IIoTset, and ToN-IoT. Berguiga et al. [25] presented an approach for identifying attacks in the IoMT environment, presenting a hybrid ensemble model, which used CNN for extracting features and LSTM for prediction. For handling class imbalance used SMOTE. Evaluations were conducted considering Edge-IIoTset and IoTID20 datasets, where they achieved 99.94% and 99.27% accuracy, respectively. Table 1 presents a summary of the literature survey.

Table 1 Summary of literature survey

Ref	Model	Feature Selection	CI	CD	Dataset	Best Accuracy
[14]	DNN, KNN, SVM, RF, DT	No	No	No	Edge-IIoTset	DNN (99.99%)
[15]	AB, MLP, LR, RF, DNN	No	No	No	CICIoT2023	RF (99.68%)
[16]	EBWO-HDLID	Yes	No	No	Edge-IIoTset, ToN-IoT	98.81%-ToN-IoT, 98.35%-Edge-IIoTset
[17]	SA-DCNN	Yes	Yes	No	Edge-IIoTset, IoTID20	99.95%-Edge-IIoTset, 96.89%-IoTID20
[18]	MLP, DNN, LR, AB, RF	Yes	Yes	No	CICIoT2023	RF (99.55%)
[19]	RNN, CNN, LSTM, DNN CNN + LSTM, CNN + RNN	No	No	No	CICIoT2023	DNN (99.56%)
[20]	BiGRU + LSTM	No	No	No	Edge-IIoTset, CIC-IDS2018, ToN-IoT	Edge-IIoTset-98.32%, CIC-IDS2018-99.82%, ToN-IoT-99.55%
[21]	CKAN	Yes	No	No	ToN-IoT, CICIoT2023, NSL-KDD	ToN-IoT (99.93%), CICIoT2023 (99.22%) and NSL-KDD (98.71%)
[22]	XGB + SHAP	Yes	No	No	CICIoT2023	99.78%
[23]	RFFI + XGB	Yes	Yes	No	CICIDS2017, Edge-IIoTset	CICIDS2017- 98.85%, Edge-IIoTset- 99.73%
[24]	FedMLDL + FedRIME	Yes	Yes	No	CICIoT2023, CICIoT2022, IoT-2023, Edge-IIoTset, ToN-IoT	CICIoT2023 (99.3%), CICIoT2022 (98.7%), IoT-2023 (97.9%), Edge-IIoTset (98.6), ToN-IoT (98.1%)
[25]	CNN + LSTM	Yes	Yes	No	Edge-IIoTset, IoTID20	Edge-IIoTset (99.94%), IoTID20 (99.27%)

From the reviewed literature, it is evident that most existing IoT intrusion detection approaches based on ML and DL have focused primarily on static data modeling, with limited consideration for challenges such as CD and CI. While models proposed by Ferrag et al. [14], Pinto et al. [15], Aburasain et al. [16], and Tseng et al. [19] contribute to attack detection, they do not address the temporal dynamics of data, making them less effective in evolving environments. On the other hand, studies like those by Javeed et al. [20] and Elaziz et al. [21] fail to mitigate CI, often resulting in biased predictions skewed towards majority classes. Although techniques such as SMOTE have been employed by Alshehri et al. [17], Khan et al. [18], and Berguiga et al. [25] to alleviate CI, they still fall short in adapting to changing data distributions caused by CD. Moreover, while Belachew et al. [23] and Chandnani et al. [24] attempt to address CI, the absence of CD handling remains a key limitation. Therefore, there exists a clear research gap in developing an intrusion detection model that can simultaneously and effectively handle both CD and CI. The proposed WO-XGB model aims to fill this gap by introducing a dynamic and adaptive learning mechanism capable of maintaining high performance in the face of both data imbalance and evolving attack behaviors.

3 Methodology

This section introduces the WO-XGB architecture, preprocessing steps, followed by a detailed discussion of the WO-XGB model and its mechanisms for addressing CD and CI. Additionally, this work presents a k-fold feature-level splitting optimization technique within WO-XGB, which enhances the training process by improving model generalization and robustness.

3.1 Architecture

Figure 1 presents a security framework designed for protecting IoT application data in an edge-cloud environment. The framework shows how potential attacks happen, i.e., how attackers/adversaries exploit vulnerabilities in IoT devices/sensors/applications for gaining access to sensitive data. In this framework, three types of attacks have been considered, i.e., DDoS, spoofing, and phishing. The attacker can use any of the three types, which either target IoT devices or the edge layer, causing system failures and service disruptions. Figure 1 also presents a feature-level ensemble ML approach, called WO-XGB, which aims at detecting and predicting these attacks. The WO-XGB model processes IoT application data both at the edge and cloud layers, which helps it to analyze different patterns for identifying malicious activity.

Further, in Fig. 2, the complete flow of WO-XGB is presented for attack detection at the edge and cloud layers. The architecture first considers the data/dataset as input to train the model, which first goes through preprocessing. Further, the WO-XGB incorporates CD and CI handling with k-fold splitting optimization during training. During the training of WO-XGB, DT weights are adjusted dynamically, which helps in achieving better accuracy. Further, the CD approach helps in learning the changing attack patterns, while the CI approach prevents the WO-XGB model from being biased towards the majority attack or normal class, making it an accurate detection/prediction model. Moreover, the k-fold splitting optimization method provides efficient training without data refitting, making the model a generalizable model. After the complete process of

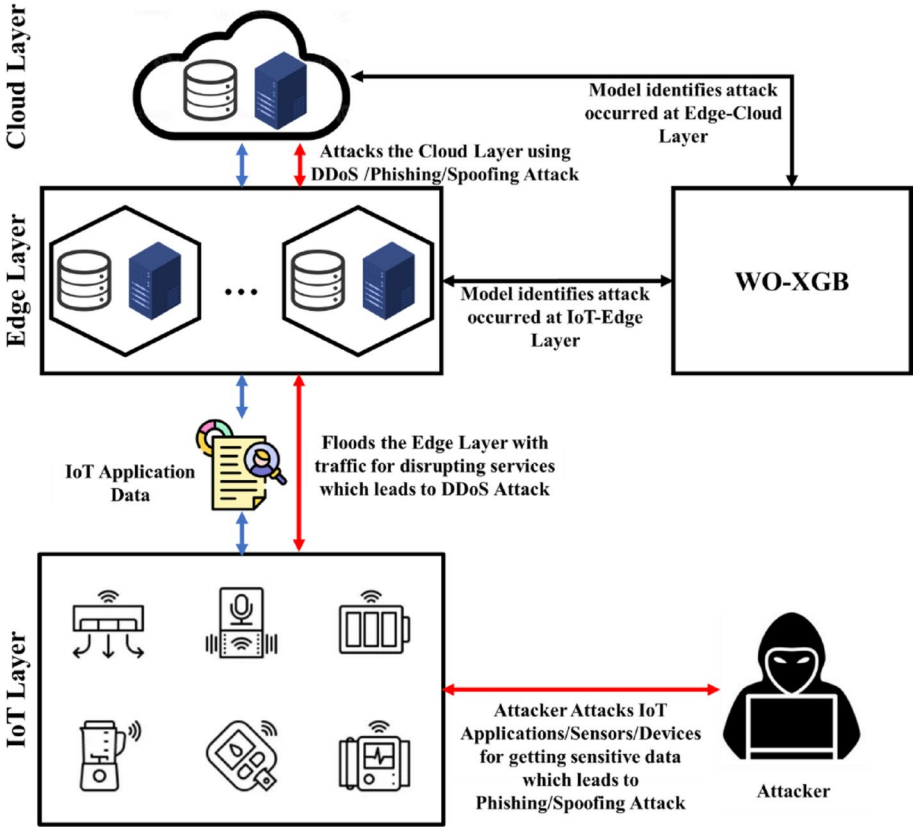

Fig. 1 Security Framework for IoT Applications Data in Edge-Cloud

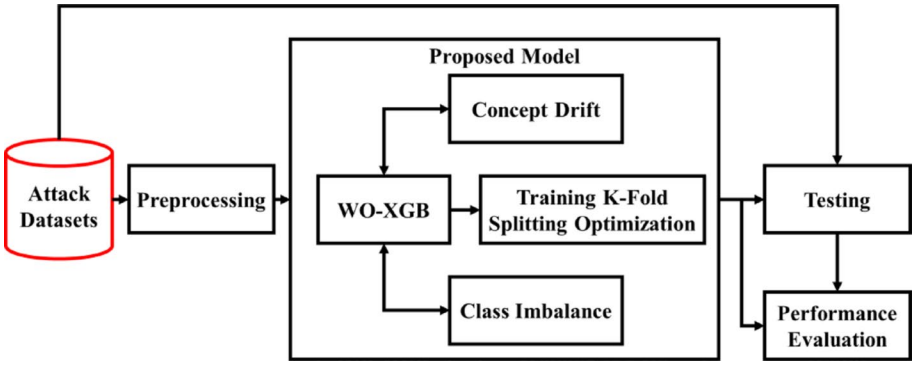

Fig. 2 WO-XGB Architecture

training, the WO-XGB test uses test data where evaluates the prediction using standard performance metrics.

3.2 Preprocessing

Before WO-XGB training, this work conducted a preprocessing on the two datasets considered for this work, i.e., Edge-IIoTset and CICIoT2023. The first step in preprocessing included examining missing or null values. From the analysis, it was found that in Edge-IIoTset, a total sample of 157,800 was there, which included 24,301 normal samples and 133,499 attack samples, where 157,586 samples had missing values. Further, in CICIoT2023, a total of 45,019,243 were there, which included 1,051,373 normal samples and 43,967,870, where 991 samples had missing values. Instances or samples containing missing values were either imputed using feature-wise median values. Further, for bringing all continuous features onto a comparable scale and accelerating the convergence of WO-XGB, normalization was applied. This work used Min-Max scaling, which transformed every numerical feature to the $[0,1]$ range, which preserved the shape of the original distribution while removing scale disparities. This step was important in both datasets because feature ranges can vary drastically because of the diverse nature of recorded network-traffic attributes. Further, the categorical variables, which included device IDs, attack labels, protocol types, and other variables, were encoded. For encoding, label-encoding was utilized for target-labels (normal or benign and attack class), while one-hot encoding was used for nominal categorical features to prevent the introduction of spurious ordinal relationships. Also, special attention was given for ensuring consistent encoding across training and testing splits. For evaluating WO-XGB model performance, similar to existing work [17, 25], both datasets were split into a training and testing ratio of 80:20, stratified split, maintaining the proportion of attack and normal classes. In the next section, the WO-XGB model is discussed in detail.

3.3 Weight optimized XGB

Consider the dataset for predicting attacks as presented in Eq. (1).

$$X_t = \{(x_1, y_1)\,,\ (x_2, y_2)\,, \ldots\ \ldots\ (x_t, y_t)\} \tag{1}$$

In Eq. (1), $x_1, x_2, \ldots, x_t$ denotes features present in a row in the dataset X_t and $y_1, y_2, \ldots, y_t$ denotes its respective true observed value at the time step t, i.e., $y_t = y_t = 0$ denotes no attack and $y_t = 1$ denotes attack.

$$L(\theta) = \sum_{t=1}^{T} l(y_t, \hat{y}_t) + \sum_{k=1}^{K} \Omega(f_k) \tag{2}$$

In Eq. (2), T denotes the total number of time steps (or training examples) in the dataset, y_t represents the true observed value at the time step t, $\widehat{y}_t$ is the predicted value for y_t generated by WO-XGB, using the input dataset features X_t, $l(y_t, \widehat{y}_t)$ is a loss function, k represents the index of DTs in the WO-XGB, K denotes total number of trees in the WO-XGB, f_k is k^{th} DT in WO-XGB which predicts the attack, $\Omega\ (f_k)$ is regularization parameter and $\sum_{k=1}^{K} \Omega\ (f_k)$ is sum of the regularization terms for all trees, penalizing complex models. Further, the regularization term $\Omega\ (f_k)$, which controls model complexity is evaluated as presented in Eq. (3).

$$\Omega\ (f_k) = \gamma\, T + \frac{1}{2}\lambda\, ||w||^2 \tag{3}$$

In Eq. (3), γ and λ are regularization terms, w denotes the weight. Further, each attack predictor f_k in DT using the dataset X_t is modelled using Eq. (4).

$$f_k(X_t) = \sum_{j=1}^{T} w_j(t)\, h_j(X_t) \tag{4}$$

In Eq. (4), $w_j(t)$ is the weight of the leaf node in the tree for time t, and $h_j(X_t)$ is the feature-based decision split. The WO-XGB model adjusts weight dynamically based on prediction errors using Eq. (5).

$$W_i(t) = \frac{\sum_{i=1} \overline{\overline{\varepsilon_i}}(t)}{\varepsilon_i(t)} \tag{5}$$

In Eq. (5), $W_i(t)$ denotes the weight assigned to WO-XGB model i for time t, $\overline{\overline{\varepsilon_i}}(t)$ denotes the mean error of WO-XGB model i over the past window of time, and $\varepsilon_i(t)$ denotes the error of model i. For effective weight calculation in the WO-XGB model and to prevent rapid fluctuations in weights and ensure stable learning, Eq. (6) is used.

$$W_i(t) = \frac{1}{m} \cdot \sum_{k=1}^{m} W_i(t-k) \tag{6}$$

In Eq. (6), m denotes the number of past time-steps used for effective learning, $W_i(t-k)$ denotes the weight of i at previous time steps. Instead of assigning a new weight at each time step abruptly, Eq. (6) averages the past m weight values to determine the new weight. This is useful to prevent overreacting to short-term variations. Further, for calculating the prediction error at time t, considering the weighted sum of predictions from the WO-XGB model, Eq. (7) is used.

$$e_{i,t} = \sum_{i=1} W_i(t) \cdot f_k(X_t) - f_k(t) \tag{7}$$

In Eq. (7), $e_{i,t}$ denotes prediction error at time t, $W_i(t)$ denotes the weight assigned to i at time t, $and\ f_k(X_t)$ denotes the output from each attack predictor f_k for X_t, and $f_k(t)$ denotes the actual observed value at time t. Further, for evaluating error for the

WO-XGB model i at the next time step $i+1$ for updating the error dynamically, Eq. (8) is presented.

$$e_{i+1,t} = \sum_{i=1} W_{i+1}(t) \cdot f_{i+1}(X_{i+1}) - f_k(t) \tag{8}$$

The Eq. (8) ensures that the model adapts to changing conditions and re-evaluates prediction error dynamically. The final prediction $\widehat{y_t}$ using the WO-XGB model is achieved using Eq. (9).

$$\hat{y}_t = \sum_{i=1} f_k(X_t) W_i(t) \tag{9}$$

Moreover, the loss for the final prediction $\widehat{y_t}$ using the WO-XGB model is evaluated using Eq. (10).

$$Loss = -\sum_{i=1}^{m} (\alpha y_t \log(\hat{y}_t) + (1-y_t) \log(1-\hat{y}_t)) \tag{10}$$

In Eq. (10), α denotes CI parameter. When α is more than 1, additional loss occurs when predicting 1 as 0. Conversely, if α is below 1, the emphasis of the loss function shifts towards correctly identifying data-streams labeled as 0. Further, for handling the CD and CI during attack detection and training, respectively, the next section presents which discusses how the WO-XGB handles these issues.

3.4 Handling concept drift and class imbalance in WO-XGB

Consider the dataset $X_t = \{(x_1, y_1), (x_2, y_2), \ldots \ldots (x_t, y_t)\}$ where $x_t \in \mathcal{X}$ and $y_t \in \mathcal{Y}$. Consider that the WO-XGB model predicts attacks for every time step as $i = m,\ m-1,\ \ ,\ m-N$, where $m = 1,2,\ldots N$, and let the attack be denoted as c. Consider a parameter β used for reducing weights dynamically using the WO-XGB model. As the attacks evolve or change over time step m, it is important to train the WO-XGB model using the evolving attacks. Hence, the WO-XGB incorporates CD handling using Algorithm 1.

Input Dataset X_t, WO-XGB attack prediction model i, different kinds of attacks c, tree removal parameter θ, dynamic weight reduction penalty parameter β, and tree-updating variable p
Output Set of DTs H_j in WO-XGB and leaf-node weights w_j in DTs
Step 1 **Start**
Step 2 Consider $m = 1$ and $w_j = 1$
Step 3 For every time step create a set of H_j using WO-XGB, i.e., $\mathcal{H} = \{H_j\}$
Step 4 **For** $i = 1$ *to* N **do**
Step 5 **For** $j = 1$ to m **do**
Step 6 **If** $H_j(x_t) \neq y_t$ and $i \; mod \; p \neq 0$ **then**
Step 7 $w_j = \beta w_j$
Step 8 **End If**
Step 9 **End For**
Step 10 **Predict attack using Eq. (4)**
Step 11 **If** $i \; mod \; p = 0$ **then**
Step 12 Normalize weights of DT using $w_j = \frac{w_j}{\sum_j w_j}$
Step 13 Remove trees having less weight than the dynamic parameter θ, i.e., using $\mathcal{H} = \overline{\{H_j | w_j < \theta\}}^{\mathcal{H}}$
Step 14 **If** $\hat{y}_t \neq y_t$ **then**
Step 15 $m = m + 1$
Step 16 Perform feature-based decision split $h_j(X_t)$
Step 17 $\mathcal{H} = H_j \cup h_j(X_t)$
Step 18 $w_j = 1$
Step 19 **End If**
Step 20 **End If**
Step 21 **For** $j = 1$ to m **do**
Step 22 **Create DTs for learning drift using** $H_j = (H_j, x_t, y_t)$
Step 23 **End for**
Step 24 **End for**
Step 25 **Stop**

Algorithm 1 Concept drift handling approach for WO-XGB

Algorithm 1 presents the CD handling approach for the WO-XGB model. The WO-XGB model adapts to evolving attack patterns by dynamically updating DTs and adjusting their weights. The algorithm begins by initializing the model with multiple DTs and setting the initial weight of each leaf node as $w_j = 1$. At each time step, the model creates a set of DTs H_j, using WO-XGB. For every DT in the set, if a tree's prediction for different features $H_j\,(x_t)$ does not match the true value y_t and the iteration index does not satisfy a periodic update condition, i.e., $i \, mod \, p \neq 0$, the tree-weight w_j is reduced by multiplying it with the parameter β, ensuring that poorly performing trees lose influence over time. The algorithm then predicts attacks using Eq. (4), which aggregates weighted DT outputs. If the iteration index satisfies the periodic update condition $i \, mod \, p = 0$, the model normalizes tree weights by dividing each weight by the sum of all tree weights. Trees with weights lower than a predefined threshold θ are removed, ensuring that the model does not retain ineffective predictors. Additionally, if the model's final prediction $\widehat{y}_t$ does not match the actual outcome y_t, the number of DTs is increased, and a new feature-based decision split is performed, incorporating recent data to improve future predictions. The algorithm further enhances adaptability by training new DTs for drift learning. This involves appending the most recent data points x_t, y_t to DTs, allowing the model to recognize emerging patterns. The process continues iteratively, enabling the WO-XGB model to dynamically adjust to shifting attack behaviors, maintain accurate predictions, and prevent model stagnation due to outdated DTs. By reducing the weights of outdated trees, removing ineffective predictors, and introducing new trees when needed, the WO-XGB model effectively handles CD, ensuring sustained performance in evolving environments. Further, for handling imbalance issues, the loss

function presented in Eq. (10) has been modified by introducing class-specific weights α_0 and α_1, where α_0 is the weight for the majority class and α_0 is the weight for the minority class. Hence, Eq. (10) is modified as Eq. (11).

$$Loss = -\sum_{i=1}^{m}(\alpha_1 y_t \log(\hat{y}_t) + \alpha_0 (1 - y_t) \log (1 - \hat{y}_t)) \tag{11}$$

In Eq. (11), $\alpha_1$is higher than α_0 to penalize misclassification of the minority class. Moreover, the class weights are determined dynamically using the inverse class frequency as presented in Eq. (12).

$$\alpha_{yt} = \frac{1}{P_{y_t}},\ y_t \in \{0,1\} \tag{12}$$

In Eq. (12), P_{y_t} denotes the proportion of attack and normal classes c in the dataset X_t. Further, weights of DTs are adjusted using the imbalance ratio. The weight formula presented in Eq. (5) in WO-XGB is modified as presented in Eq. (13).

$$W_i(t) = \frac{\sum_{i=1} \left(\overline{\overline{\varepsilon_i}}(t) \cdot \alpha_{y_t}\right)}{\varepsilon_i(t)} \tag{13}$$

In Eq. (13), α_{y_t} increases the weight of the underrepresented class. Moreover, during WO-XGB model training, w_j are updated dynamically based on misclassification rates as presented in Eq. (14).

$$w_j = \beta\, w_j \text{ if } H_j(x_t) \neq y_t \text{ and } y_t = 1 \tag{14}$$

In Eq. (14), β is a penalty factor that decreases tree weights when misclassifying the minority class. This ensures trees focusing on rare attacks are not removed prematurely. Further, for preventing bias towards the majority class, the tree weights are normalized, and trees with very little contribution are removed using Eq. (15) and Eq. (16), respectively.

$$w_j = \frac{w_j}{\sum_j w_j} \tag{15}$$

$$\mathcal{H} = \frac{\mathcal{H}}{\{H_j | w_j < \theta\}} \tag{16}$$

In Eq. (16), θ denotes the threshold for pruning trees that do not contribute significantly. This approach dynamically re-weights the loss function and adjusts DT weights to prioritize minority class predictions without generating synthetic data. It ensures that the WO-XGB model effectively learns from imbalanced data while preventing overfitting to the majority class. For effectively learning different features for predicting attacks, this work presents a training k-fold optimization in WO-XGB, which is discussed in detail in the next section.

3.5 Training K-Fold splitting optimization in WO-XGB

During training, WO-XGB splits the dataset into subsets, where some data is used for training and some data for testing, and the process is repeated multiple times to reduce

bias and variance during performance evaluation, considering the testing data. Consider the dataset $X_t = \{(x_1, y_1), (x_2, y_2), \ldots \ldots (x_t, y_t)\}$ where $\mathcal{X} = \{x_1, x_2, \ldots, x_t\}$ denotes features and $\mathcal{Y} = \{y_1, y_2, \ldots, y_t\}$ denotes response variable (attack labels). As the dataset X_t consists of independent and identically distributed points $(\mathcal{X}, \mathcal{Y})$ sampled from an unknown distribution P. For evaluating how the WO-XGB generalizes to unseen data, a loss function is used during training as presented in Eq. (17).

$$l(\widehat{y}_t, y_t) : y \times y \rightarrow \mathbb{R}_{\geq 0} \tag{17}$$

In Eq. (17), it measures the discrepancy between the predicted value $\widehat{y}_t$ and the actual true value y_t. Also, in Eq. (17), $l(\widehat{y}_t, y_t)$ quantifies the difference between prediction $\widehat{y}_t$ and actual true value y_t, $y \times y$ denotes the input $\widehat{y}_t$ and y_t belongs to the same space y_t, meaning they are comparable and $\mathbb{R}_{\geq 0}$ denotes output of the loss function is a non-negative real number, meaning the loss is always zero or positive. Let $f_k(X_t)$ which handles CD and CI issues, be represented as $f_{\widehat{\alpha}}(X_t)$. Let the $f_{\widehat{\alpha}}(X_t)$ be the prediction function parameterized by α, which belongs to a space Θ. Consider $\mathcal{X}$ is the model-fitting approach that returns parameter $\widehat{\alpha}$ based on training data. Then, the out-of-sample error is given using Eq. (18).

$$E_{\mathcal{X}\mathcal{Y}} = E\left[l\left(f_{\widehat{\alpha}}(\mathcal{X}_{n+1}), \mathcal{Y}_{n+1}\right) | (\mathcal{X}, \mathcal{Y})\right] \tag{18}$$

In Eq. (18), $(\mathcal{X}_{n+1}, \mathcal{Y}_{n+1}) \sim P$ is a test point from the same distribution. Hence, the overall expected prediction error is represented as Eq. (19).

$$E = E[E_{\mathcal{X}\mathcal{Y}}] \tag{19}$$

Further, for estimating error considering different features $\mathcal{X}$, this work considers k-fold CV, wherein the dataset X_t is randomly partitioned to N disjoint subsets $O_1, O_2, \ldots, O_N$, where every subset has size $s = n/N$. Further, the WO-XGB is trained considering $N - 1$ subsets and tested on the remaining folds. For every fold, first fit model on the training set using Eq. (20).

$$\widehat{\alpha}^{(-1)} = Q(\mathcal{X}_t, \mathcal{Y}_t)_{t \in m} \tag{20}$$

In Eq. (20), $\widehat{\alpha}^{(-1)}$ denotes leave-one-out estimation, $Q(\mathcal{X}_t, \mathcal{Y}_t)$ denotes a function that takes input data X_t or input data features at time t and target variable or response at time t, and $t \in m$ denotes that $Q(\mathcal{X}_t, \mathcal{Y}_t)$ is evaluated over multiple time points m. Further, for every test point, the prediction error is computed using Eq. (21).

$$e_i = l\left(f_{\widehat{\alpha}^{-1}}(\mathcal{X}_t), \mathcal{Y}_t\right), \forall \in m \tag{21}$$

Further, the average CV errors and standard error estimates are evaluated using Eq. (22) and Eq. (23), respectively.

$$\widehat{E}^{(CV)} = \frac{1}{n}\sum_{i=1}^{n} e_i \tag{22}$$

$$E = \frac{1}{\sqrt{n}} \cdot \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(e_i - \bar{e})^2} \tag{23}$$

Finally, for prediction error, the confidence intervals are constructed using Eq. (24).

$$A = \left(\bar{e} - z_{1-\frac{\alpha}{2}} \cdot E, \bar{e} + z_{1-\frac{\alpha}{2}} \cdot E\right) \tag{24}$$

In Eq. (24), $z_{1-\frac{\alpha}{2}}$ is a critical value for the standard normal distribution. Further, the WO-XGB incorporates two methods for predicting accuracy, i.e., using data split with refitting and without refitting. In the data split without refitting, the first dataset X_t is split into disjoint training sets M^{train} and M^{test}. First, the WO-XGB is trained using M^{train} using Eq. (25).

$$\widehat{\alpha}^{train} = Q(\mathcal{X}_t, \mathcal{Y}_t)_{t \in m^{train}} \tag{25}$$

The prediction error is evaluated considering the test set M^{test} using Eq. (26).

$$E^{split} = \frac{1}{|M^{out}|} \sum_{i \in M^{out}} e_i \tag{26}$$

The standard error estimate is evaluated using Eq. (27).

$$E^{split} = \sqrt{\frac{1}{|M^{out}| - 1} \sum_{i \in M^{out}} \left(e_i - \bar{E}^{split}\right)^2} \tag{27}$$

Using the data split without refitting provides an unbiased estimate but is dependent on the subset used for training. Similar to the previous method, i.e., data split without refitting, the data split with refitting is performed in a similar vein, but after evaluating E^{split}, the model is refitted on the X_t using Eq. (28).

$$\widehat{\alpha} = Q(\mathcal{X}_t, \mathcal{Y}_t) \tag{28}$$

Using the data split with refitting has shown better performance, as it has access to the complete X_t, but the error E^{split} remains low for predicting E, and the standard error estimate becomes unreliable. Hence, this WO-XGB considered data split without refitting for training, considering the features. This optimization approach in WO-XGB ensured that the model is evaluated using a robust CV technique. By dynamically adjusting weights and incorporating k-fold CV, the model achieves better generalization, reduces bias, and prevents overfitting. Furthermore, data splitting methods provide additional validation to ensure the model's reliability in detecting evolving attack

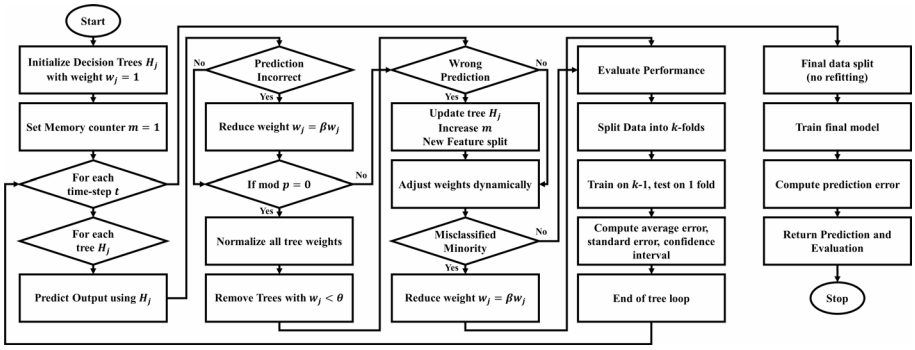

Fig. 3 Flowchart for WO-XGB for attack prediction

patterns. The complete model of WO-XGB is presented in Algorithm 2. For simplification and better understanding, the complete flow of the WO-XGB prediction is presented in Fig. 3.

Input Dataset X_t with features $x_1, x_2, \ldots, x_t$ and attack labels $y_1, y_2, \ldots, y_t$, number of DTs K, regularization parameters γ and λ, dynamic weight reduction penalty parameter β, tree removal threshold θ, CI weight parameters α_0 and α_1, number of past time steps m for weight optimization, number of k-folds for CV
Output Attack prediction $\hat{y}_t$ and performance evaluation
Step 1 Start
Step 2 Initialize DTs H_j with weight $w_j = 1$
Step 3 **For** all j
Step 4 Set $m = 1$
Step 5 **End for**
Step 6 **For** each t **do**
Step 7 **For** each DT H_j **do**
Step 8 Compute prediction using Eq. (4)
Step 9 **If** $H_j(X_t) \neq y_t$ and $i \; mod \; p \neq 0$, **then**
Step 10 $w_j = \beta w_j$
Step 11 **Else if** $i \; mod \; p = 0$
Step 12 $w_j = \frac{w_j}{\sum_j w_j}$
Step 13 $\mathcal{H} = \frac{\mathcal{H}}{\{H_j | w_j < \theta\}}$
Step 14 **Else** $\hat{y}_t \neq y_t$
Step 15 $m = m + 1$
Step 16 Perform feature-based split and update H_j
Step 17 **End if**
Step 18 Adjust tree weights dynamically using Eq. (13)
Step 19 **If** the tree misclassifies the minority class, **then**
Step 20 $w_j = \beta w_j$
Step 21 **End if**
Step 22 Evaluate the loss function using Eq. (11).
Step 23 Split the dataset X_t into k-folds
Step 24 **For** each fold O_i
Step 25 Train the model using k-1 folds using Eq. (20).
Step 26 Compute prediction error using Eq. (21)
Step 27 **End for**
Step 28 Compute average error using Eq. (22)
Step 29 Compute standard error using Eq. (23)
Step 30 Compute confidence interval using Eq. (24)
Step 31 **End for**
Step 32 **End for**
Step 33 Perform data splitting without refitting
Step 34 Train the model using Eq. (25)
Step 35 Compute prediction error using Eq. (26)
Step 36 Compute the standard error estimate using Eq. (27)
Step 37 Return attack prediction, performance evaluation results
Step 38 Stop

Algorithm 2 WO-XGB for attack prediction

For effectively addressing CI, WO-XGB presents a reweighting-based mechanism that assigns distinct weight parameters (α_0 for the majority class and α_1 for the minority class) within the loss function to penalize misclassification of minority attack classes more heavily. This ensures that rare yet critical cyber threats are not overlooked during training. Additionally, the model includes a dynamic weight reduction penalty β that adjusts tree weights when misclassification of minority classes occurs, promoting fairer learning across imbalanced datasets. To tackle CD, WO-XGB employs temporal ensemble pruning and adaptive weight updates over time, where trees that consistently underperform or fail to adapt to new patterns are pruned if their weight drops below a defined threshold θ. Furthermore, the model dynamically updates decision tree weights based on real-time prediction performance and recent history, using a sliding time window m to recalibrate importance. These mechanisms allow WO-XGB to learn incrementally,

Table 2 Hardware and software configuration for experimental setup

Component	Specification
Operating System	Windows 11
Processor	Intel Core i7
RAM	16 GB
Storage	1 TB SSD
Graphics Card (GPU)	NVIDIA GeForce GTX 1650
Programming Language	Python

Table 3 WO-XGB hyperparameters

Hyperparameter	Value	Description
$n\_estimators$	150	Total boosting rounds (trees).
$learning\_rate$	0.05	Step size shrinkage is used in the update to prevent overfitting.
max_depth	6	Maximum depth of a tree.
$subsample$	0.8	Fraction of training instances used to grow each tree.
$colsample\_bytree$	0.8	Fraction of features to be randomly sampled for each tree.
$gamma$	0.1	Minimum loss reduction required to make a further partition.
min_child_weight	1	Minimum sum of instance weight needed in a child.
$objective$	binary: logistic	Learning task and the corresponding objective function.
$eval\_metric$	AUC	Evaluation metric used during training.
$scale\_pos\_weight$	1	Controls the balance of positive and negative weights (adjusted for CI handling).
$random\_state$	42	Ensures reproducibility of results.
$early\_stopping\_rounds$	10	Stops training early if performance does not improve.

adapt to evolving attack patterns, and maintain high detection accuracy even as threat behaviors change over time. Using the following WO-XGB model, the attack prediction was conducted on multiple datasets, which are discussed in the results and discussion section.

4 Results and discussion

For this study, two standard datasets, Edge-IIoTset [14] and CICIoT2023 [15], were utilized to evaluate the WO-XGB model. The Edge-IIoTset dataset comprises diverse attack scenarios in IIoT environments, incorporating multiple protocols, devices, and attack types. Similarly, the CICIoT2023 dataset, which includes a wide range of IoT-specific cyber threats, is an important resource for assessing intrusion detection models. The performance metrics used in this study are presented in Eqs. (29) and (36), where FP represents False-Positive, TP represents True-Positive, FN represents False-Negative, TN represents True-Negative, and N denotes test data. The Eq. (29) to (32), i.e., accuracy, precision, recall, and f-score, were considered because the existing approaches [17, 21, 24], and [25] have used the same metrics for results evaluation.

$$Accuracy = \frac{TP + TN}{TN + FP + TP + FN} \tag{29}$$

$$Precision = \frac{TP}{FP + TP} \tag{30}$$

$$Recall = \frac{TP}{FN + TP} \tag{31}$$

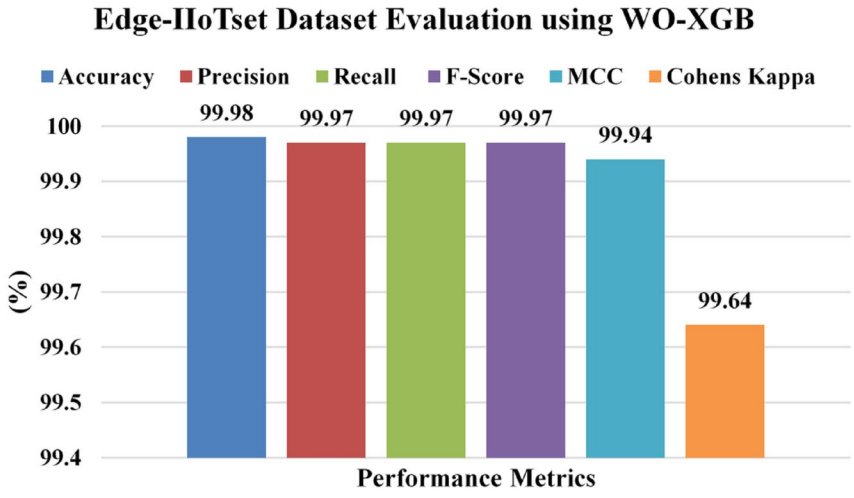

Fig. 4 WO-XGB evaluation on Edge-IIoTset dataset

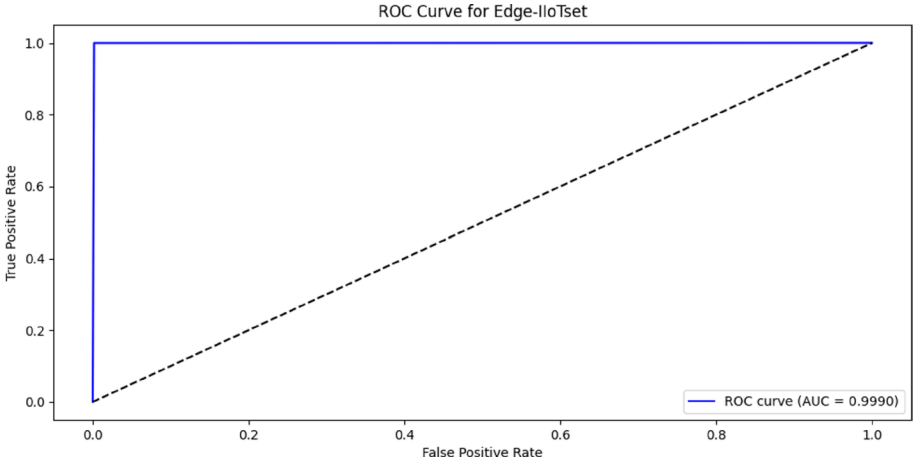

Fig. 5 ROC-AUC curve achieved by WO-XGB for Edge-IIoTset

$$F - Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{32}$$

$$MCC = \frac{(TP \cdot TN - FP \cdot FN)}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{33}$$

$$Cohen's\ Kappa = \kappa = \frac{P_o - P_e}{1 - P_e} \tag{34}$$

$$P_o = \frac{TP + TN}{N} \tag{35}$$

$$P_e = \frac{(TP + FP)(TP + FN) + (TN + FN) + (TN + FP)}{N^2} \tag{36}$$

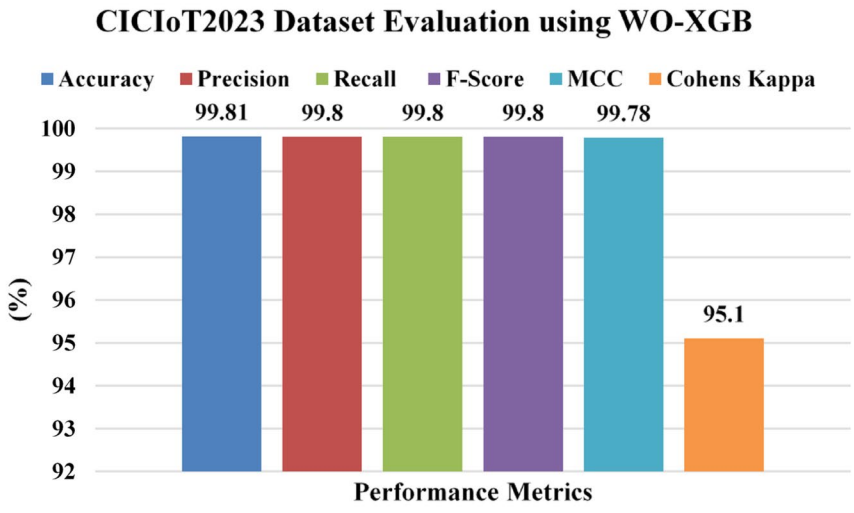

Fig. 6 WO-XGB evaluation on CICIoT dataset

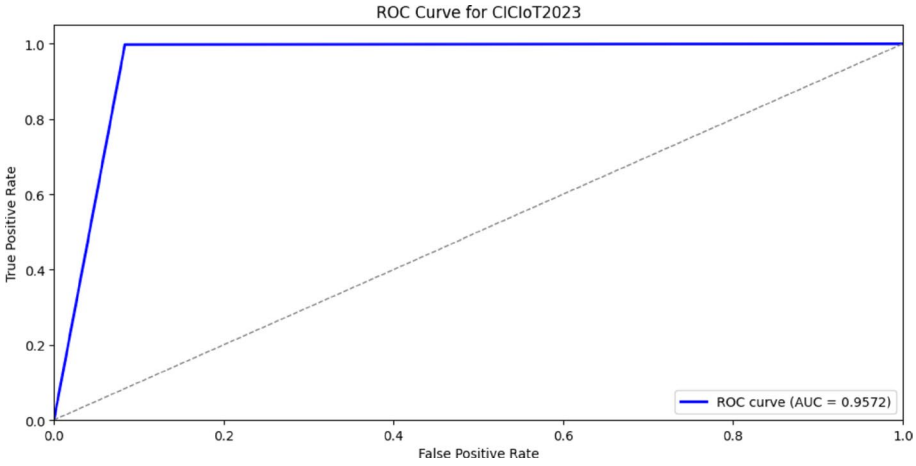

Fig. 7 ROC-AUC curve achieved by WO-XGB for CICIoT2023

4.1 Experimental setup

The WO-XGB model was developed and evaluated on a system running the Windows 11 operating system using Python-based ML libraries. For accelerated computation and efficient processing of large datasets, the system was equipped with an NVIDIA GeForce GTX 1650 GPU. The complete setup is detailed in Table 2.

4.2 WO-XGB hyperparameters

The hyperparameters used in WO-XGB are discussed in detail in Table 3.

4.3 Edge-IIoTSet evaluation

This section discusses results achieved by the WO-XGB model on Edge-IIoTset. The findings evaluated from Eq. (29) to Eq. (34) are presented in Fig. 4. Also, the Receiver Operating Characteristic - Area Under the Curve (ROC-AUC_ curve achieved by WO-XGB for Edge-IIoTSet is presented in Fig. 5. In Fig. 4, the WO-XGB model achieves

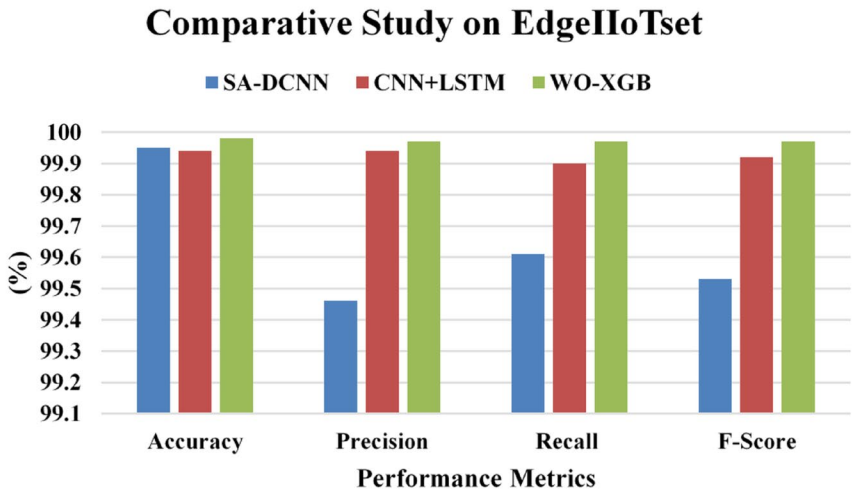

Fig. 8 Comparative Study of WO-XGB with other ML and DL models on Edge-IIoTset

Table 4 Comparative study of WO-XGB with other ML and DL models on Edge-IIoTset

Ref No.	Model	Accuracy	CD	CI	Training Time (sec)	Testing Time (sec)
[17]	SA-DCNN	99.95	No	Yes	213	22
[25]	CNN + LSTM	99.94	No	Yes	-	59.11
Proposed	WO-XGB	99.98	Yes	Yes	160	16.5

near-perfect results on the Edge-IIoTset dataset, with 99.98% accuracy, and equally high precision, recall, and F-score (all at 99.97%). The Matthews Correlation Coefficient (MCC) of 99.94 and Cohen's Kappa score of 99.64 indicates that WO-XGB not only performs well overall but maintains strong reliability across both classes, despite CI. This high level of consistency shows WO-XGB's capacity to generalize well without overfitting. Further, Fig. 5 confirms the effectiveness of WO-XGB on Edge-IIoTset through its ROC-AUC score of 99.90, signifying exceptional class separability. The high AUC value suggests that WO-XGB maintains a low FPs rate while effectively detecting attacks, which is critical in IoT security settings.

4.4 CICIoT2023 evaluation

This section discusses results achieved by the WO-XGB model on CICIoT2023. The findings evaluated from Eq. (29) to Eq. (34) are presented in Fig. 6. Also, the ROC-AUC curve achieved by WO-XGB for CICIoT2023 is presented in Fig. 7. Figure 6 shows WO-XGB maintains strong performance on the CICIoT2023 dataset, achieving 99.81% accuracy, with precision, recall, and F-score all at 99.8%. Although the dataset is more complex than Edge-IIoTset, WO-XGB still delivers high MCC (99.78) and Cohen's Kappa (95.1), indicating resilience in more dynamic and heterogeneous environments. Further, Fig. 7 presents the ROC-AUC curve for WO-XGB on CICIoT2023, with a 95.72 score, slightly lower than Edge-IIoTset but still very strong. This illustrates the WO-XGB's ability to maintain reliable class discrimination under varying dataset characteristics, highlighting its adaptability to unseen or evolving attack behaviors.

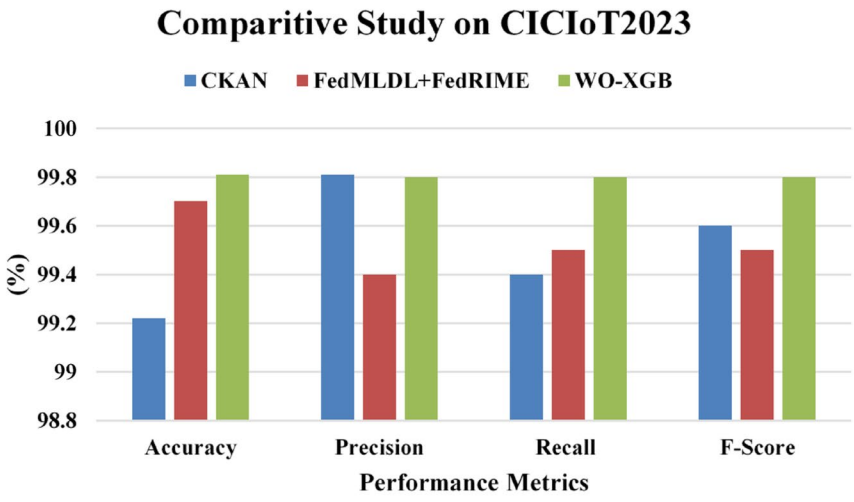

Fig. 9 Comparative Study of WO-XGB with other ML and DL models on CICIoT2023

Table 5 Comparative study of WO-XGB with other ML and DL models on CICIoT2023

Ref No.	Model	Accuracy	CD	CI	Training Time (sec)	Testing Time (sec)
[21]	CKAN	99.22	No	No	21,210	-
[24]	FedMLDL + FedRIME	99.7	No	Yes	-	-
Proposed	WO-XGB	99.81	Yes	Yes	5302.5	1800

4.5 Comparative study-edge-IIoTset

This section conducts a comparative study with existing models. The comparative study considered the standard performance metrics as presented in Eq. (29) to Eq. (30). The comparative study on the Edge-IIoTset using WO-XGB, SA-DCNN [17], and CNN + LSTM [25] is presented in Fig. 8. The comparative study shows that WO-XGB achieved 99.98% accuracy, whereas the SA-DCNN achieved 99.95% and CNN+-STM achieved 99.94% accuracy. Also, when compared to other metrics, the WO-XGB showed better performance. Further, in Table 4, a comparative study has been conducted, which shows accuracy and whether they handled CI and CD. Findings show that SA-DCNN and CNN + LSTM models handled CI but failed to handle the CD issue, whereas the proposed WO-XGB model handled CD and CI issues efficiently. The WO-XGB model's ability to dynamically adapt to changing attack patterns and provide high accuracy shows it is suitable for intrusion detection in IoT, edge, and cloud environments.

From comparative results on the Edge-IIoTset dataset as presented in Fig. 8, WO-XGB achieves the highest overall performance, with accuracy of 99.98%, precision and recall of 99.97%, and F-score of 99.97%. These results are marginally but meaningfully better than SA-DCNN (99.95% accuracy) and CNN + LSTM (99.94% accuracy). While SA-DCNN uses a self-attention mechanism combined with DCNN for feature extraction and includes a feature-filtering approach based on MI, it does not address CD, a crucial limitation in dynamic IIoT environments where attack patterns evolve and training time and testing time are high in comparison with SA-DCNN. Similarly, CNN + LSTM combines feature extraction with temporal modeling and employs SMOTE for CI, but again fails to consider CD. Also, the time used for testing is more in comparison with WO-XGB. In contrast, WO-XGB directly integrates mechanisms to tackle both CI and

CD, dynamically optimizing tree weights and pruning outdated decision trees, making it more resilient in continuously changing data environments, reducing training time and testing time in comparison with SA-DCNN [17] and CNN + LSTM [25].

4.6 Comparative study-CICIoT2023

This section conducts a comparative study with existing models. The comparative study considered the standard performance metrics as presented in Eq. (29) to Eq. (30). The comparative study on the CICIoT2023 using WO-XGB, CKAN [21], and FedMLDL + FedRIME [25] is presented in Fig. 9. The comparative study shows that WO-XGB achieved 99.81% accuracy, whereas the FedMLDL + FedRIME achieved 99.7% and CKAN achieved 99.22% accuracy. Also, when compared to other metrics, the WO-XGB showed better performance. Further, in Table 5, a comparative study has been conducted, which shows accuracy and whether they handled CI and CD. Findings show that CKAN failed to handle both CI and CD, whereas the FedMLDL + FedRIME model handled CI but failed to handle CD issues, whereas the proposed WO-XGB model handled CD and CI issues efficiently. The WO-XGB model's ability to dynamically adapt to changing attack patterns and provide high accuracy shows it is suitable for intrusion detection in IoT, edge, and cloud environments.

From comparative results on the CICIoT2023 dataset, as presented in Fig. 9, WO-XGB again demonstrates superior performance with 99.81% accuracy, outperforming CKAN (99.22%) and FedMLDL + FedRIME (99.7%). CKAN introduces an innovative use of KAN layer for improved non-linear modeling within CNNs and shows good results, especially in binary classification. However, it lacks mechanisms for handling either CD or CI, making it less effective in real-world scenarios, where attack types and distributions shift frequently. Also, the time for training is high in comparison with WO-XGB. FedMLDL + FedRIME incorporates FL and feature engineering using PCC, along with physics-based hyperparameter optimization. While it does address CI, it does not account for CD, and its complex architecture might introduce additional overheads. The key differentiator for WO-XGB is its dynamic weight optimization mechanism that adapts to temporal changes in attack patterns. By penalizing poorly performing DTs and removing them based on weight thresholds, WO-XGB maintains an efficient ensemble of relevant models over time. Furthermore, its incorporation of k-fold CV, loss-based learning, and confidence interval estimations ensures robust generalization without overfitting. This adaptability allows WO-XGB to perform consistently well across both balanced and imbalanced datasets, and in environments affected by CD, reducing both training and testing time.

4.7 Discussion

The experimental results and comparative evaluations presented in previous sections demonstrate the effectiveness, robustness, and adaptability of the WO-XGB model across two benchmark datasets, Edge-IIoTset and CICIoT2023. The metrics results, i.e., accuracy, precision, recall, F-score, MCC, and Cohen's Kappa, indicate that WO-XGB is capable of detecting diverse and complex attack types with minimal FPs or FNs. Notably, the model consistently outperforms existing state-of-the-art methods such as SA-DCNN, CNN + LSTM, CKAN, and FedMLDL + FedRIME, with substantial margins in some cases. This superiority is attributed to WO-XGB's novel design, which integrates

dynamic weight optimization, confidence interval-aware pruning, and temporal adaptation mechanisms to directly tackle both CI and CD, two critical challenges often overlooked in traditional and even deep learning-based intrusion detection approaches.

A key finding is that existing models, while achieving high accuracy, fail to generalize effectively in environments where the data distribution shifts over time or where minority attack classes are underrepresented. In contrast, WO-XGB demonstrates strong resilience and adaptability by dynamically penalizing underperforming trees and pruning outdated classifiers. This not only ensures the ensemble remains focused on relevant and recent data patterns but also reduces computational overhead by avoiding redundant or obsolete models. Furthermore, integration of CV and statistical estimation (confidence intervals and error bounds) enhances WO-XGB's generalization capability and stability across unseen data.

The marginal yet meaningful performance improvements observed in both datasets, especially in challenging scenarios like CICIoT2023, highlight the practical significance of addressing CI and CD simultaneously. While models like FedMLDL + FedRIME include FL and optimization layers, they introduce additional architectural complexity without fully resolving CD, limiting their deployment, resource-constrained environments. WO-XGB, in contrast, offers a simpler yet more adaptive alternative, with minimal tuning and strong real-time suitability for IoT, edge, and cloud-based intrusion detection systems. These results confirm that WO-XGB not only achieves superior predictive performance but also introduces a scalable and generalizable approach that bridges the gap between traditional ensemble learning and modern adaptive intelligence. Its success across two heterogeneous datasets further supports its applicability in diverse cybersecurity contexts.

5 Conclusion and future work

The IoT has revolutionized various domains, including healthcare, smart cities, industrial automation, and transportation, by enabling seamless connectivity between physical and digital systems. However, as IoT devices generate vast amounts of data and operate in distributed environments, they become vulnerable to cyber threats such as DoS, MiTM, phishing, and spoofing attacks. The dynamic and evolving nature of these attacks presents significant challenges for traditional ML and DL security models, particularly due to CD and CI issues. To address these challenges, this study proposed a feature-level ensemble ML approach, WO-XGB, which dynamically adapts to evolving attack patterns through a weight optimization mechanism and handles CI using a reweighting strategy. Additionally, k-fold CV optimization was integrated into WO-XGB to enhance model generalization and prevent overfitting. The model was evaluated on two standard intrusion detection datasets, Edge-IIoTset and CICIoT2023, to assess its effectiveness in detecting cyberattacks. Experimental results demonstrated that WO-XGB outperformed existing ML and DL models, achieving 99.98% accuracy on Edge-IIoTset and 99.81% accuracy on CICIoT2023. Also, a comparative analysis showed that WO-XGB effectively handled both CD and CI, whereas existing models have failed to address these issues.

Future work could focus on incorporating FL-based security models into WO-XGB to enhance its adaptability in distributed IoT networks, ensuring real-time attack prevention with minimal computational overhead.

Acknowledgements
The authors acknowledge the use of Grammarly AI for proofreading and correcting English grammatical errors in the manuscript.

Author contributions
Conceptualization, FK and SK; methodology, FK; validation, SK; formal analysis, FK; investigation, resources, FK; data curation, FK; writing the original draft, FK; writing-review and editing, FK, SK, SS; visualization, FK; supervision and project administration, SK; funding acquisition, SS; methodology support, SS and SK; literature review, FK; data analysis, SK and SS; project coordination, FK.

Funding
Open access funding provided by Manipal Academy of Higher Education, Manipal. This work was supported by the Manipal Academy of Higher Education (Open Access Funding).

Data availability
The datasets used and/or analyzed during the current study are available from the corresponding author on reasonable request.

Declarations

Ethics approval and consent to participate
Not applicable.

Consent for publication
Authors provided written consent for the publication of their findings.

Competing interests
The authors declare no competing interests.

Received: 24 April 2025 / Accepted: 14 July 2025
Published online: 24 July 2025

References

1. Chataut R, Phoummalayvane A, Akl R. Unleashing the power of iot: a comprehensive review of IoT applications and future prospects in healthcare, agriculture, smart homes, smart cities, and industry 4.0. Sensors. 2023;23(16):7194. https://doi.org/10.3390/s23167194.
2. Nguyen T, Nguyen H, Gia TN. Exploring the integration of edge computing and blockchain IoT: principles, architectures, security, and applications. J Network Computer Appl. 2024;103884. https://doi.org/10.1016/j.jnca.2024.103884.
3. Yaacoub J-PA, Noura HN, Salman O, Chehab A. Ethical hacking for iot: security issues, challenges, solutions and recommendations. Internet Things Cyber-Phys Syst. 2023;3. https://doi.org/10.1016/j.iotcps.2023.04.002.
4. Sadhu PK, Yanambaka VP, Abdelgawad A. Internet of Things: security and solutions survey. Sensors. 2022;22(19):7433. https://doi.org/10.3390/s22197433.
5. Nazir A, et al. Empirical evaluation of ensemble learning and hybrid CNN-LSTM for IoT threat detection on heterogeneous datasets. J Supercomput. 2025;81(6). https://doi.org/10.1007/s11227-025-07255-1.
6. Sivasankari N, Kamalakkannan S. Detection and prevention of man-in-the-middle attack in Iot network using regression modeling. Adv Eng Softw. Jul. 2022;169:103126. https://doi.org/10.1016/j.advengsoft.2022.103126.
7. Alotaibi SR et al. Oct., Explainable artificial intelligence in web phishing classification on secure IoT with cloud-based cyber-physical systems. Alex Eng J. 2024;110:490–505. https://doi.org/10.1016/j.aej.2024.09.115.
8. AboulEla S, Ibrahim N, Shehmir S, Yadav A, Kashef R. Navigating the cyber threat landscape: an in-depth analysis of attack detection within IoT ecosystems. AI. 2024;5(2):704–32. https://doi.org/10.3390/ai5020037.
9. Tauqeer H, Iqbal MM, Ali A, Zaman S, Chaudhry MU. Cyberattacks detection in IoMT using machine learning techniques. J Comput Biomed Inform. 2022;4(01):13–20. https://doi.org/10.56979/401/2022/80.
10. Saheed YK, Misra S, CPS-IoT-PPDNN. A new explainable privacy preserving DNN for resilient anomaly detection in cyber-physical systems-enabled IoT networks. Chaos Solit Fract. 2024;191:115939. https://doi.org/10.1016/j.chaos.2024.115939.
11. El-Shafeiy E, Elsayed WM, Elwahsh H, Alsabaan M, Ibrahem MI, Elhady GF. Deep complex gated recurrent networks-based IoT network intrusion detection systems. Sensors. 2024;24(18):5933. https://doi.org/10.3390/s24185933.
12. J NAH, Prakash SPS, Krinkin K. Class imbalance and concept drift invariant online botnet threat detection framework for heterogeneous IoT edge. Computers Secur. 2024;141:103820. https://doi.org/10.1016/j.cose.2024.103820.
13. Priya S, Uthra A. Ensemble framework for concept drift detection and class imbalance in data streams. Multimed Tools Appl. 2024. https://doi.org/10.1007/s11042-024-18349-y.
14. Ferrag MA, Friha O, Hamouda D, Maglaras L, Janicke H. Edge-IIoTset: a new comprehensive realistic cyber security dataset of IoT and IIoT applications for centralized and federated learning. IEEE Access. 2022;10:40281–306. https://doi.org/10.1109/ACCESS.2022.3165809.
15. Pinto C, Sajjad Dadkhah R, Ferreira A, Zohourian R, Lu, Ghorbani AA. CICIoT2023: a real-time dataset and benchmark for large-scale attacks in IoT environment. Sensors. 2023;23(13):5941. https://doi.org/10.3390/s23135941.
16. Aburasain RY. Enhanced black widow optimization with hybrid deep learning enabled intrusion detection in internet of things-based smart farming. IEEE Access. 2024;12:16621–31. https://doi.org/10.1109/ACCESS.2024.3359043.
17. Alshehri MS, Saidani O, Alrayes FS, Abbasi SF, Ahmad J. A self-attention-based deep convolutional neural networks for IIoT networks intrusion detection. IEEE Access. 2024;12:45762–72. https://doi.org/10.1109/ACCESS.2024.3380816.
18. Khan MM, Alkhathami M. Anomaly detection in IoT-based healthcare: machine learning for enhanced security. Sci Rep. 2024;14(1). https://doi.org/10.1038/s41598-024-56126-x.

19. Tseng S-M, Wang Y-Q, Wang Y-C. Multi-class intrusion detection based on transformer for IoT networks using CIC-IoT-2023 dataset. Future Internet. 2024;16(8):284. https://doi.org/10.3390/fi16080284.
20. Javeed D, Gao T, Saeed MS, Kumar P. An intrusion detection system for Edge-Envisioned smart agriculture in extreme environment. IEEE Internet Things J. 2024;11(16):26866–76. https://doi.org/10.1109/JIOT.2023.3288544.
21. Abd Elaziz M, Ahmed Fares I, Aseeri AO. CKAN: Convolutional Kolmogorov–Arnold networks model for intrusion detection in IoT environment. IEEE Access. 2024;12:134837–51. https://doi.org/10.1109/ACCESS.2024.3462297.
22. Almadhor A, Altalbe A, Bouazzi I, Hejaili AA, Kryvinska N. Strengthening network DDOS attack detection in heterogeneous IoT environment with federated XAI learning approach. Sci Rep. 2024;14(1). https://doi.org/10.1038/s41598-024-76016-6.
23. Belachew HM, Beyene MY, Desta AB, Alemu BT, Musa SS, Muhammed AJ. Design a robust DDoS attack detection and mitigation scheme in SDN-Edge-IoT by leveraging machine learning. IEEE Access. 2025;13:10194–214. https://doi.org/10.1109/ACCESS.2025.3526692.
24. Chandnani CJ, Agarwal V, Chetan Kulkarni S, Aren A, Amali DGB, Srinivasan K. A physics-based hyper parameter optimized federated multi-layered deep learning model for intrusion detection in IoT networks. IEEE Access. 2025;13:21992–2010. https://doi.org/10.1109/ACCESS.2025.3535952.
25. Berguiga A, Harchay A, Massaoudi A. HIDS-IoMT: a deep learning-based intelligent intrusion detection system for the internet of medical things. IEEE Access. 2025;13:32863–82. https://doi.org/10.1109/ACCESS.2025.3543127.

Publisher's note