IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 29 August 2023, accepted 23 September 2023, date of publication 27 September 2023, date of current version 4 October 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3319814

RESEARCH ARTICLE

IGWO-SoE: Improved Grey Wolf Optimization Based Stack of Ensemble Learning Algorithm for Anomaly Detection in Internet of Things Edge Computing

J. MANOKARAN[1], AND G. VAIRAVEL[2], (Senior Member, IEEE)
[1]Department of Electronics and Communication Engineering, SRM Institute of Science and Technology, Kattankulathur, Chennai, Tamil Nadu 603203, India
[2]Directorate of Learning and Development, SRM Institute of Science and Technology, Kattankulathur, Chennai, Tamil Nadu 603203, India

Corresponding authors: G. Vairavel (vairavelg@protonmail.com) and J. Manokaran (manoraj3@gmail.com)

ABSTRACT With the tremendous growth and popularization of the Internet of Things (IoT), the number of attacks targeting such devices has also increased. Therefore, enhancing the anomaly detection model to maximize detection accuracy and mitigate cyber-attacks in time-critical IoT edge scenarios is essential. Furthermore, there is a lack of vivid, precise, cross-layered, and diverse datasets in IoT for evaluating these anomaly detection models. This paper aims to develop an improved anomaly detection model based on an optimized stacked ensemble learning algorithm at edge computing. Initially, a novel synthetic dataset with multiple cross-layer attacks is generated using the Cooja simulator to train our proposed model. In addition, by introducing an improved grey wolf optimization (IGWO) approach, the parameters of ensemble learning algorithms, such as number of trees, learning rate, and sample rate, are tuned precisely, and the stacking ensemble concept is applied to the optimized ensemble learning algorithms to enhance their prediction capabilities. The experimental results demonstrate that the developed model produces a detection accuracy of 99.44% for our proposed Cooja simulated dataset, which is higher than the contemporary methods. The generalizability of the proposed model is expressed explicitly using four different datasets: NSL KDD, UNSW NB 15, MQTTset, and CICIDS 2017. Finally, we assess the befitting of the proposed model using a chi-square statistical significance test, thereby providing an enriched contribution to the recent works in anomaly detection.

INDEX TERMS Anomaly detection, cooja simulator, edge computing, ensemble learning, improved grey wolf optimization, Internet of Things, statistical significance test.

I. INTRODUCTION

A. BACKGROUND

Due to the widespread deployment of the IoT and 5G technology, billions of smart devices are interconnected, making our lives smarter. The IoT has created tremendous opportunities for innovation in various applications, such as intelligent transport, smart homes, e-healthcare, smart agriculture, and Industry 4.0 [1], [2]. It enhances the quality of life, productivity, and profitability. Cisco estimates that 75 billion smart devices will be interlinked with the Internet by 2026 [3]. According to the IDC report, the global IoT expenditure is anticipated to cross $1.2 trillion in 2023 [4]. Figure. 1 (a) shows the number of interlinked devices to IoT globally from 2020 to 2025 [5]. The extended use of IoT leads to the misuse of IoT services, and it becomes a threat to critical information. Security and privacy are the two major concerns that must be considered for IoT to become a universally adopted technology [6]. Compared to the traditional system of security measures, IoT network security is very complex because of its vast volume, ad-hoc nature, heterogeneity, multi-dimensionality, and less

The associate editor coordinating the review of this manuscript and approving it for publication was Kashif Sharif.

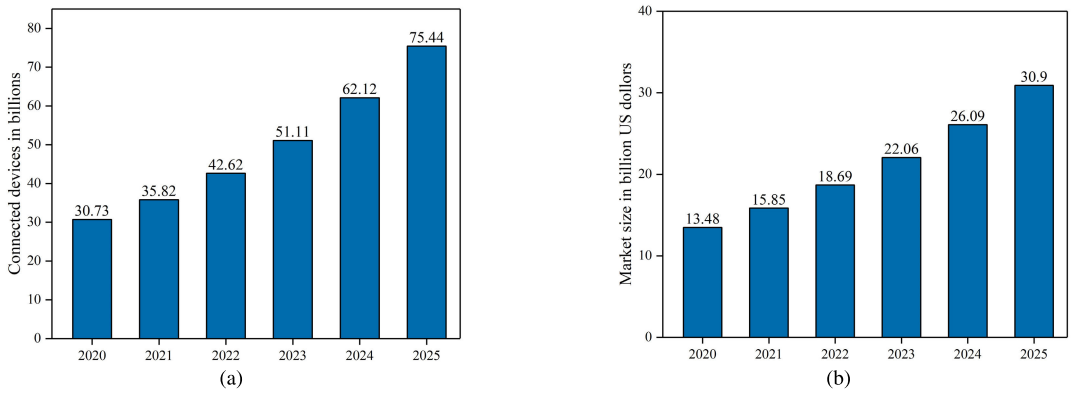

FIGURE 1. (a) Number of connected devices in billions (2020-2025) [5] (b) IoT security market size in billion US dollars (2020-2025) [7].

computation memory. The market size of the IoT security from 2020 to 2025 is represented in Figure. 1 (b) [7].

IoT development is tremendous and ubiquitous in modern cultures, where intruders create malicious activity towards the IoT system hence security failure is a serious topic. The development of new technology aims to tackle these issues by reducing attacks and defending existing services by regulating abnormal behaviors [8]. Intrusion detection systems (IDS), which are frequently used to observe network behavior and abnormal activity, have a noteworthy place in the IoT. There are two distinct sets of IDS based on their detection methodologies: anomaly-based detection and signature-based detection. A signature-based detection using predefined traffic patterns can efficiently distinguish legitimate activities from fraudulent ones [9]. Despite the effectiveness of signature-based detectors for recognizing known attacks, they cannot detect unprecedented attacks. Anomaly-based detection detects irregularities in regular network traffic. Although anomaly-based IDS are better at detecting recent attacks, the accuracy of predicting anomalies could be better [10].

Numerous studies have shown that machine learning (ML) algorithm can provide better detection mechanisms against new attacks across all cyber-security domains in IoT. The automation and quick learning capabilities of the ML algorithm make it the backbone of an IDS. In previous research, individual ML algorithms have been examined, but their false alarm rates and detection rates are not more effective. Many ML algorithms have been considered for building IDS, such as decision tree (DT) [11], dimensionality reduction algorithms [12], random forest (RF) [13], swarm intelligence techniques [14], support vector machine (SVM) [15], K-nearest neighbor (KNN) [16], logistic regression (LR) [16], and naive Bayes (NB) [17]. However, designing a robust anomaly detection model using a single ML algorithm is a challenging endeavor.

Nowadays, ensemble learning algorithms have been introduced to increase the performance and robustness of ML algorithms. Ensemble learning can be used in various real-time prediction problems, and IDSs are no exception to this. The core idea behind the ensemble learning algorithm is that when several ML algorithms are used, the detection errors of a single ML algorithm are balanced by other ML algorithms. As a result, an ensemble's final prediction will be more accurate than that of a single classifier [18]. Several ensemble learning algorithms are used to develop an effective IDS, such as extra trees (ET) [19], light gradient boosting machine (LGBM), adaptive boosting (Ada Boost), gradient boosting (Gr Boost), extreme randomized trees (ERT), and extreme gradient boosting (XG Boost) algorithms [20].

The ensemble learning algorithm provides better prediction accuracy, but there is still a need to improve its detection accuracy. Hyper-parameter tuning is vital for improving detection accuracy and reducing training time. The tuned ML algorithm's parameter produces better performance than the normal parameter. Finding the ideal hyper-parameter values is a tedious and expensive computation. Random search, Bayesian optimization algorithm, and grid search are the most utilizable tuning techniques [21].

Recently, the grey wolf optimization (GWO) strategy has emerged as a potential meta-heuristic method for resolving various optimization issues by imitating the social structure and propensity for hunting grey wolves [22]. Although the performance of the GWO is good, it has a limitation of imbalance between exploitation and exploration. It is easily trapped into the local best value while cracking multi-dimensional problems. Hence, an improved grey wolf optimization is proposed, which integrates two principles: survival of the fittest (SOF) and opposition-based learning (OBL).

B. MOTIVATION AND CONTRIBUTION

Most of the existing anomaly detection models have been implemented in the cloud. Still, modern fields like smart cities, smart agriculture, connected cars, and Industry 4.0 need the detection of anomalies at the edge, an emerging and less explored area. Furthermore, several current techniques are evaluated using an obsolete data set (i.e.,

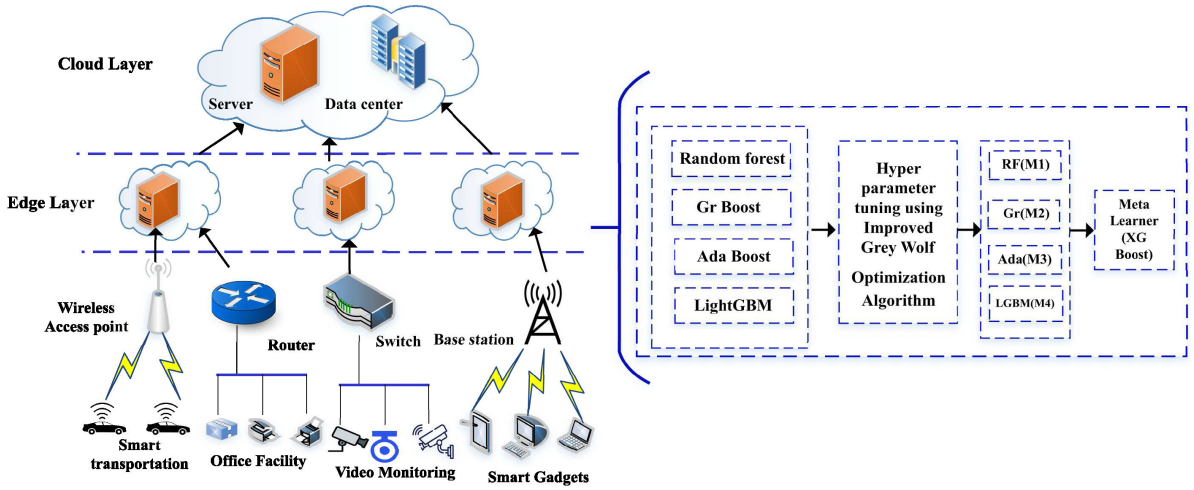

FIGURE 2. Network geography of IoT-edge: construction and placement.

NSL-KDD), which has limitations regarding the most recent IoT-based attack vectors. Based on the ''no free lunch theorem'', there isn't a universal optimization algorithm that can resolve all optimization issues, whereas it is always feasible to enhance the existing optimization algorithms. This study aims to fill these gaps by creating and using a novel Cooja-simulated dataset and a modern data source that is publicly accessible and comprises several IoT-based network threat vectors. The significant anomaly detection challenge was solved using an improved grey wolf-optimized stack of ensemble classifiers. We have analyzed the developed system with five different datasets, namely Cooja simulated, NSL KDD [23], UNSW NB 15 [24], CICIDS 2017 [25], and MQTTset [26] dataset. Finally, the proposed system is evaluated using standard classification metrics. Figure. 2 shows the IoT edge computing framework; here, the node information is processed near the user and sent to the cloud.

The objective of this study is to develop an effective anomaly detection model using an optimized ensemble learning algorithm. The ***key contributions*** of this work is as follows.

(i) To create many IoT attack scenarios using the Cooja simulator, the novel dataset is generated to train and test our developed algorithm.
(ii) Using recent efficient ensemble learning algorithms such as Ada Boost, Gr Boost, LGBM, and RF algorithm for anomaly detection and their performance is compared.
(iii) Using IGWO, the hyper-parameters of the ensemble learning algorithms, such as number of trees, learning rate, and sample rate are optimized to enhance the detection accuracy.
(iv) The final model is built with a stack of optimized ensemble learning algorithms and the performance is analyzed using a simulated dataset and then compared with the open-source dataset.
(v) A chi-square statistical test verifies the developed model's goodness-of-fit.

The arrangement of this paper is as follows: Section II explains the recent related studies for anomaly detection systems. Section III illustrates the proposed methodology, ML algorithms, and optimization techniques used for our proposed system. Section IV creates a different experimental setup and investigates the results. Finally, section V concludes the work.

II. RELATED WORK

The protection system of IoT devices is highly prone to attack due to increased cyber-attacks. Recent studies present numerous ways for the identification of these attacks with the use of ML algorithms and soft computing techniques. Some of the related research in a similar direction is discussed in this section.

Ensemble learning algorithms have shown a significant expansion over single ML algorithm. Recently, Tama et al. systematically studied a different ensemble learning algorithm for anomaly detection. The author discussed the current trends in ensemble learning algorithms and developed an anomaly detection system using advanced ensemble learning concepts. Finally, the developed ensemble learning performance is compared with a single ML algorithm [27]. Thakkar et al. surveyed various IDS using ML algorithms [28]. The author concludes that, compared to a single ML algorithm, the ensemble learning algorithm's detection accuracy and false alarm rate have better performance.

We have discussed various existing anomaly detection models using ensemble learning and artificial intelligence methods in Table 1. Based on the classifier used, the ensemble algorithm is divided into homogeneous and heterogeneous. In homogeneous, the same type of classifiers are used; in heterogeneous, different types of classifiers are used. The classifiers are combined using the concepts of bagging and boosting for the homogeneous algorithm and stacking for the heterogeneous algorithm.

In [29], Tama et al. developed abnormality detection in network traffic using stacking ensemble learning, where

TABLE 1. Summary of ensemble learning techniques in anomaly detection.

Study	Ensemble learning Techniques	Base Learning Algorithm	Optimization Techniques	Dataset	Performance Measure	Limitations
[29]	Stacking	XG Boosting, RF, GBM	Grid Search	NSL-KDD, CIC-IDS2017, UNSW-NB15	AUC value, Accuracy, FPR	Does not cover multi-classification
[30]	Boosting	XG Boosting	GA	N-BaIoT	Accuracy, Precision, Recall, F1 Score	In ability to explore the search of the solution
[31]	B-Stacking	KNN, RF, XG Boost	Not mentioned	CICIDS 2017, NSL KDD	Accuracy, Precision, Recall, F1 Score	Limited number of dataset
[32]	Bagging	RF	PSO	Simulated dataset	Accuracy, Precision, Recall	Not cover wide range of attack
[33]	LGBM	OCSVM	PSO	UNSW-NB15	Accuracy, FAR	Less accuracy & High FAR
[34]	Boosting	XG Boosting	PSO	NSL KDD	Mean average precision	Easily fall into local optimum
[35]	Majority voting	Rotation forest, Bagging	Not mentioned	NSL-KDD, UNSW-NB15	Accuracy, FPR, Recall	Does not cover multi-classification
[36]	LGBM	XG Boosting	GA	DS2OS	Accuracy, ROC curve, AUC value	Limited data size restriction
[37]	MLP	Not applicable	BAT Algorithm	KDD CUP 99, ISCX 2012, NLS-KDD, UNSW NB15	Accuracy, FPR, Recall	Does not cover multi-classification
[38]	Boosting	XG Boosting	Firefly algorithm	NLS-KDD, UNSW NB15	Accuracy, Precision, Recall, F1 Score	Low accuracy rate
[39]	Not applicable	DNN	Bayesian Optimization	BoT-IoT	Accuracy, Loss	Limited dataset
[40]	Boosting	XG Boosting	Whale Optimization Algorithm	KDD CUP 99	Accuracy, Sensitivity, Specificity	Not cover new dataset
[41]	Voting	Boosted Trees, RUS Boosted Trees	Not mentioned	Simulated	Accuracy, ROC curve,AUC value	Less accuracy & single dataset

the RF, Gr Boost, and XG Boost algorithms are used as base learners and using grid search, the parameters are tuned. A generalized linear algorithm is used as a meta-classifier. The suggested algorithm performed better, with an accuracy of 92.45%. Alqahtani et al. created an effective botnet detection model based on an optimized XG Boost algorithm and feature selection techniques [30]. The Fisher score feature selection method is used to identify the optimum features, and the XG Boost algorithm parameters are optimized using a genetic algorithm. The suggested method attained a botnet detection rate of 99.67% by utilizing three data traffic attributes.

Roy et al. introduced an IDS using a novel B-stacking algorithm, which is a combination of boosting and stacking techniques. The author reduces the dimension of the feature using the auto-encoder technique. The KNN, RF, and SVM algorithms are used as base learners and combined using stacking with the XG Boost algorithm. The proposed model performed better with an accuracy of 99.5% for the CICIDS-2017 dataset [31]. Maleh et al. developed an IDS using optimized ML algorithms. The particle swarm optimization (PSO) algorithm is used to select the optimum attributes. The author concludes that RF-based IDS have better performance compared to other ML algorithms [32]. Liu et al. developed an anomaly detection system using PSO-based gradient descent algorithms. A one-class SVM algorithm is used to select the optimum attributes. From this study, we observe that the hybrid model has good robustness and generalization in detecting novel attacks [33].

Jiang et al.developed a NIDS using the PSO-XG Boost algorithm and the parameters are tuned using PSO. Regarding accuracy and recall, the suggested model performs better than other comparative models, especially when identifying attacks from minority groups [34]. In [35], Tama et al. developed a two-stage anomaly-based IDS using ensemble learning algorithms. The best features are chosen using a hybrid optimization technique, which is a combination of PSO, ant colony optimization (ACO), and genetic algorithm (GA). Hyper-parameters are tuned using a reduced error pruning tree algorithm. In the classification stage, rotation forests and bagging algorithms are combined using the voting concept. According to statistical significance tests, the

suggested method outperforms other cutting-edge individual classifiers and Meta classifier algorithms.

Mishra et al. developed an anomaly detection model using an LGBM algorithm with optimized hyper-parameters, where GA is used for parameter optimization. The proposed GA+LGBM performs better, with a detection accuracy of 99.99% for the DS2OS dataset [36]. Ghanem et al. developed an anomaly detection model using a multilayer perceptron (MLP) algorithm with novel feature selection and hyper-parameter tuning. BAT algorithm is used for feature selection and parameter tuning. The proposed BAT+MLP performs better with the detection accuracy of 98.05% for KDD CUP-99 and 99.16% for NSL KDD and 99.96% for ISCX 2012, and 97.63% for UNSW NB15 dataset [37].

In [38], Zivkovic et al. developed an intrusion classification model using the XG Boost algorithm with novel hyper-parameter tuning. A hybrid firefly algorithm is used for parameter optimization. In the comparative analysis, the proposed CFAEE-SCA model produced five percent higher detection accuracy than the conventional algorithm. In [39], Kunang et al. established an improved IDS using a deep learning algorithm with hyper-parameter tuning. The Bayesian optimization algorithm is used for parameter optimization. The proposed algorithm performs better with an accuracy of 99.99% for the BoT-IoT dataset. In [40], Song et al. developed an intrusion detection model using the XG Boost algorithm with hyper-parameter tuning. The whale optimization algorithm is used for parameter optimization. The proposed algorithm performs better, with a detection accuracy of 99.06% for the KDD CUP dataset. Verma et al. developed an ML-based IDS for sensing attacks against IPv6 RPL Networks. The suggested ensemble learning algorithm has a higher area under the curve (AUC) value than the existing techniques [41]. I A Zhan developed an efficient cyber-attack detection system using ML and DL algorithms for various domains like IIoT [42], Internet of Medical Things [43], Industrial control systems [44], [45], and autonomous vehicles [46].

Several limitations are identified in the existing anomaly detection model, as discussed in the literature survey. The bi-classification method used in many current studies can only determine whether the sample is an anomaly or legitimate. The single dataset is preferred in many models, making it impossible to prove model generalization. Besides these issues, many developed models were not verified statically by any of the significant tests. Moreover, ensemble learning algorithms like RF, Gr Boost, Ada Boost, and Light GBM are used as standalone classifiers rather than base classifiers in other ensemble models.

Researchers in this study have made valuable contributions to distinguish this study from earlier investigations. i) The behaviors of anomalies are identified better with the multi-classification technique ii) The robustness of the model is verified by testing the developed model with five different datasets. iii) Two-step anomaly detection model is implemented by integrating two different ensemble classifiers by the concept of stacking iv) Effectiveness is verified in the proposed model by chi-square statistical significance test.

III. PROPOSED ANOMALY DETECTION SYSTEM AND METHOD

This section explains the proposed ensemble-based anomaly detection model with several autonomous stages. Figure. 3 shows the proposed framework for anomaly detection. The Cooja simulated dataset is preprocessed using the widely available methods, and the synthetic minority oversampling technique (SMOTE) is used to balance the imbalanced anomaly dataset. The balanced data is trained using various ensemble learning algorithms and the performance is evaluated. Further, the parameters are tuned using the IGWO technique to enhance the learning algorithm's performance. The optimized learning algorithms are combined using a stacked concept for better anomaly prediction.

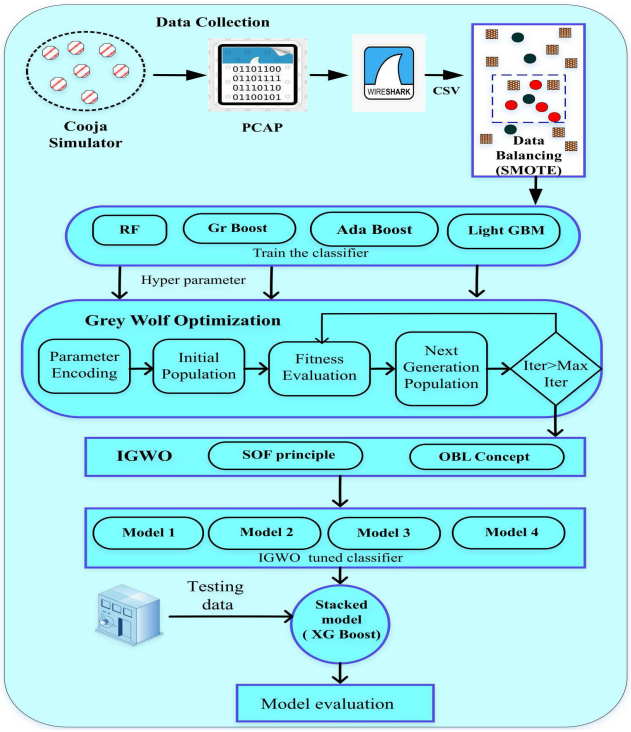

FIGURE 3. Proposed methodology.

A. DATASET GENERATION AND DESCRIPTION

Due to the lack of availability of open-source IoT anomaly datasets, we generate a new dataset using the Cooja simulator, part of the Contiki operating system. Cooja is an admirable RPL(routing protocol for low-power and lossy networks) simulator. Since it is an open-source software and easy to learn it is suggested for our model [47]. The impact of malicious insiders on RPL-based IoT sensor nodes is examined and the data are extracted in various scenarios in the simulation setup. The simulation time is fixed to one hour, and the motes outputs are displayed in the output window while simulations run, as shown in Figure. 4.

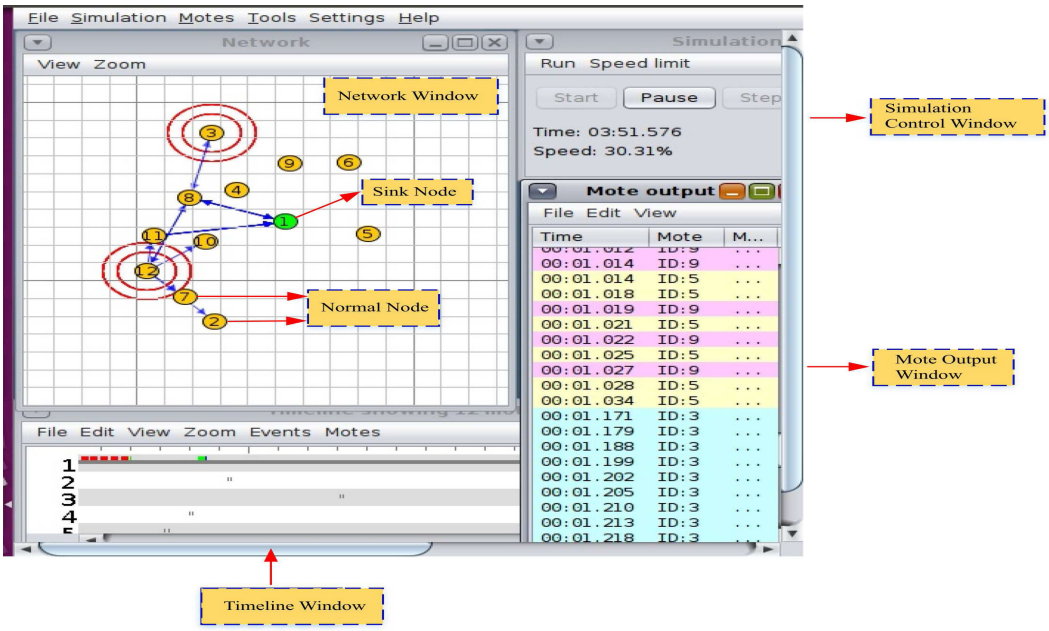

FIGURE 4. Cooja Scenario for creating the IoT environment.

TABLE 2. IoT framework parameter setting.

Index	Parameter	Representation
1	Simulator	Contiki-2.7 Cooja
2	Data communication range	200m × 200m
3	Node type	Sky mote
4	Data bytes	100−1000 bytes/cycle
5	Network layer	IPV6-ICMP-RPL
6	Number of nodes	10/1/2-Normal/Sink/Malicious
7	Network area	100m × 100m
8	Radio medium	Unit disk graph medium
9	Run time	1 hour
10	Application layer	IEEE 802.15.4
11	Transport layer	UDP
12	Topology	Random

TABLE 3. Various attack scenario for data collection.

S.No	Dataset	Benign	Malicious
1	Sinkhole(SH)	53,004	68,788
2	Block hole(BH)	53,004	69,454
3	DIS flood attack(FA)	53,004	53,038
4	Version attack(VA)	53,004	44,002
	Total	2,12,016	2,35,282

The dataset has five classes: normal, sinkhole attack, blockhole attack, DIS (DODAG Information Solicitation) flood attack, and version attack. Each category has a various number of legitimate and attack records. A "Hello" packet is send to their adjacent node with their identification number in the simulation. In attacks of types such as versions, DIS floods, sinkholes, and blackhole the attacked nodes send out DIS packets that make them appear as neighbors. The parameters for the dataset scenario are shown in Table 2. The Cooja simulator "Radio messages" tool could collect data about the network traffic features as a pcap (packet capture) file. Using wireshark software, the pcap file is exported to csv (comma-separated values) format. The details of the stated scenarios and collected samples are listed in Table 3. Table 4 shows the sample of the raw extracted dataset from the Cooja simulator. We elicit 15 network features from feature extraction, which is listed in Table 5.

B. DATA PRE-PROCESSING

Prior to the modelling stage, the acquired data should be evaluated and transformed into the proper formats if not, it will deteriorate the reliability of the model. With label encoding, distinguishing characteristics are represented by numerical integers. To increase the speed of the ML algorithm, a well-known Min-Max normalization is performed, and the original features (β_{max}, β_{min}) are shifted to the new interval (N_{max}, N_{min}). The formula is given by,

$$\beta_i' = N_{min} + (N_{max} - N_{min}) * (\frac{\beta_i - \beta_{min}}{\beta_{max} - \beta_{min}}) \quad (1)$$

C. DATA BALANCING

The real-time IoT data contain multiple classes, which are unbalanced in nature. The unbalanced samples in the class directly influence the performance of the classifier. Before creating an ideal ML-based prediction model, every data scientist's primary goal is to balance the dataset. Manokaran et al. empirically compared malware detection model performance using various balancing techniques and concluded that the SMOTE algorithm's performance is superior [48]. In SMOTE algorithm, each minimum class features μ gets its nearest neighbour from other minimum

TABLE 4. A sample of raw traffic dataset.

No	Time	Source	Destination	Protocol	Length	Info
3809	240.092	fe80::212:7408:8:808	fe80::212:7409:9:909	ICMPv6	102	RPL Control (DODAG Information Object)
3810	240.271	fe80::212:7408:8:808	fe80::212:7409:9:909	ICMPv6	102	RPL Control (DODAG Information Object)
3811	240.696	::212:7403:3:303	::1	UDP	97	8775 > 5688 Len=46
3812	240.852	::212:7403:3:303	::1	UDP	97	8775 > 5688 Len=46

TABLE 5. List of attributes extracted from Cooja simulation.

S.No	Features	Illustration
1	Number	Packet sequence number
2	nData	Number of Data
3	TRT	Total Reception Time
4	Time	Simulation time
5	Source	Source Mote IP
6	RR	Reception Rate
7	Destination	Destination Mote IP
8	TR	Transmission Rate
9	DIO	DIO Packets
10	Length	Length of the packet
11	Info	Packet information
12	TTT	Total Transmission Time
13	DAO	DAO Packets
14	DIS	DIS Packets
15	Label	Benign/Malicious

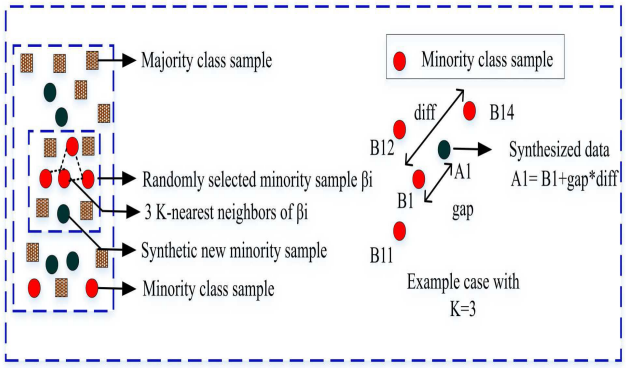

FIGURE 5. Synthetic minority sample creation using SMOTE algorithm [49].

class features. Similarly, another feature $\overline{\mu}$ get its K nearest neighbour. Finally, the artificial features μ_{new} is generated from μ, $\overline{\mu}$, and a random value (rand) is shown in Equation (2).

$$\mu_{new} = \mu + rand(0, 1) \times (\overline{\mu} - \mu) \tag{2}$$

For the implementation of the SMOTE algorithm, 5000 sample points were taken. The proportion of minority to the majority sample are 1:10, 1:50, and 1:100. Figure 5 shows the working principle of the SMOTE algorithm. In our SMOTE technique, the closest k value is chosen as three, and Figure. 6 displays the sample of data before and after SMOTE oversampling.

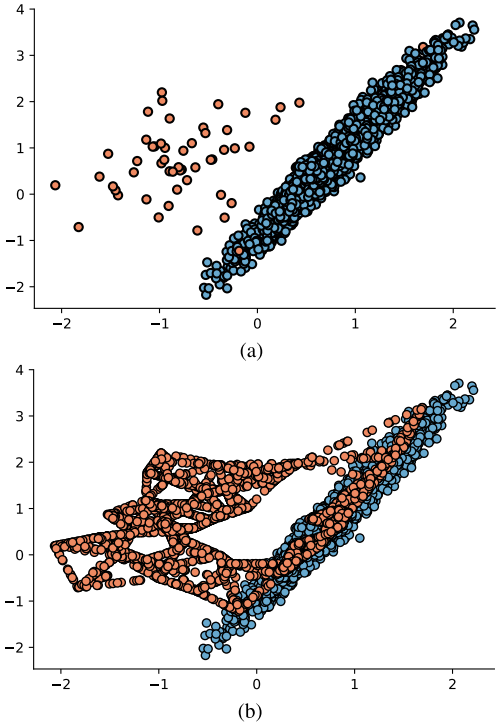

FIGURE 6. (a) Data point spreading before SMOTE (b) Data point spreading after SMOTE.

D. CLASSIFICATION ALGORITHMS

The literature review shows that the ensemble learning algorithms perform better in prediction accuracy than the shallow learning algorithms. We use dual stage ensemble algorithm for anomaly detection. Random forest, Ada boost, Gr Boost, and Light GBM algorithms are used as base learners, and the XG Boost algorithm is used as a Meta learner. Various ensemble algorithms used for our classification purposes are explained below. We employ all the ensemble learning algorithms (RF, Gr Boosting, Ada Boosting, and Light GBM) in the sklearn package of the Python environment.

1) RANDOM FOREST (RF)

The RF algorithm is an efficient ML algorithm where multiple DT are fused using the bagging concept as shown in Figure. 7 [50]. The detection of the RF algorithm is faster and more effective in various cases than the other boosting techniques. The RF algorithm's significant advantage is that it does not need a cross-validation process to train all the

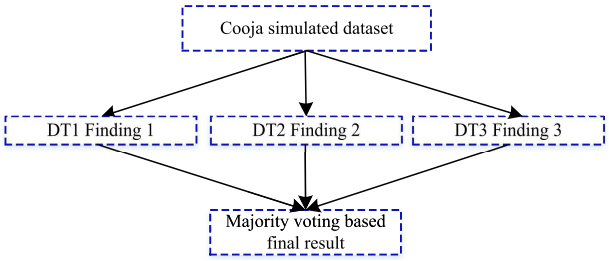

FIGURE 7. Random forest working principle.

Algorithm 1 RF Algorithm

Input: Cooja simulated dataset
Output: Categorized data
Step 1: Select X number of trees to grow
Step 2: Select v number of variables used to separate each feature points. v $\ll$ V, Where V is the number of input attributes
Step 3: Develop the tree structure with the following operation
Step 4: Total M training samples are replaced with a portion of new sample size M.
Step 5: To develop a tree at every node select v variables arbitrarily from V and use them to detect the optimum split.
Step 6: Extend the tree to a maximum level further it has no pruning.
Step 7: To classify point P gather majority vote of every tree in the forest and find the final result.

samples. The bootstrap sample is used to construct each tree, so approximately one-third of the cases are left out and are not used in training.

2) GRADIENT BOOSTING (GR BOOST)

The Gradient Boosting algorithm is a flexible homogeneous ensemble learning algorithm that sequentially combines multiple weak classification and regression trees (CART). The primary objective of the gradient is to reduce the loss function. It is an iterative algorithm for every stage; the error value is calculated and compared with the previous steps. Based on the error value, a different loss function is calculated. Given a Cooja dataset X with s sample points and n variables $X = (x_i, y_i)$ $(|X| = s, x_i \in M^s, y_i \in M)$, a tree ensemble perform Γ additive function to detect the anomaly [51]. Figure. 8 shows the structure of the Gr Boosting algorithm.

$$\widehat{y_i} = Q(x_i) = \sum_{l=1}^{\Gamma} f_l(x_i),\ f_l \in \lambda \tag{3}$$

where the CART space function,

$$\lambda = \left\{\alpha_{p(x)}\right\} (p : M^s \to \tau, \alpha \in M^\tau) \tag{4}$$

where p denotes the configuration of each tree, τ denotes tree size, f_l is a tree configuring p and leaf weight denotes as α.

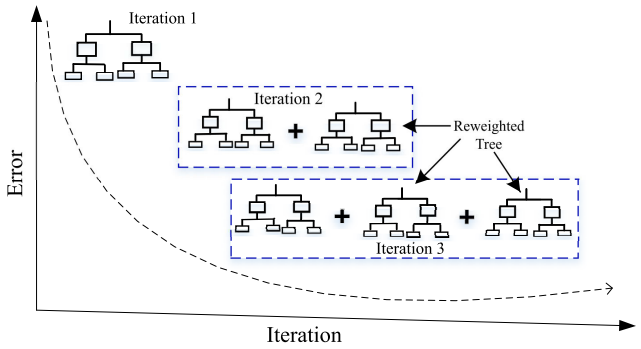

FIGURE 8. Gradient Boosting working principle.

3) ADAPTIVE BOOSTING (ADA BOOST)

Adaptive boosting is an efficient ensemble learning algorithm. The accuracy of the ada boost algorithm is improved by combing multiple weak classifiers (Decision tree). Ada boost trains the data sample to predict unusual observations accurately and adjust classifier weights in each iteration. Initially, the sample weight is given by $W = 1/\eta$, where η is the record number. The model is generated by combining several stomps that is one stage decision tree. For each feature, a stomp will be created to classify the data. The stomps which have less entropy value is selected as base learner. The total error (TE) of the stomps is calculated in each iteration. Next, the performance of the stomps is calculated using Equation (5). Update the weight based on the performance of the decision tree in the previous stage. The weight update formula is shown in Equation (6,7). Normalize the weight and sequentially execute all the decision tree algorithm. The weak learner is combined to produce a robust learning algorithm with less error [52]. Figure. 9 shows the structure of the Ada Boosting algorithm.

$$Performance\ of\ stomps(P) = \frac{1}{2}\log_e\left[(1 - TE)/TE\right] \tag{5}$$

$$Weight_{new} = Weight_{old} * e^{P} \quad \{for\ incorrect\ classified\ point\} \tag{6}$$

otherwise,

$$Weight_{new} = Weight_{old} * e^{-P} \quad \{for\ correct\ classified\ point\} \tag{7}$$

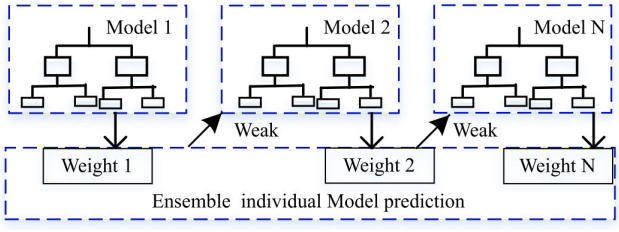

FIGURE 9. Ada Boost working principle.

4) LIGHT GBM

The Light GBM implementation uses leaf-wise algorithms and histograms to optimize prediction accuracy and computation power. Mutually incompatible feature pairs are fused using a histogram approach. The continuous attributes are discretized into m integers before constructing an m-width histogram. As a result, the histogram's discretized values are used to test the training data. Histograms reduce the complexity of time significantly. Light GBM identifies the leaf with the most significant splitting gain and divides it leaf-by-leaf. In this way, Light GBM ensures high efficiency and eliminates inefficiency by limiting the leaf-wise depth. Currently, tree splitting takes place with the use of information gain, but in LGBM, gradient-based one-side sampling is used to find the variance gain [53]. Let D be the anomaly detection dataset. The training examples' absolute gradient values are initially sorted in descending order, and the top $a \times 100\%$ data samples of gradient values, designated B, are then selected. B subset E of size $b \times |B^c|$ is then randomly selected from the remaining samples B^c. The variance gain of the decision tree node split attribute f at point P is denoted as follows,

$$V_f(P) = \frac{1}{m}\left(\left[\frac{\left(\sum_{x_i \in B_l} \alpha_i + \frac{1-a}{b}\sum_{x_i \in E_l} \alpha_i\right)^2}{n_l^j(p)}\right] + \left[\frac{\left(\sum_{x_i \in B_r} \alpha_i + \frac{1-a}{b}\sum_{x_i \in E_r} \alpha_i\right)^2}{n_r^j(p)}\right]\right) \quad (8)$$

where $B_l = \{X_i \in B : X_{ij} \leq P\}$, $B_r = \{X_i \in B : X_{ij} \succ P\}$, $E_l = \{X_i \in B : X_{ij} \leq P\}$, $E_r = \{X_i \in E : X_{ij} \succ P\}$, α is a negative gradient of loss function, $\frac{1-a}{b}$ - Normalized sum of gradient.

5) XG BOOSTING

XG Boost is known as "Extreme Gradient Boosting", an improved gradient-boosting decision tree that is extensible and highly scalable. This method constructs classifiers and regression trees simultaneously using boosted regression. In this way, the objective function's value can be optimized. XG Boost measures the coverage and frequency of a selected feature's impact on a function's output. XG Boost uses additive training optimization, where each next iteration depends on the previous one [53]. The method used to calculate the objective function in the i^{th} iteration illustrates this:

$$g_j = \partial_{\bar{O}k^{i-1}} l\left(O_j, \bar{O}k^{i-1}\right) \quad (9)$$

$$h_j = \partial^2_{\bar{O}k^{i-1}} l\left(O_j, \bar{O}k^{i-1}\right) \quad (10)$$

$$W_j^* = -\frac{\sum g_t}{\sum h_t + \beta} \quad (11)$$

$$R(f_i) = \lambda T_i + \frac{\beta}{2}\sum_{j=1}^{T} W_j^2 \quad (12)$$

$$F_o^i = \sum_{k=1}^{n} l\left(O_k, \bar{O}_k^{i-1} + f_i(x_k)\right) + R(f_i) + C \quad (13)$$

where g, h are the 1^{st} and 2^{nd} derivatives, C is the constant, R is the system formalization term, λ, β are the tree configuring parameter. F_o^i is the i^{th} iteration objective function, and l is the loss term.

E. GREY WOLF OPTIMIZATION (GWO)

GWO is a meta-heuristic optimization technique inspired from grey wolves and proposed by Mirjalili [22]. GWO simulates the hierarchical ordering and attacking behavior of wolves. The grey wolves are ordered into four types: alpha (α), beta (β), delta (δ), and omega (ω). The three main steps of attacking implemented in GWO are seeking out the prey, surrounding the prey, and attacking the prey.

Let the position of the four wolves are expressed in examination space as X_α, X_β, X_δ, X_ω [54]. The first major step is prey encircling, i.e.; the wolf surrounds the food. The mathematical representations are shown in equations (14)-(18).

$$\vec{D} = \left|\vec{C} \cdot \vec{X}_{prey}(t) - \vec{X}(t)\right| \quad (14)$$

$$\vec{X}(t+1) = \vec{X}_{prey}(t) - \vec{A} \cdot \vec{D} \quad (15)$$

where $\vec{X}(t)$, $\vec{X}_{prey}(t)$ denotes the position vector of wolves and prey at present iteration. $\vec{A}$, $\vec{C}$ are coefficient vectors, computed as follows,

$$\vec{A} = 2\vec{a} \cdot r_1 - \vec{a} \quad (16)$$

$$\vec{C} = 2 \cdot r_2 \quad (17)$$

$$\vec{a} = 2 - \left(\frac{2 * t}{Iter_{max}}\right) \quad (18)$$

r_1, r_2 denotes random value between [0, 1]. $\vec{a}$ is a vector whose value reduces over the iteration, form two to zero. These three wolves have information of the likely prey zone, allowing for the development of three excellent search agents and the subsequent update of the positions of other wolves as shown in equations (19)-(23).

$$\vec{D}_\alpha = \left|\vec{C}_1 \cdot \vec{X}_\alpha - \vec{X}\right| \quad (19)$$

$$\vec{D}_\beta = \left|\vec{C}_2 \cdot \vec{X}_\beta - \vec{X}\right| \quad (20)$$

$$\vec{D}_\delta = \left|\vec{C}_3 \cdot \vec{X}_\delta - \vec{X}\right| \quad (21)$$

$$\vec{X}_1 = \vec{X}_\alpha - \vec{A}_1 \cdot \vec{D}_\alpha,\ \vec{X}_2 = \vec{X}_\beta - \vec{A}_2 \cdot \vec{D}_\beta,\ \vec{X}_3 = \vec{X}_\delta - \vec{A}_3 \cdot \vec{D}_\delta \quad (22)$$

$$\vec{X}_{t+1} = \frac{\vec{X}_1 + \vec{X}_2 + \vec{X}_3}{3} \quad (23)$$

The next position is determined by its recent position and the prey location. The position updating of wolf is shown in Figure. 10.

Researchers usually abstract GWO as a random search problem in multi-dimensional space, intending to optimize the objective function. For each wolf in the group, fitness

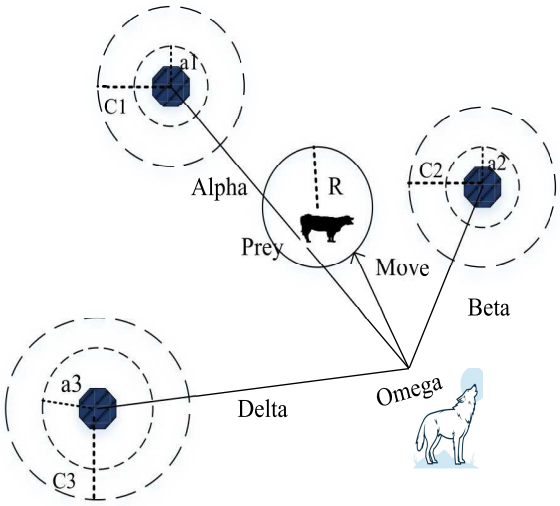

FIGURE 10. Position updating of GWO [54].

value can be calculated according to fitness function to evaluate the fitness of the wolf. The accuracy value is considered as a fitness function which is shown in equation (24), where the parameter terms are explained in section IV.

$$F(x) = Accuracy = \frac{tp + tn}{tp + tn + fp + fn} \tag{24}$$

1) IMPROVED GREY WOLF OPTIMIZATION

Though GWO is good at solving many optimization problems, it can reduce the probability of prematuring and prevent it from falling into the local best value. To balance exploration and exploitation properly, we propose an IGWO algorithm over the traditional GWO algorithm. It has two modifications like i) the SOF principle and ii) the OBL method. First, the wolves are arranged depending on the fitness value after each iteration. The wolves with the most negligible fitness value are removed, and new wolves are generated based on the OBL method.

Exploration and exploitation are the two major components of any optimization method. An algorithm can produce better outcomes in locating global optima and producing superior convergence profiles by carefully balancing the effects of the two occurrences. Currently, the notion of OBL has been utilized to achieve the same. Instead of generating random wolf for the replacement of least fitness wolf, the wolves are generated based on the OBL method to overcome the problem of computational expensiveness and time consumption. The placements of the grey wolves in GWO have a major impact on the solutions' outcomes and the convergence rate. OBL helps the algorithm to change the search positions repeatedly by generating opposing random numbers to provide diversity to the original population and locate fresh solutions. Let $r \in [x, y]$ be a real number. Its opposite $\vec{r}$ is represented as follow:

$$\vec{r} = x + y - r \tag{25}$$

This principles are also applicable to multi-dimensions. Let $O = \{r_1, r_2, \ldots r_D\}$ be a point in the D-dimensional search space, and $r_i \{1 \leq i \leq D\}$ is a range of x_i and y_i. The opposite point $\vec{O} = \{\vec{r}_1, \vec{r}_2, \ldots \vec{r}_D\}$ can be obtained using equation (26)

$$\vec{r}_i = x_i + y_i - r_i \tag{26}$$

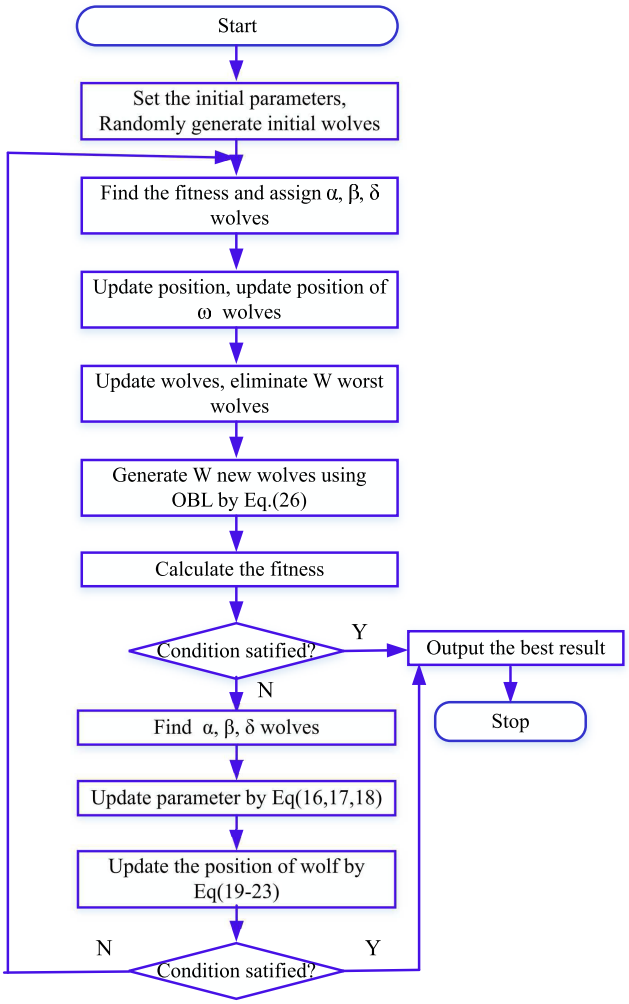

FIGURE 11. Flowchart of the proposed IGWO algorithm.

The primary flowchart of the IGWO algorithm is shown in Figure 11. The procedures of the developed IGWO algorithm is shown in Figure 12.

2) COMPLEXITY INVESTIGATION

This section discusses the IGWO algorithm's computational complexity. Assume that PS is the wolf population size, D_{dim} is the dimension of the problem, and $Iter_{max}$ is the maximum number of iterations. The worst W wolves are removed after each cycle, and W new wolves are created following the OBL principle. The OBL evaluation function needs O(PS), the wolf selection process needs O(PS), and the position-updating technique needs O(PS $\times$ D_{dim}). As a result, each iteration's overall computational complexity is O(PS $\times$ D_{dim}). O(PS $\times$ D_{dim} $\times$ $Iter_{max}$) is the computational complexity of the entire iteration. Therefore,

Proposed IGWO algorithm

Initialize the positions of the wolves, X_i (i=1, 2,…, n)
Initialize GWO parameters $\vec{a}$, $\vec{A}$, and $\vec{C}$
Compute the fitness of each search agent
X_α, X_β, X_δ - the first 3 best search agent
While (t < $Iter_{max}$)
  For each individual **do**
    Update the position of the current individual by equation (23)
  end
  Update $\vec{a}$, $\vec{A}$, and $\vec{C}$
  Calculate the fitness of each individual
  For (i=0; i< W_{best}; i++)
    For (j=0; j<D; j++)
      Eliminate W worst wolves
      Generate W new wolves using equation (26)
    end
  end
  Evaluate the fitness function
 While (t<$Iter_{max}$)
   Find X_α, X_β, X_δ
   Update $\vec{a}$, $\vec{A}$, and $\vec{C}$ using equation (16 - 18)
   For (i=0; i< W_{best} ; i++)
     Update the position using equation (19 - 23)
   end
   t=t+1
 end
end
Return X_α

FIGURE 12. Pseudo-code of Proposed IGWO algorithm.

the computational complexity has not increased and it is consistent with the traditional GWO [55].

F. STACKED ENSEMBLE LEARNING

Ensemble learning combines ML algorithms with bagging, stacking, and boosting principles. Due to its generic nature, stacking acts as an efficient ensemble approach. There are bi-level of learners, base learner, and meta-learner. The base learners are trained using training data. The meta-learner creates a new dataset after training the base learners. The meta-learner is then trained using new training data. The test set is categorized using the trained meta-learner. Choosing the best base learner is an essential component of stacking [56].

This paper has used a stacked ensemble model in which IGWO-tuned ensemble learning algorithms such as RF, Ada Boost, Gr Boost, and LGBM are base learners. The XG Boost algorithm is used as a meta-classifier. Compared to most of the existing ensemble models, we have selected a better base learners and a better meta learner. The steps of the stacking ensemble algorithm is provided in Algorithm 2.

IV. PERFORMANCE EVALUATION

This section evaluates the proposed ensemble-based anomaly detection model using standard parameters such as accuracy, precision, recall, ROC (receiver operating characteristic) curve, and F1 score. The proposed model falls under the multi-classification problem, which has more than two class labels. There are five distinct entries in the class label, where the first four entries represent attack data, and the fifth entry represents normal data in an IoT environment. Figure. 13 (a) illustrates the confusion matrix for multi-classification with

Algorithm 2 Stacked of Ensemble algorithm

Input: A training anomaly dataset X with Features set F = {A1, A2, …, An}
Output: Classified dataset
Step 1:
  Train the level 1 ensemble learning algorithm,
  Number of level 1 classifier = 4
Step 2:
  Learn the Random forest algorithm (X, F, $Tree_{size}$)
    G=β
  for i=1 to $Tree_{size}$ do
    $T^{(i)} = X_i$
  for i=1 to $T^{(i)}$ do
    t= subset of total attributes
    r_i=best attributes of t
    $G \leftarrow G \cup r_i$
  end for
Step 3:
  Learn gradient boosting, LGBM algorithm (X, F, $Tree_{size}$)
    Assign $Tree_{size} = \{50, 100, 150, 200, 250\}$
    $f_0(c) = argmin \sum_{i=1}^{m} (\chi - d_i)$
    Modified the model based on M number of target values.
  for i=1 to m do
    $f_i(c) = f_{i-1}(c) + g_{i-1}(c)$
    find residue $g_i(c)$
    $g_i(c) = expected\ value - orginal\ vaule$
  end for
Step 4:
  Learn Ada Boost algorithm (X, F)
    Assign number of iterations I
    Initial weight $\lambda_i \in \{1, 2, .n\}$ $X'(i) = 1/n$
  for i=1 to I do
    Train the basic weak learners
    Select the proper weight
    Update the distribution over the data
    $X_{t+1}(i) = \frac{X_t(i).e^{-\lambda_i y_i h_t(x_i)}}{Z_t}$
    Z_t - Normalization factor
    Final vote weight $V(x) = sign \sum_{t=1}^{I} \lambda_i h_t(x_i)$
  end for
Step 5:
  Generate new dataset of detections to Meta classifier
  for i=1 to n do
    $N_h = (\alpha'_1, \beta_1)$, where $\alpha'_1 = \{h_1(\alpha'_1) \ldots h_n(\alpha'_1)\}$
  end for
Step 6:
  Learn Meta classifier (M)
  Learn M based on N_h
  Predict $\alpha'_1 \ldots \alpha'_n$ as anomaly or normal

five class labels. Here Z_{11} to Z_{44} represent the number of anomaly data predicted as anomaly, Z_{55} represents the number of normal data envisioned as normal. All the

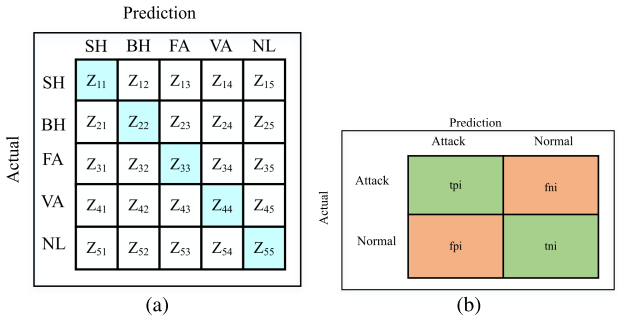

FIGURE 13. (a) Confusion matrix for anomaly of 5 label classification problem (b) Confusion matrix for each end every class label (i).

evaluation metrics are calculated from the confusion matrix, which has four parameters tp (True positive), fp (False positive), tn (True negative), fn (False negative) represented in Figure. 13 (b) for each and every class label (i=1 to c). Similarly, we have to calculate the parameters: average accuracy, error rate, precision, recall, and F-score for micro and macro averaging for each and every class (c = 1 to 5) in our five classification problems through the following equations (Equation 27 to 34) with $\beta = 1$ [57].

$$Average\ accuracy = \frac{\sum_{i=1}^{c} \frac{(tp_i+tn_i)}{tp_i+tn_i+fn_i+fp_i}}{c} \tag{27}$$

$$Error\ rate = \frac{\sum_{i=1}^{c} \frac{(fp_i+fn_i)}{tp_i+tn_i+fn_i+fp_i}}{c} \tag{28}$$

$$Precision_{Macro} = \frac{\sum_{i=1}^{c} \frac{(tp_i)}{tp_i+fp_i}}{c} \tag{29}$$

$$Recall_{Macro} = \frac{\sum_{i=1}^{c} \frac{(tp_i)}{tp_i+fn_i}}{c} \tag{30}$$

$$F-score_{Macro} = \frac{(\beta^2+1)Precision_{macro} * Recall_{macro}}{\beta^2 * Precision_{macro} + Recall_{macro}} \tag{31}$$

$$Precision_{micro} = \frac{\sum_{i=1}^{c} (tp_i)}{\sum_{i=1}^{c} (tp_i + fp_i)} \tag{32}$$

$$Recall_{micro} = \frac{\sum_{i=1}^{c} (tp_i)}{\sum_{i=1}^{c} (tp_i + fn_i)} \tag{33}$$

$$F-score_{micro} = \frac{(\beta^2+1)Precision_{micro} * Recall_{micro}}{\beta^2 * Precision_{micro} + Recall_{micro}} \tag{34}$$

A. EXPERIMENTAL SETUP AND ANALYSIS

The proposed system effectiveness is analyzed using various experimental setups such as in setup I the performance of various base ensemble learning algorithm is compared with the parameter of accuracy, precision, recall, and F-score value for different datasets. In setup II, the base ensemble learning algorithms are optimized using the IGWO technique and combined with the stacking algorithm for the performance evaluation. In setup III, the developed model performance is compared with the recent methods.

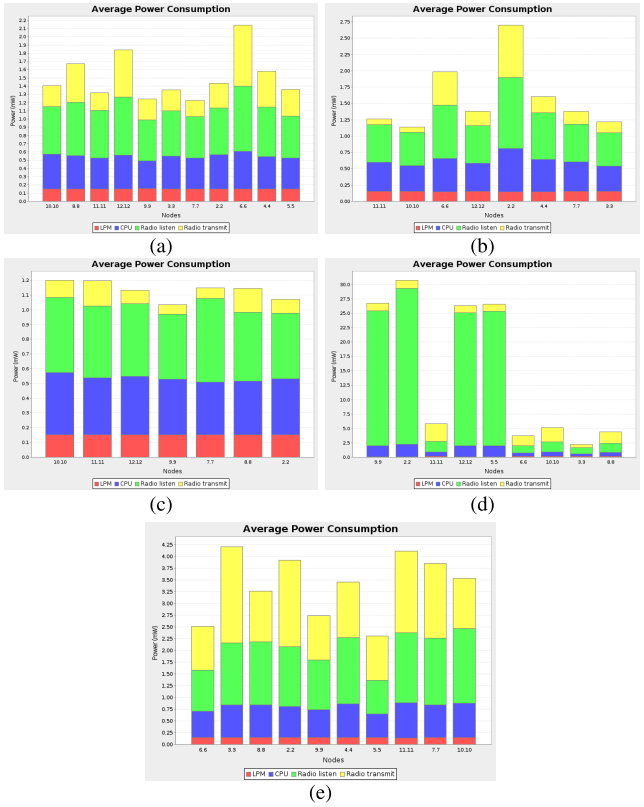

FIGURE 14. (a) Power consumption of normal scenario (b) Sinkhole attack (c) Blockhole attack (d) DIS flood attack (e) Version attack.

Python-based experiments are conducted on an Intel(R) Core(TM) i5 CPU processor running 64-bit Windows 10 and 16.00 GB of RAM in order to evaluate the proposed approach. Twenty runs of experiments are conducted, and the findings are averaged. We analyze the effect of different attacks on Cooja simulated dataset, as shown in Figure. 14.

Figure. 14(a) shows the power consumption of a normal scenario with 11 nodes. Figure. 14(b) shows the average power consumption of the sinkhole attack scenario; here, the node's rank will be decreased to eight, and their packets will also drop. Figure. 14(c) shows the average power consumption of the block hole attack scenario; here, the node's rank will be decreased to seven, and their packets will also drop. Figure. 14(d) shows the average power consumption of the DIS flood attack scenario; from the figure, the power consumption is 30 times higher than the normal scenario, which negatively affects IoT nodes. Figure. 14(e) shows the average power consumption of the version attack scenario; from the figure, the attacks deteriorate the node resource and increase the power consumption. From Figure. 14, the attack scenario's average power consumption is higher than the normal scenario.

1) SETUP I

The performance of the base ensemble learning algorithm before optimization is shown in Tables 6-10. Table 6 shows the interpretation of the Cooja simulated dataset

TABLE 6. Performance of the Cooja dataset with multi-classification techniques.

Classifier	RF	GR Boost	Ada Boost	Light GBM
Average accuracy	98.61	93.36	93.20	98.71
Error rate	1.39	6.64	6.8	1.29
Precision (Macro)	95.71	79.92	79.48	95.87
Recall (Macro)	95.12	79.36	78.94	95.77
F-score (Macro)	95.42	79.64	79.21	95.82
Precision (micro)	96.53	82.40	83.01	96.79
Recall (micro)	96.53	82.40	83.01	96.79
F-score (micro)	96.53	82.40	83.01	96.79

TABLE 7. Performance of the NSL KDD dataset with multi-classification techniques.

Classifier	RF	GR Boost	Ada Boost	Light GBM
Average accuracy	98.93	98.81	86.72	98.84
Error rate	1.07	1.19	13.28	1.16
Precision (Macro)	97.57	97.44	57.75	97.47
Recall (Macro)	94.72	94.53	67.59	94.58
F-score (Macro)	96.12	95.96	62.28	96.00
Precision (micro)	97.33	97.02	66.79	97.10
Recall (micro)	97.33	97.02	66.79	97.10
F-score (micro)	97.33	97.02	66.79	97.10

TABLE 8. Performance of the UNSW NB 15 dataset with multi-classification techniques.

Classifier	RF	GR Boost	Ada Boost	Light GBM
Average accuracy	93.41	92.64	91.93	93.29
Error rate	6.59	7.36	8.07	6.71
Precision (Macro)	66.73	63.45	59.32	66.50
Recall (Macro)	66.13	62.01	52.82	65.87
F-score (Macro)	66.42	62.72	55.88	66.18
Precision (micro)	67.05	63.23	59.96	66.44
Recall (micro)	67.05	63.23	59.96	66.44
F-score (micro)	67.05	63.23	59.96	66.44

with base ensemble classifiers (RF, Gr Boost, Ada Boost, and Light GBM). Light GBM has the highest detection accuracy of 98.71%, and the RF algorithm also has a good detection accuracy of 98.61%. Ada Boosting and Gr Boosting algorithms have a lower detection accuracy in the range of 93%. Tables 7 and 8 show the performance of NSL KDD and UNSW NB-15 datasets with base ensemble classifiers. For the NSL KDD and UNSW NB 15 datasets, the RF algorithm outperforms the other algorithms with the detection accuracy of 98.93% and 93.41%, respectively.

Table 9 shows the performance of the CICIDS 2017 dataset with base ensemble classifier. The CICIDS 2017 dataset is large and has unbalanced feature classes. Here also, the RF algorithm performs better with the highest detection accuracy of 99.13%. Even on the remaining parameters (precision, recall, F1-score), the performance of the RF and LGBM is found to be superior. Table 10 shows the performance of the MQTTset dataset, a new dataset with the most recent attack types. In this instance as well, the RF algorithm has the highest performance metrics compared to the remaining

TABLE 9. Performance of the CICIDS 2017 dataset with multi-classification techniques.

Classifier	RF	GR Boost	Ada Boost	Light GBM
Average accuracy	99.13	98.90	98.15	99.07
Error rate	0.87	1.10	1.85	0.93
Precision (Macro)	86.01	80.98	75.31	88.23
Recall (Macro)	86.07	77.89	75.51	85.69
F-score (Macro)	86.04	79.40	75.41	86.94
Precision (micro)	96.94	96.16	93.53	96.73
Recall (micro)	96.94	96.16	93.53	96.73
F-score (micro)	96.94	96.16	93.53	96.73

TABLE 10. Performance of the MQTTset dataset with multi-classification techniques.

Classifier	RF	GR Boost	Ada Boost	Light GBM
Average accuracy	99.25	99.01	98.27	99.14
Error rate	0.75	0.99	1.73	0.86
Precision (Macro)	97.08	95.71	87.29	96.68
Recall (Macro)	91.67	91.43	86.08	94.64
F-score (Macro)	94.30	93.52	86.68	95.65
Precision (micro)	97.74	97.03	94.82	97.42
Recall (micro)	97.74	97.03	94.82	97.42
F-score (micro)	97.74	97.03	94.82	97.42

algorithms. Figure. 15 shows the performance comparison of the base ensemble classifier with default parameter settings. Though the ensemble learning algorithm provides better prediction accuracy, there is a need to improve its detection accuracy. Therefore, we perform two operations i) parameter optimization by IGWO ii) stacking by combining all the optimized ensemble classifiers. It is more comprehensively explained in Setup II.

2) SETUP II

In Setup II, the base ensemble learning algorithms are optimized using the IGWO technique and combined with the stacking algorithm for performance enhancement. The parameter settings of GWO algorithm are shown in Table 11. Figure. 16 shows the different possible values of hyper-parameters and selected optimal parameter of ensemble learning algorithm using IGWO for five different datasets, namely Cooja (D1), NSL KDD (D2), UNSW NB-15 (D3), CICIDS 2017 (D4), and MQTTset (D5).

TABLE 11. Parameter settings of GWO techniques.

S.No	Items	Values
1	Number of populations	12
2	Number of iterations	100
3	Number of groups	03(α, β, δ)
4	Range of search space	[0,1]
5	Stopping criteria	Max. Iteration

a: DETERMINING THE OPTIMAL GWO ITERATIONS

Finding the maximum number of iterations necessary to get a stable and low error rate is a key parameter in the

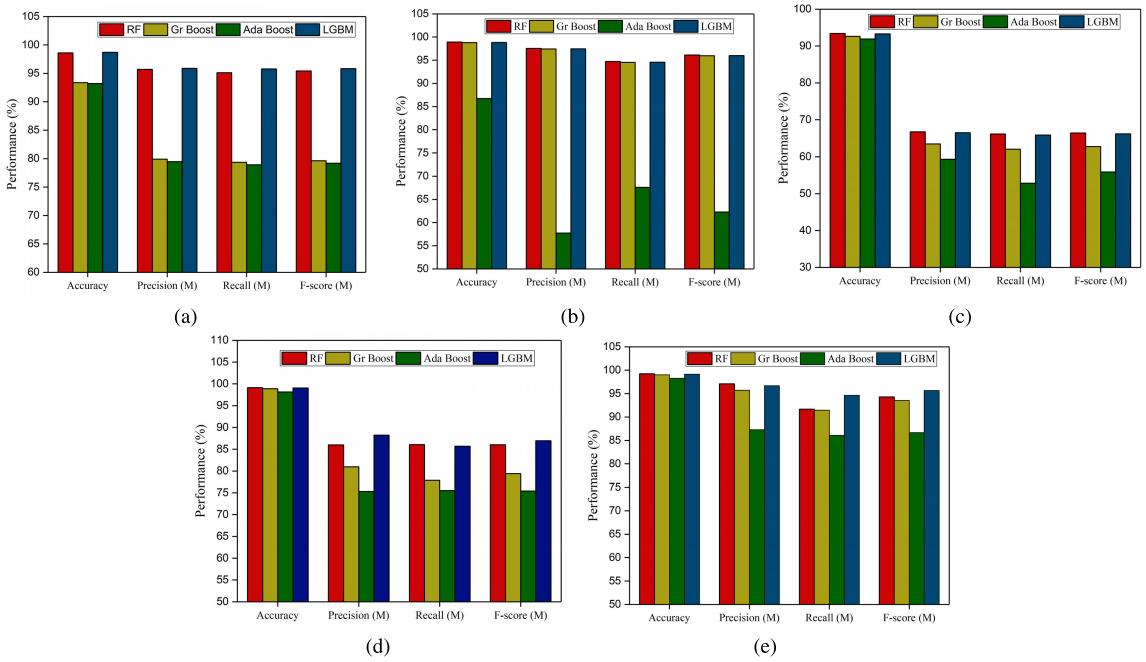

FIGURE 15. Performance comparison using base ensemble learning algorithms (a) Cooja (b) NSL KDD (c) UNSW NB 15 (d) CICIDS 2017 (e) MQTTset.

Base model	Hyperparameters	Range value	Best parameter value				
			D1	D2	D3	D4	D5
RF	criterion	{gini, entropy}	entropy	gini	gini	gini	entropy
	max_depth	{1,2,3…..25}	9	12	10	12	12
	col_sample_rate_per_tree	{0.1,0.2,…1}	0.68	0.84	0.73	0.84	0.84
	n_trees	{100,200,…600}	500	500	500	100	400
	col_sample_rate_chg_per_level	{0.8,0.9,…..1.2}	1	1.02	1.07	1.1	1.1
	min_rows	{2,4,8,…1024}	32	32	16	32	32
	nbins	{2,4,8,…1024}	32	32	512	32	64
	nbins_cats	{2,4,8,…1024}	64	32	256	32	512
Gr Boosting	subsample	{0.1,0.2,…1.00}	0.44	0.56	0.57	0.64	0.5
	max_depth	{1,2,3…..25}	24	8	12	7	8
	col_sample_rate_per_tree	{0.1,0.2,…1.00}	0.7	0.82	0.66	0.89	0.82
	col_sample_rate_chg_per_level	{0.8,0.9,…..1.2}	1	1.02	1.06	1.02	1
	n_trees	{100,200,…600}	100	500	500	450	400
	min_rows	{2,4,8,……1024}	16	64	512	2	4
	nbins	{2,4,8,……1024}	32	256	32	16	16
	nbins_cats	{2,4,8,……1024}	256	16	1024	128	64
Ada Boosting	n_estimator	{100,200,…1000}	200	500	500	700	700
	learning rate	{0.01,0.02,….0.5}	0.02	0.12	0.2	0.1	0.1
	n_trees	{100,200,…600}	200	500	500	300	500
LGBM	max_bin	{100,200,…1000}	100	100	100	100	100
	data_in_leaf	{200,300…900}	600	700	700	800	800
	n_trees	{100,200,…600}	200	500	500	400	500
	max_depth	{2,4,……..24}	6	16	4	10	8
	learning_rate	{0.01,0.02,…0.5}	0.06	0.13	0.2	0.1	0.1
	lambda_l1	{10,20,….100}	10	30	20	40	40
	lambda_l2	{10,20,….100}	60	70	100	100	100
	num_iterations	{100,200,500}	500	300	300	400	400

FIGURE 16. The representation of hyper-parameters search and the optimum values of every dataset.

prediction performance of ML models. To ensure that the model performs properly throughout numerous iterations, it is essential to test it extensively.

Figure. 17 plots the minimum error rate evolution along with the number of iterations for the proposed model with different datasets. The figure shows that the error rate decreases when the number of iterations increases for all the datasets. The optimal iteration point for Cooja, NSL KDD, UNSW NB 15, CICIDS 2017, and MQTTset datasets are 28, 50, 30, 58, and 60, respectively.

TABLE 12. Performance of the Cooja simulated dataset with optimized classifier.

Classifier	IGWO-RF	IGWO-GR Boost	IGWO-Ada Boost	IGWO-Light GBM	Proposed SoE
Average accuracy	99.11	97.84	97.22	99.32	99.44
Error rate	0.89	2.16	2.78	0.68	0.56
Precision (Macro)	97.37	93.22	91.09	98.10	98.41
Recall (Macro)	96.90	93.55	92.85	97.70	98.06
F-score (Macro)	97.14	93.39	91.86	97.90	98.24
Precision (micro)	97.76	94.60	93.05	98.30	98.59
Recall (micro)	97.76	94.60	93.05	98.30	98.59
F-score (micro)	97.76	94.60	93.05	98.30	98.59

TABLE 13. Performance of the NSL KDD dataset with optimized classifier.

Classifier	IGWO-RF	IGWO-GR Boost	IGWO-Ada Boost	IGWO-Light GBM	Proposed SoE
Average accuracy	99.30	99.27	92.48	99.34	99.60
Error rate	0.70	0.73	7.52	0.66	0.40
Precision (Macro)	95.13	95.08	68.33	95.17	98.07
Recall (Macro)	95.11	95.07	68.94	95.27	97.15
F-score (Macro)	95.12	95.07	68.64	95.22	97.61
Precision (micro)	98.25	98.16	81.20	98.34	99.01
Recall (micro)	98.25	98.16	81.20	98.34	99.01
F-score (micro)	98.25	98.16	81.20	98.34	99.01

The performance of the optimized ensemble learning algorithms and the proposed stack of ensemble algorithms are shown in Tables 12-16. Table 12 shows the performance of the Cooja simulated dataset with optimized ensemble classifier (IGWO-RF, IGWO-Gr Boost, IGWO-Ada Boost, IGWO-Light GBM) and the proposed stack of ensemble

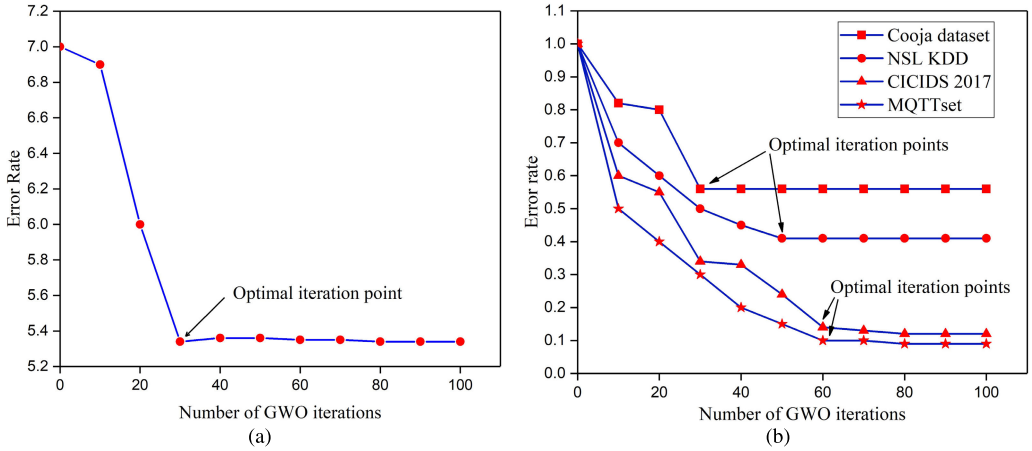

FIGURE 17. Error rate convergence with iteration (a) UNSW-NB15 (b) Cooja dataset, NSL KDD, CICIDS 2017, MQTTset.

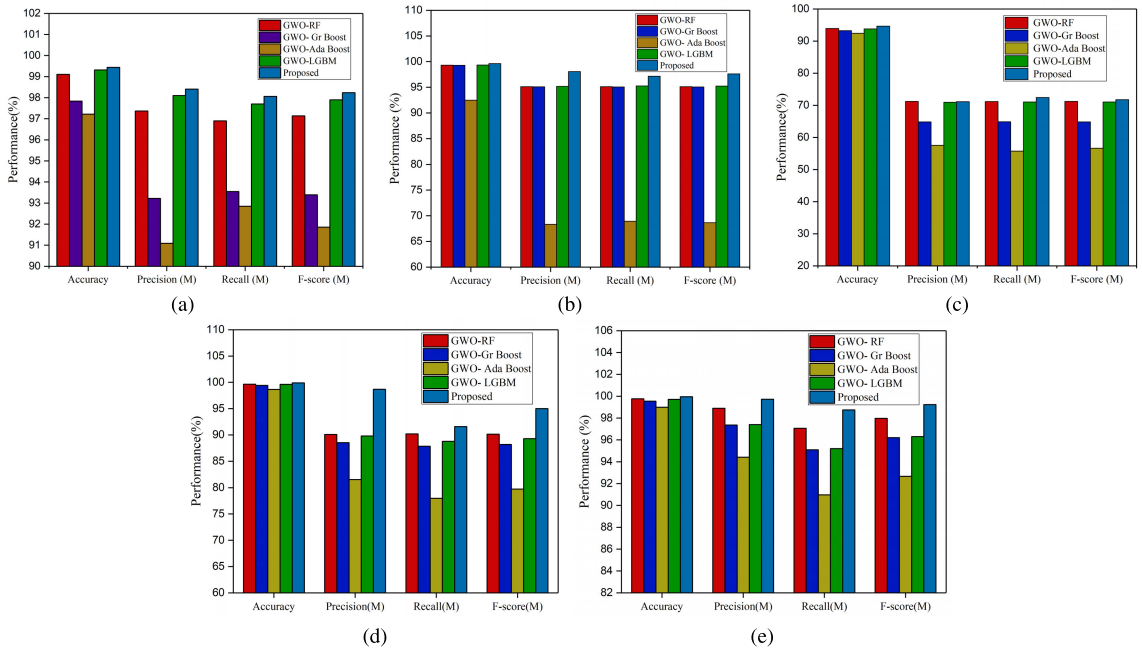

FIGURE 18. Performance comparison of proposed method (a) Cooja (b) NSL KDD (c) UNSW-NB15 (d) CICIDS 2017 (e) MQTTset.

TABLE 14. Performance of the UNSW NB 15 dataset with optimized classifier.

Classifier	IGWO-RF	IGWO-GR Boost	IGWO-Ada Boost	IGWO-Light GBM	Proposed SoE
Average accuracy	93.92	93.24	92.45	93.79	94.64
Error rate	6.08	6.76	7.55	6.21	5.36
Precision (Macro)	71.21	64.84	57.54	70.96	71.11
Recall (Macro)	71.18	64.86	55.75	71.05	72.41
F-score (Macro)	71.19	64.85	56.64	71.01	71.76
Precision (micro)	69.58	66.22	62.27	68.97	73.18
Recall (micro)	69.58	66.22	62.27	68.97	73.18
F-score (micro)	69.58	66.22	62.27	68.97	73.18

TABLE 15. Performance of the CICIDS 2017 dataset with optimized classifier.

Classifier	IGWO-RF	IGWO-GR Boost	IGWO-Ada Boost	IGWO-Light GBM	Proposed SoE
Average accuracy	99.65	99.43	98.66	99.62	99.90
Error rate	0.35	0.57	1.34	0.38	0.10
Precision (Macro)	90.09	88.52	81.54	89.79	98.69
Recall (Macro)	90.20	87.86	77.99	88.78	91.59
F-score (Macro)	90.14	88.19	79.72	89.29	95.01
Precision (micro)	98.76	98.01	95.32	98.68	99.64
Recall (micro)	98.76	98.01	95.32	98.68	99.64
F-score (micro)	98.76	98.01	95.32	98.68	99.64

(SoE) classifier. From the table it is evident that the proposed SoE has the highest accuracy of 99.44% among the other classifiers. Tables 13 and 14 show the performance of the proposed SoE for NSL KDD and UNSW NB 15 datasets.

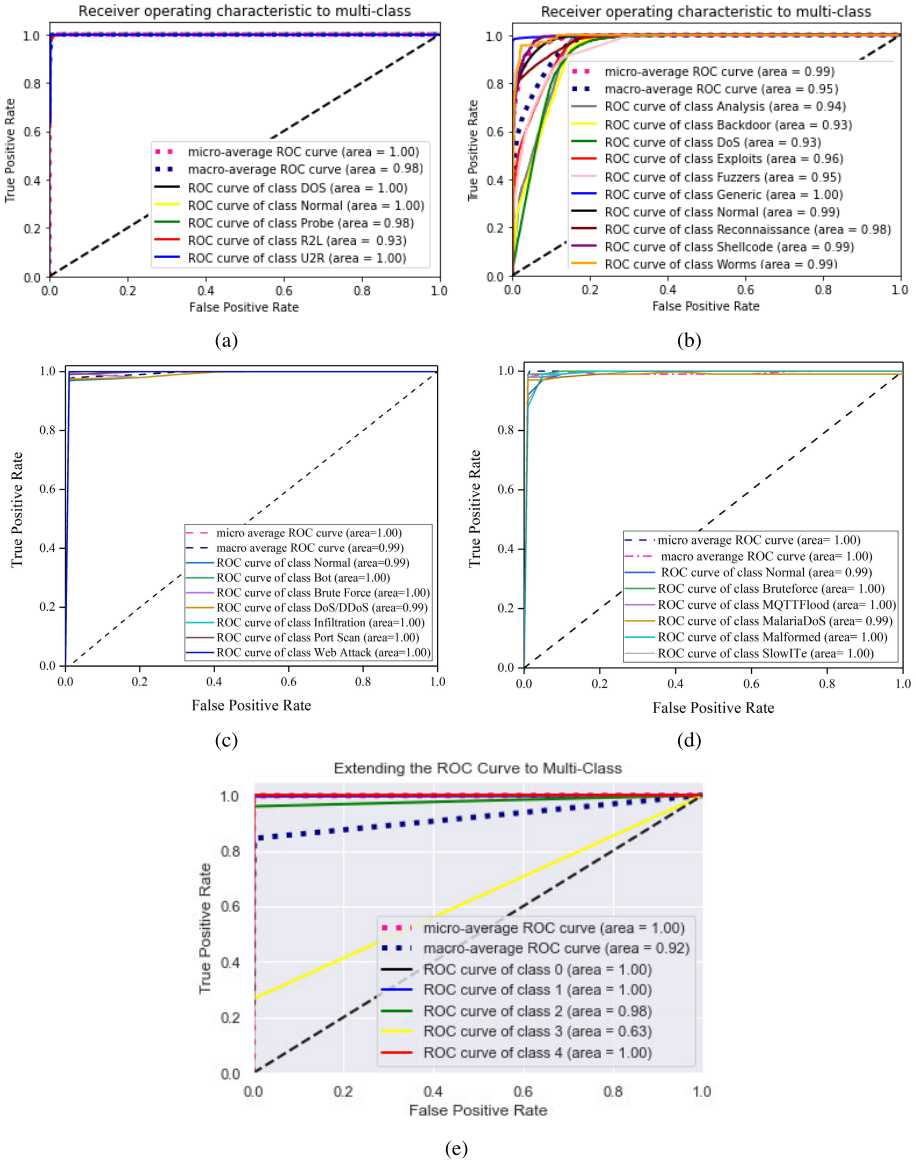

FIGURE 19. ROC curve for the different attacks present in the dataset (a) NSL KDD (b) UNSW-NB15 (d) CICIDS 2017 (e) MQTTset (c) Simulated dataset.

TABLE 16. Performance of the MQTTSet dataset with optimized classifier.

Classifier	IGWO-RF	IGWO-GR Boost	IGWO-Ada Boost	IGWO-Light GBM	Proposed SoE
Average accuracy	99.76	99.55	98.99	99.71	99.95
Error rate	0.24	0.45	1.01	0.29	0.05
Precision (Macro)	98.90	97.36	94.42	97.41	99.73
Recall (Macro)	97.07	95.10	90.97	95.20	98.75
F-score (Macro)	97.97	96.21	92.67	96.30	99.24
Precision (micro)	99.27	98.65	96.96	99.12	99.84
Recall (micro)	99.27	98.65	96.96	99.12	99.84
F-score (micro)	99.27	98.65	96.96	99.12	99.84

Here also the proposed SoE has the highest detection accuracy of 99.60% for NSL KDD and 94.64% for UNSW NB 15 dataset. Further, for the CICIDS 2017 dataset and the MQTTSET dataset the detection accuracy of proposed SoE is excellent and it is explained in Tables 15 and 16.

The performance of the ensemble learning algorithms is greatly influenced by the parameters such as number of trees, learning rate, and sample rate. When the number of trees in a model grows, it is more probable that the model may overfit the training data, which can result in inaccurate predictions and incapability for different kinds of new data. Here, our proposed IGWO algorithm is used to find the optimum tree size to avoid over-fitting and increase the detection accuracy. IGWO also optimizes the ensemble algorithms' learning rate, thus increasing the training speed and reducing the error function. Further, the sample rate of the ensemble algorithms is also tuned by the IGWO algorithm to increase the stability of the model, and the class imbalance problem of poorer predictive accuracy over the minority attacks is resolved.

There is a significant effect of hyper-parameter optimization on different datasets. The Cooja dataset has the performance values of 98.61%, 93.36%, 93.20%, and 98.71% in the default settings of the RF, Gr Boost, Ada Boost, and LGBM classifiers, at the same time, the performance reached 99.11%, 97.84%, 97.22%, and 99.32%, respectively, after the IGWO optimization process. The RF, Gr Boost, Ada Boost, and LGBM classifiers initial settings for the NSL KDD dataset yielded performance values of 98.93%, 98.81%, 86.72%, and 98.84%, respectively; however, after optimization, the performances increased to 99.30%, 99.27%, 92.48%, and 99.34%. After the IGWO optimization, the performance of the RF, GR Boost, Ada Boost, and LGBM classifiers improved form 93.41%, 92.64%, 91.93%, and 93.29% to 93.92%, 93.24%, 92.45%, and 93.79% for UNSW NB 15 dataset. Similarly, the performance of the RF, GR Boost, Ada Boost, and LGBM classifiers also exhibits a notable improvement after IGWO optimization for the CICIDS 2017 and MQTTset datasets.

Figure. 18 compares the proposed model's performance for Cooja, NSL KDD, UNSW NB 15, CICIDS 2017, and MQTTset dataset. According to the figure, the proposed SoE algorithm performs better in all parameters than other algorithms. Figure. 19 shows the multi-classification ROC curve of Cooja, NSL KDD, UNSW NB 15, CICIDS 2017, and MQTTset dataset. The ROC Curve illustrates how different probability thresholds impact TPR and FPR. Here, the AUC value of the different classes are denoted individually. The micro-average value of the ROC curve shows that our model has excellent performance. Thus the proposed classifier outperforms the base classifier and the existing anomaly detection models based on all the performance metrics.

b: STATISTICAL SIGNIFICANCE TESTS

The two variant objectives of the non-parametric chi-square test are i) to inquire about the autonomy of the two variables (ii) to determine the resemblance between the practical distribution of data and the expected distribution of data. Chi-square statistic is computed using Equation (35). Where O indicates observed value and E indicates expected value [58].

$$\chi^2 = \sum_{i=1}^{n} \frac{(O_i - E_i)^2}{E_i} \tag{35}$$

Chi-square analysis is done to determine how closely the experimental samples match the predicted samples. Our null hypothesis can be widely accepted if the chi-squared score is low. If the value is discovered to be high, our null hypothesis can be refuted and it will become clear that a significant cause is at play. The proposed SoE algorithm with the bi-classification (Normal, Anomaly) problem and the chi-square test are explained in this part. The null hypothesis (H0) denotes no coalition between observed and expected values. The alternative hypothesis (H1) indicates the coalition between observed and expected values.

TABLE 17. Observed values for computing chi-square statistics.

	Predict		
Actual	Anomaly	Normal	Total
Anomaly	46998	202	47200
Normal	299	41961	42260
Total	47297	42163	89460

TABLE 18. Expected values for computing chi-square statistics.

	Predict		
Actual	Anomaly	Normal	Total
Anomaly	24954.38	22245.62	47200
Normal	22342.62	19917.38	42260
Total	47297	42163	89460

Tables 17 and 18 show the actual and predicted value to compute the Chi-square test for the developed model.

The expected value (E) for a particular cell can be determined using the given formula

$$E = \frac{Row\ total\ anomalies \times Column\ total\ anomalies}{Total\ no.of\ \ anomalies} \tag{36}$$

The degrees of freedom (DOF) between one sample and another when the comparison is made is given by (number of columns-1) × (number of rows-1). In this anomaly detection problem, DOF = (2-1) × (2-1) = 1. The Chi-square value can be computed from Equation (35), where the value is found to be 87461.34. $chi^2contingency()$ function in a Python is used to find χ^2 value and DOF, here the DOF value equals 1. The ideal value is found to be 0.0000 using the Chi-square value. For 5% level of significance, we need an alpha level of 0.05. From the Chi-square distribution table the row with one DOF and column with 0.05 significance level gives the critical χ^2 (3.841) value and it is lesser than our proposed χ^2 (87461.34) value. The null hypothesis is disproved, or in another way, the alternate hypothesis is approved because a P value of 0.00 is less than the generally accepted significance level of 0.05, at which point it is inferred that the system is highly significant and have strong correlation between actual and estimated value.

3) SETUP III

a: PERFORMANCE COMPARISON OF EXISTING PARAMETER TUNING METHODS

Our proposed stacked ensemble learning algorithms are optimized using existing hyper-parameter tuning algorithms

TABLE 19. Performance of the Cooja dataset with existing optimizer.

Classifier	PSO-SoE	GWO-SoE	IGWO-SoE
Average accuracy	99.40	99.31	99.44
Error rate	0.60	0.69	0.56
Precision (Macro)	98.24	97.94	98.41
Recall (Macro)	98.02	97.75	98.06
F-score (Macro)	98.13	97.85	98.24
Precision (micro)	98.49	98.27	98.59
Recall (micro)	98.49	98.27	98.59
F-score (micro)	98.49	98.27	98.59

TABLE 20. Performance of the NSL KDD dataset with existing optimizer.

Classifier	PSO-SoE	GWO-SoE	IGWO-SoE
Average accuracy	99.39	99.52	99.60
Error rate	0.61	0.48	0.40
Precision (Macro)	95.90	95.10	98.07
Recall (Macro)	95.50	97.16	97.15
F-score (Macro)	95.70	96.12	97.61
Precision (micro)	98.47	98.81	99.01
Recall (micro)	98.47	98.81	99.01
F-score (micro)	98.47	98.81	99.01

TABLE 21. Performance of the UNSW NB 15 dataset with existing optimizer.

Classifier	PSO-SoE	GWO-SoE	IGWO-SoE
Average accuracy	94.11	94.39	94.64
Error rate	5.89	5.61	5.36
Precision (Macro)	69.46	70.69	71.16
Recall (Macro)	71.79	72.14	72.47
F-score (Macro)	70.61	71.41	71.81
Precision (micro)	70.53	71.97	73.38
Recall (micro)	70.53	71.97	73.38
F-score (micro)	70.53	71.97	73.38

TABLE 22. Performance of the CICIDS 2017 dataset with existing optimizer.

Classifier	PSO-SoE	GWO-SoE	IGWO-SoE
Average accuracy	99.76	99.79	99.90
Error rate	0.24	0.21	0.10
Precision (Macro)	90.26	90.29	98.69
Recall (Macro)	88.87	88.99	91.59
F-score (Macro)	89.56	89.64	95.01
Precision (micro)	99.15	99.27	99.64
Recall (micro)	99.15	99.27	99.64
F-score (micro)	99.15	99.27	99.64

TABLE 23. Performance of the MQTTset dataset with existing optimizer.

Classifier	PSO-SoE	GWO-SoE	IGWO-SoE
Average accuracy	99.85	99.88	99.95
Error rate	0.15	0.12	0.05
Precision (Macro)	99.58	99.61	99.73
Recall (Macro)	98.65	98.69	98.75
F-score (Macro)	99.11	99.15	99.24
Precision (micro)	99.55	99.65	99.84
Recall (micro)	99.55	99.65	99.84
F-score (micro)	99.55	99.65	99.84

like the PSO and GWO algorithm, and their results are compared and tabulated in Tables 19 - 23. The table shows that our proposed method has superior accuracy compared to the existing algorithms.

We also verified the developed method using the benchmark dataset and evaluated them by comparing them to the other existing models, which are tabulated in Table 24. It demonstrates how the suggested model is very competitive as an efficient method for the task of anomaly identification.

TABLE 24. A comparison of the proposed IGWO-SoE model with existing methodologies.

Ref	Methodology	Dataset	Accuracy	Statistical Test
[20]	RF, CART	NSL KDD, UNSW NB 15	94.44, 96.74	Yes
[29]	SoE	NSL KDD, UNSW NB 15	92.17, 92.45	Yes
[30]	GXG Boost	N-BaIoT	99.96	Yes
[31]	B-Stacking	NSL KDD	98.50	No
[33]	PSO-LightGBM	UNSW-NB15	86.68	No
[35]	TSE-IDS	NSL KDD, UNSW NB 15	85.80, 91.27	Yes
[36]	GA-LGBM	DS2OS	99.99	No
[37]	MOB-EBATMLP	CICIDS 2017	99.23	No
[38]	WOA-XG Boost	KDD CUP 99	99.06	No
[40]	FA-XG Boost	UNSW NB 15	94.64	No
[53]	Dual-IDS	NSL KDD, UNSW NB 15	91.57, 94.60	No
[59]	MGWO-SVM	NSL KDD	87.59	No
[60]	GTO-BSA	NSL KDD	95.59	No
[61]	LGBM	UNSW NB 15	88.34	No
[62]	LGBM	UNSW NB 15	85.89	No
	Proposed	NSL KDD, UNSW NB 15, CICIDS 2017, MQTTset	99.60, 94.66, 99.90, 99.95	Yes

Along with performance analysis, statistical significance tests show that the suggested classifier performs significantly better than contemporary methods. Furthermore, no statistical tests have been included in any of the numerous methods presented thus far in the literature.

V. CONCLUSION

This study proposes an effective anomaly detection model using an optimized stack of ensemble learning algorithms, i.e., RF, Gr Boosting, Ada Boosting, and Light GBM for IoT edge scenarios. We have demonstrated that the concepts of stacking combined with the IGWO-optimized ensemble learning algorithms can effectively detect the different types of anomalies. A novel synthetic dataset was created using the Cooja simulator to train and test our proposed model, whereas the existing studies use standard benchmark datasets. The robustness of the developed model is also expressed by four different datasets: NSL KDD, UNSW NB 15, CICIDS 2017, and MQTTset. The empirical results reveal that the proposed techniques produce an accuracy of 99.44%, 99.60%, 94.64%, 99.90%, and 99.95% for our Cooja, NSL-KDD, UNSW-NB 15, CICIDS 2017, and MQTTset datasets, respectively. Compared to the existing state-of-the-art method, our proposed algorithm performs better in terms of accuracy, precision, recall, ROC curve, and F-score value. The future perspectives of this study aim to detect the zero-day attack by including more attack datasets. Furthermore, evaluation can also be performed on real-time

IoT application scenarios by the stack of deep learning algorithms.

REFERENCES

[1] S. Kumar, P. Tiwari, and M. Zymbler, "Internet of Things is a revolutionary approach for future technology enhancement: A review," *J. Big Data*, vol. 6, no. 1, pp. 1–21, Dec. 2019, doi: 10.1186/s40537-019-0268-2.

[2] S. A. Kumar, N. Ahmed, and A. Bikos, "SWIoTA: Anomaly detection for distributed ledger technology-based Internet of Things (IOTA) using sliding window (SW) technique," in *Proc. IFIP Int. Internet Things Conf.*, 2022, pp. 177–194.

[3] H. F. Atlam and G. B. Wills, "Technical aspects of blockchain and IoT," in *Advances in Computers*. Amsterdam, The Netherlands: Elsevier, 2019, pp. 1–39, doi: 10.1016/bs.adcom.2018.10.006.

[4] M. Suwannakit, "Aurelia Tamò–Larrieux, designing for privacy and its legal framework: Data protection by design and default for the Internet of Things," *Int. Data Privacy Law*, vol. 9, no. 4, pp. 302–304, Jul. 2019, doi: 10.1093/idpl/ipz013.

[5] I. Statista. (2018). *Internet of Things (IoT) Connected Devices Installed Base Worldwide From 2015 to 2025 (in Billions)*. [Online]. Available: https://www.statista.com/statistics/471264/iot-number-of-connecteddevicesworldwide/

[6] S. A. Kumar, T. Vealey, and H. Srivastava, "Security in Internet of Things: Challenges, solutions and future directions," in *Proc. 49th Hawaii Int. Conf. Syst. Sci. (HICSS)*, Jan. 2016, pp. 5772–5781, doi: 10.1109/hicss.2016.714.

[7] Statista. (2022). *Size of the Internet of Things (IoT) Security Market Worldwide From 2016 to 2025*. [Online]. Available: https://www.statista.com/statistics/993789/worldwide-internet-of-things-security-market-size

[8] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): A vision, architectural elements, and future directions," *Future Gener. Comput. Syst.*, vol. 29, no. 7, pp. 1645–1660, Sep. 2013, doi: 10.1016/j.future.2013.01.010.

[9] C. Ioannou, V. Vassiliou, and C. Sergiou, "An intrusion detection system for wireless sensor networks," in *Proc. 24th Int. Conf. Telecommun. (ICT)*, May 2017, pp. 1–5, doi: 10.1109/ICT.2017.7998271.

[10] S. Rajasegarar, C. Leckie, and M. Palaniswami, "Anomaly detection in wireless sensor networks," *IEEE Wireless Commun.*, vol. 15, no. 4, pp. 34–40, Aug. 2008, doi: 10.1109/mwc.2008.4599219.

[11] M. Douiba, S. Benkirane, A. Guezzaz, and M. Azrour, "Anomaly detection model based on gradient boosting and decision tree for IoT environments security," *J. Reliable Intell. Environ.*, vol. 2022, pp. 1–12, Jul. 2022, doi: 10.1007/s40860-022-00184-3.

[12] G. T. Reddy, M. P. K. Reddy, K. Lakshmanna, R. Kaluri, D. S. Rajput, G. Srivastava, and T. Baker, "Analysis of dimensionality reduction techniques on big data," *IEEE Access*, vol. 8, pp. 54776–54788, 2020, doi: 10.1109/ACCESS.2020.2980942.

[13] P. A. A. Resende and A. C. Drummond, "A survey of random forest based methods for intrusion detection systems," *ACM Comput. Surv.*, vol. 51, no. 3, pp. 1–36, May 2018, doi: 10.1145/3178582.

[14] S. Mishra, R. Sagban, A. Yakoob, and N. Gandhi, "Swarm intelligence in anomaly detection systems: An overview," *Int. J. Comput. Appl.*, vol. 43, no. 2, pp. 109–118, Sep. 2018, doi: 10.1080/1206212x.2018.1521895.

[15] J. Manokaran and G. Vairavel, "Smart anomaly detection using data-driven techniques in IoT edge: A survey," in *Proc. 3rd Int. Conf. Commun., Comput. Electron. Syst.* Singapore: Springer, 2022, pp. 685–702.

[16] J. Manokaran and G. Vairavel, "An empirical comparison of machine learning algorithms for attack detection in Internet of Things edge," *ECS Trans.*, vol. 107, no. 1, pp. 2403–2417, Apr. 2022, doi: 10.1149/10701.2403ecst.

[17] N. Ashraf, W. Ahmad, and R. Ashraf, "A comparative study of data mining algorithms for high detection rate in intrusion detection system," *Ann. Emerg. Technol. Comput.*, vol. 2, no. 1, pp. 49–57, Jan. 2018, doi: 10.33166/aetic.2018.01.005.

[18] Q. Abu Al-Haija and M. Al-Dala'ien, "ELBA-IoT: An ensemble learning model for botnet attack detection in IoT networks," *J. Sensor Actuator Netw.*, vol. 11, no. 1, p. 18, Mar. 2022, doi: 10.3390/jsan11010018.

[19] I. Abrar, Z. Ayub, F. Masoodi, and A. M. Bamhdi, "A machine learning approach for intrusion detection system on NSL-KDD dataset," in *Proc. Int. Conf. Smart Electron. Commun. (ICOSEC)*, Sep. 2020, pp. 919–924, doi: 10.1109/icosec49089.2020.9215232.

[20] A. Verma and V. Ranga, "Machine learning based intrusion detection systems for IoT applications," *Wireless Pers. Commun.*, vol. 111, no. 4, pp. 2287–2310, Nov. 2019, doi: 10.1007/s11277-019-06986-8.

[21] M. Feurer and F. Hutter, "Hyperparameter optimization," in *Automated Machine Learning*. Cham, Switzerland: Springer, 2019, pp. 3–33.

[22] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Adv. Eng. Softw.*, vol. 69, pp. 46–61, Mar. 2014, doi: 10.1016/j.advengsoft.2013.12.007.

[23] (2014). *NSL KDD Dataset for Network-Based Intrusion Detection Systems*. [Online]. Available: http://nsl.cs.unb.ca/nsl-kdd/.html

[24] N. Moustafa and J. Slay, "UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," in *Proc. Mil. Commun. Inf. Syst. Conf. (MilCIS)*, Nov. 2015, pp. 1–6, doi: 10.1109/milcis.2015.7348942.

[25] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *Proc. ICISS*, 2018, pp. 108–116, doi: 10.5220/0006639801080116.

[26] I. Ullah and Q. H. Mahmoud, "Design and development of a deep learning-based model for anomaly detection in IoT networks," *IEEE Access*, vol. 9, pp. 103906–103926, 2021, doi: 10.1109/ACCESS.2021.3094024.

[27] B. A. Tama and S. Lim, "Ensemble learning for intrusion detection systems: A systematic mapping study and cross-benchmark evaluation," *Comput. Sci. Rev.*, vol. 39, Feb. 2021, Art. no. 100357, doi: 10.1016/j.cosrev.2020.100357.

[28] A. Thakkar and R. Lohiya, "A review on machine learning and deep learning perspectives of IDS for IoT: Recent updates, security issues, and challenges," *Arch. Comput. Methods Eng.*, vol. 28, no. 4, pp. 3211–3243, Oct. 2020, doi: 10.1007/s11831-020-09496-0.

[29] B. A. Tama, L. Nkenyereye, S. M. R. Islam, and K.-S. Kwak, "An enhanced anomaly detection in web traffic using a stack of classifier ensemble," *IEEE Access*, vol. 8, pp. 24120–24134, 2020, doi: 10.1109/ACCESS.2020.2969428.

[30] M. Alqahtani, H. Mathkour, and M. M. Ben Ismail, "IoT botnet attack detection based on optimized extreme gradient boosting and feature selection," *Sensors*, vol. 20, no. 21, p. 6336, Nov. 2020, doi: 10.3390/s20216336.

[31] S. Roy, J. Li, B.-J. Choi, and Y. Bai, "A lightweight supervised intrusion detection mechanism for IoT networks," *Future Gener. Comput. Syst.*, vol. 127, pp. 276–285, Feb. 2022, doi: 10.1016/j.future.2021.09.027.

[32] Y. Maleh, A. Sahid, and M. Belaissaoui, "Optimized machine learning techniques for IoT 6LoWPAN cyber attacks detection," in *Proc. 12th Int. Conf. Soft Comput. Pattern Recognit.* Cham, Switzerland: Springer, 2021, pp. 669–677.

[33] J. Liu, D. Yang, M. Lian, and M. Li, "Research on intrusion detection based on particle swarm optimization in IoT," *IEEE Access*, vol. 9, pp. 38254–38268, 2021, doi: 10.1109/ACCESS.2021.3063671.

[34] H. Jiang, Z. He, G. Ye, and H. Zhang, "Network intrusion detection based on PSO-XGBoost model," *IEEE Access*, vol. 8, pp. 58392–58401, 2020, doi: 10.1109/ACCESS.2020.2982418.

[35] B. A. Tama, M. Comuzzi, and K.-H. Rhee, "TSE-IDS: A two-stage classifier ensemble for intelligent anomaly-based intrusion detection system," *IEEE Access*, vol. 7, pp. 94497–94507, 2019, doi: 10.1109/ACCESS.2019.2928048.

[36] D. Mishra, B. Naik, J. Nayak, A. Souri, P. B. Dash, and S. Vimal, "Light gradient boosting machine with optimized hyperparameters for identification of malicious access in IoT network," *Digit. Commun. Netw.*, vol. 9, no. 1, pp. 125–137, Feb. 2023, doi: 10.1016/j.dcan.2022.10.004.

[37] W. A. H. M. Ghanem, S. A. A. Ghaleb, A. Jantan, A. B. Nasser, S. A. M. Saleh, A. Ngah, A. C. Alhadi, H. Arshad, A. H. Y. Saad, A. E. Omolara, Y. A. B. El-Ebiary, and O. I. Abiodun, "Cyber intrusion detection system based on a multiobjective binary bat algorithm for feature selection and enhanced bat algorithm for parameter optimization in neural networks," *IEEE Access*, vol. 10, pp. 76318–76339, 2022, doi: 10.1109/ACCESS.2022.3192472.

[38] M. Zivkovic, M. Tair, N. Bacanin, Š. Hubálovský, and P. Trojovský, "Novel hybrid firefly algorithm: An application to enhance XGBoost tuning for intrusion detection classification," *PeerJ Comput. Sci.*, vol. 8, p. e956, Apr. 2022, doi: 10.7717/peerj-cs.956.

[39] Y. N. Kunang, S. Nurmaini, D. Stiawan, and B. Y. Suprapto, "Improving classification attacks in IoT intrusion detection system using Bayesian hyperparameter optimization," in *Proc. 3rd Int. Seminar Res. Inf. Technol. Intell. Syst. (ISRITI)*, Dec. 2020, pp. 146–151.

[40] Y. Song, H. Li, P. Xu, and D. Liu, "A method of intrusion detection based on WOA-XGBoost algorithm," *Discrete Dyn. Nature Soc.*, vol. 2022, pp. 1–9, Feb. 2022, doi: 10.1155/2022/5245622.
[41] A. Verma and V. Ranga, "ELNIDS: Ensemble learning based network intrusion detection system for RPL based Internet of Things," in *Proc. 4th Int. Conf. Internet Things, Smart Innov. Usages (IoT-SIU)*, Apr. 2019, pp. 1–6.
[42] I. A. Khan, N. Moustafa, D. Pi, K. M. Sallam, A. Y. Zomaya, and B. Li, "A new explainable deep learning framework for cyber threat discovery in industrial IoT networks," *IEEE Internet Things J.*, vol. 9, no. 13, pp. 11604–11613, Jul. 2022, doi: 10.1109/JIOT.2021.3130156.
[43] I. A. Khan, N. Moustafa, I. Razzak, M. Tanveer, D. Pi, Y. Pan, and B. S. Ali, "XSRU-IoMT: Explainable simple recurrent units for threat detection in Internet of Medical Things networks," *Future Gener. Comput. Syst.*, vol. 127, pp. 181–193, Feb. 2022, doi: 10.1016/j.future.2021.09.010.
[44] I. A. Khan, D. Pi, Z. U. Khan, Y. Hussain, and A. Nawaz, "HML-IDS: A hybrid-multilevel anomaly prediction approach for intrusion detection in SCADA systems," *IEEE Access*, vol. 7, pp. 89507–89521, 2019, doi: 10.1109/ACCESS.2019.2925838.
[45] I. A. Khan, D. Pi, M. Z. Abbas, U. Zia, Y. Hussain, and H. Soliman, "Federated-SRUs: A federated-simple-recurrent-units-based IDS for accurate detection of cyber attacks against IoT-augmented industrial control systems," *IEEE Internet Things J.*, vol. 10, no. 10, pp. 8467–8476, May 2023, doi: 10.1109/JIOT.2022.3200048.
[46] I. A. Khan, N. Moustafa, D. Pi, W. Haider, B. Li, and A. Jolfaei, "An enhanced multi-stage deep learning framework for detecting malicious activities from autonomous vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 12, pp. 25469–25478, Dec. 2022, doi: 10.1109/TITS.2021.3105834.
[47] I. Essop, J. C. Ribeiro, M. Papaioannou, G. Zachos, G. Mantas, and J. Rodriguez, "Generating datasets for anomaly-based intrusion detection systems in IoT and industrial IoT networks," *Sensors*, vol. 21, no. 4, p. 1528, Feb. 2021, doi: 10.3390/s21041528.
[48] J. Manokaran and G. Vairavel, "GIWRF-SMOTE: Gini impurity-based weighted random forest with SMOTE for effective malware attack and anomaly detection in IoT-edge," *Smart Sci.*, vol. 11, no. 2, pp. 276–292, Dec. 2022, doi: 10.1080/23080477.2022.2152933.
[49] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," *J. Artif. Intell. Res.*, vol. 16, pp. 321–357, Jun. 2002, doi: 10.1613/jair.953.
[50] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001.
[51] B. A. Tama and K.-H. Rhee, "An in-depth experimental study of anomaly detection using gradient boosted machine," *Neural Comput. Appl.*, vol. 31, no. 4, pp. 955–965, Jul. 2017, doi: 10.1007/s00521-017-3128-z.
[52] W. Hu, W. Hu, and S. Maybank, "AdaBoost-based algorithm for network intrusion detection," *IEEE Trans. Syst., Man, Cybern., B, Cybern.*, vol. 38, no. 2, pp. 577–583, Apr. 2008, doi: 10.1109/TSMCB.2007.914695.
[53] M. H. L. Louk and B. A. Tama, "Dual-IDS: A bagging-based gradient boosting decision tree model for network anomaly intrusion detection system," *Expert Syst. Appl.*, vol. 213, Mar. 2023, Art. no. 119030, doi: 10.1016/j.eswa.2022.119030.
[54] R. Mohakud and R. Dash, "Designing a grey wolf optimization based hyper-parameter optimized convolutional neural network classifier for skin cancer detection," *J. King Saud Univ.-Comput. Inf. Sci.*, vol. 34, no. 8, pp. 6280–6291, Sep. 2022, doi: 10.1016/j.jksuci.2021.05.012.
[55] X. Yu, W. Xu, and C. Li, "Opposition-based learning grey wolf optimizer for global optimization," *Knowl.-Based Syst.*, vol. 226, Aug. 2021, Art. no. 107139, doi: 10.1016/j.knosys.2021.107139.
[56] H. Rajadurai and U. D. Gandhi, "A stacked ensemble learning model for intrusion detection in wireless network," *Neural Comput. Appl.*, vol. 34, no. 18, pp. 15387–15395, May 2020, doi: 10.1007/s00521-020-04986-5.
[57] R. Rajamohamed and J. Manokaran, "Improved credit card churn prediction based on rough clustering and supervised learning techniques," *Cluster Comput.*, vol. 21, no. 1, pp. 65–77, Jun. 2017, doi: 10.1007/s10586-017-0933-1.
[58] E. Sivasankar and J. Vijaya, "Hybrid PPFCM-ANN model: An efficient system for customer churn prediction through probabilistic possibilistic fuzzy clustering and artificial neural network," *Neural Comput. Appl.*, vol. 31, no. 11, pp. 7181–7200, May 2018, doi: 10.1007/s00521-018-3548-4.
[59] T. A. Alamiedy, M. Anbar, Z. N. M. Alqattan, and Q. M. Alzubi, "Anomaly-based intrusion detection system using multi-objective grey wolf optimisation algorithm," *J. Ambient Intell. Humanized Comput.*, vol. 11, no. 9, pp. 3735–3756, Nov. 2019, doi: 10.1007/s12652-019-01569-8.
[60] S. S. Kareem, R. R. Mostafa, F. A. Hashim, and H. M. El-Bakry, "An effective feature selection model using hybrid metaheuristic algorithms for IoT intrusion detection," *Sensors*, vol. 22, no. 4, p. 1396, Feb. 2022, doi: 10.3390/s22041396.
[61] Z. Wang, J. Liu, and L. Sun, "EFS-DNN: An ensemble feature selection-based deep learning approach to network intrusion detection system," *Secur. Commun. Netw.*, vol. 2022, pp. 1–14, Apr. 2022, doi: 10.1155/2022/2693948.
[62] J. Liu, Y. Gao, and F. Hu, "A fast network intrusion detection system using adaptive synthetic oversampling and LightGBM," *Comput. Secur.*, vol. 106, Jul. 2021, Art. no. 102289, doi: 10.1016/j.cose.2021.102289.

J. MANOKARAN received the bachelor's degree in electronics and communication engineering and the master's degree in applied electronics from Anna University, in 2009 and 2011, respectively. He is currently pursuing the Ph.D. degree with the SRM Institute of Science and Technology, Kattankulathur, Chennai, India. His list of academic achievements includes five publications, two journals, and three conference papers. His area of interests include the Internet of Things, computational intelligence, machine learning, deep learning, edge computing, cluster computing, and intrusion detection. During his post-graduation, he received the Best Technical Paper Award in the national level conference on "Computer Intelligent and Application" organized by the CARE College of Engineering, Tiruchirappalli.

G. VAIRAVEL (Senior Member, IEEE) received the bachelor's degree in electronics and communication engineering from Madras University, India, in 2001, and the master's degree in communication systems and the Ph.D. degree in wireless communication from Anna University, Chennai, India, in 2004 and 2014, respectively. He is currently a Professor with the Directorate of Learning and Development, SRM Institute of Science and Technology, Kattankulathur, Chennai. His research interests are massive MIMO communication systems, antenna design, machine learning, the Internet of Things, and engineering education. He is an Active Member of the American Society for Engineering Education and the TPACK Special Interest Group and a fellow of the Institute of Engineers, India. Currently, he is elected as a Member-at-Large with the CDIO International Council. He is serving as an editor and a guest editor for reputed journals and chaired reputed conferences.