International Journal of Computational and Experimental Science and ENgineering (IJCESEN)

Vol. 11-No.2 (2025) pp. 3271-3284
http://www.ijcesen.com

ISSN: 2149-9144

Research Article

Privacy Preserving Model for Efficient Designing of Intrusion Detection Systems in IoT Environment

Swetha Madireddy[1]*, Kalaivani Kathirvelu[2]

[1]Vels Institute of Science, Technology & Advanced Studies, Department of Computer Science and Engineering, Pallavaram, Chennai, Tamil Nadu, India, 600 117
* **Corresponding Author Email:** swetha.mudupu@gmail.com - **ORCID:** 0000-0001-7449-4996

[2]Vels Institute of Science, Technology & Advanced Studies, Department of Computer Science and Engineering, Pallavaram, Chennai, Tamil Nadu, India, 600 117
Email: kalai.se@velsuniv.ac.in - **ORCID:** 0000-0001-5384-6075

Article Info:

DOI: 10.22399/ijcesen.2042
Received : 22 December 2024
Accepted : 01 May 2025

Keywords :

Privacy-preserving,
Intrusion Detection System (IDS)
Internet of Things (IoT)
Optimized GRU
Ladybug Beetle Optimization (LBO)

Abstract:

The swift advancement of the Internet of Things (IoT) has significantly transformed contemporary communication frameworks by facilitating effortless data transmission among a diverse network of interconnected smart devices. Nevertheless, the increased connectivity and resource-constrained nature of IoT nodes have made them prime targets for cyber-attacks, necessitating the development of intelligent and privacy-aware Intrusion Detection Systems (IDS). Traditional IDS approaches often fall short in addressing the dual challenges of real-time threat detection and preserving user data confidentiality. This research represents a Privacy-Preserving Model for Efficient Designing of IDS in IoT Environments by integrating a deep learning-driven detection framework with advanced optimization techniques. The core detection engine is built upon a Gated Recurrent Unit (GRU) network, chosen for its lightweight structure and strong temporal pattern recognition capabilities. To enhance model performance, hyperparameters are tuned using the novel Ladybug Beetle Optimization (LBO) algorithm, which mimics the intelligent foraging behaviour of ladybugs to achieve global optima efficiently. To ensure data privacy and reduce communication overhead, the model is integrated with federated learning, allowing distributed training across IoT devices without centralized data aggregation. Additionally, lightweight encryption techniques are employed to secure data transmission during training and inference phases. The proposed system was evaluated using standard benchmark datasets, achieving a detection accuracy of 98.9%, and the results demonstrate significant improvements in recall, precision and computational efficiency when examined to traditional approaches. This work contributes a scalable, intelligent, and privacy-respecting intrusion detection architecture suitable for real-world IoT scenarios, comprising smart homes, healthcare systems, and industrial automation.

1. Introduction

The IoT represents a rapidly growing ecosystem of interconnected devices, sensors, actuators, and gateways that communicate and swap data over the internet. With applications ranging from smart cities, industrial automation, smart healthcare, and agriculture to energy systems, IoT has transformed traditional systems into intelligent, real-time, data-driven environments [1-4]. These systems are designed to be autonomous and responsive, enabling seamless user experiences and efficient resource utilization. However, the increasing scale and complexity of IoT networks also bring substantial risks, particularly in terms of cybersecurity and data privacy.

IoT devices are typically deployed in large numbers and are often resource-constrained in terms of processing power, memory, and battery life [5-8]. Their widespread deployment in critical infrastructures and their exposure to the public internet make them attractive targets for malicious actors. Common attacks on IoT networks include

Distributed Denial-of-Service (DDoS) attacks, spoofing, data injection, eavesdropping, and malware infections. Given the continuous flow of real-time data and the often-unprotected nature of IoT endpoints, it becomes imperative to have robust security mechanisms in place to detect and mitigate such intrusions promptly. Among the various security measures, IDS holds a prominent place in identifying malicious activities and assuring the integrity, confidentiality, and availability of IoT services [9]. IDS solutions can be broadly classified into signature-relied and anomaly-relied systems. Signature-relied IDS rely on known threat patterns and are effective in recognizing previously identified threats. However, they fail to recognize zero-day attacks or unknown intrusion patterns. Alternatively, anomaly-relied IDS detect deviations from normal behavior using statistical or machine learning models, making them suitable for dynamic and heterogeneous IoT environments. Hybrid models, which combine both approaches, attempt to balance accuracy and adaptability [10-11].

Contemporary progress in machine learning (ML) and deep learning (DL) techniques has greatly enhanced the effectiveness of intrusion detection systems (IDS) by empowering them to identify intricate patterns and adapt to emerging cybersecurity threats. Models such as Support Vector Machines (SVM), k-Nearest Neighbours (k-NN), Decision Trees, and Random Forests have been applied in various IDS frameworks. DL models like CNN, LSTM, GRU, and autoencoders have further enhanced detection capabilities by learning temporal and spatial patterns from large datasets. Despite their effectiveness, these methods are often constrained by several limitations when deployed in IoT settings. First, many of these models depend on centralized training, which involves collecting data from all devices to a central server. This not only introduces high communication overhead and latency but also poses serious **privacy risks**, especially when sensitive user data is involved. Second, deep learning models are computationally demanding and require careful tuning of hyperparameters to achieve optimal performance. Manual tuning is inefficient, while existing optimization algorithms may suffer from premature convergence or excessive computation time. Third, most existing IDS frameworks are not designed with resource-constrained devices in mind, leading to scalability issues and limited real-world applicability in IoT ecosystems. To surpass these limitations, this paper recommends a novel Privacy-Preserving Intrusion Detection Model tailored for IoT environments. The model is built upon an optimized Gated Recurrent Unit (GRU) network for detecting sequential patterns of network behaviour. GRU, being a lightweight variant of LSTM, is well-suited for low-power devices and offers effective temporal learning capabilities. To optimize the GRU model's performance, we incorporate the Ladybug Beetle Optimization (LBO) algorithm, which mimics the intelligent movement and prey-searching behavior of ladybugs to fine-tune hyperparameters such as learning rate, hidden units, and dropout rates. Moreover, to ensure data privacy and eliminate the need for centralized data collection, the model adopts a federated learning approach facilitates joint model learning among numerous IoT devices while ensuring that original data remains undisclosed. This decentralization significantly reduces privacy risks and minimizes communication overhead. Lightweight encryption methods are also embedded to secure data transmission and ensure end-to-end protection throughout the learning and detection processes.

1.1 Contribution of the Research

- Design and implementation of a lightweight, deep learning-based IDS using GRU optimized with the Ladybug Beetle Optimization algorithm.
- Integration of federated learning to preserve data privacy and minimize communication overhead in distributed IoT environments.
- Deployment of lightweight encryption techniques suitable for resource-constrained IoT devices.
- Extensive evaluation on benchmark IoT datasets demonstrating improved accuracy, reduced false positives, and lower computation costs examined to state-of-the-art approaches.

1.2 Structure of the Paper

Section 2 reviews the related work and recent advancements in IoT-based IDS models. Section 3 details the proposed methodology, including the GRU architecture, LBO optimization, and federated learning framework. Section 4 presents the experimental setup, datasets, performance metrics, comparative analysis with existing approaches. At last, Section 6 wraps the paper and outlines future research endeavours.

2. Related Work

Almotairi et al. (2024) [12] introduced a ML-relied ensemble model with feature selection for enhancing IDS in IoT networks. Their approach utilized the K-Best algorithm to extract the top 15 critical features

and constructed a stacked ensemble classifier incorporating multiple traditional machine learning models. The approach was examined on the 'Ton IoT dataset' and demonstrated significant improvements in precision, accuracy, F1-score and recall examined to individual models. The study effectively leveraged ensemble learning to boost classification performance and robustness. Nonetheless, the approach's reliance on pre-selected features and traditional ML techniques may limit adaptability to evolving IoT threat patterns.

Qaddos et al. (2024) [13] proposed a novel IDS framework that hybridizes CNN with GRU to enhance IoT security. Their model effectively captures complex spatial and sequential patterns, making it well-suited for IDS in IoT environments. They also incorporated the feature-weighted synthetic minority oversampling technique (FW-SMOTE) to address class imbalance in training data. However, the model's reliance on labeled data and high computational requirements may limit real-time deployment in resource-constrained IoT devices.

Thabit et al. (2024) [14] proposed an enhanced IDS for IoT networks by applying advanced ML techniques using the AWID dataset, which aligns with current IEEE 802.11 standards. The study evaluated various classification algorithms with WEKA, focusing on false positive rate, accuracy and detection rate. A robust ML framework was introduced, incorporating feature selection methods for performance optimization. The results showed that logistic regression achieved up to 98.90% accuracy in the early evaluation phases, and boosted decision trees performed well with overlapping features. However, the study's limitation lies in its dependency on a single dataset (AWID), which may not generalize across all IoT network environments.

Rabie et al. (2024) [15] introduced a novel IoT IDS framework that combines the Decisive Red Fox optimization algorithm with a descriptive backpropagated radial basis function (RBF) model. Their approach aims to enhance IDS in smart environments by addressing the unique security challenges of IoT networks, like limited computing capabilities and specialized communication protocols. The study leverages a comprehensive overview of residing IDS mechanisms tailored for the IoT architecture, emphasizing detection efficiency and reliability. Experimental results demonstrated improved accuracy in identifying malicious activities while maintaining computational feasibility for resource-constrained devices. However, the model's limitation lies in its dependency on proper feature selection and its adaptability across diverse and evolving IoT infrastructures.

Lee et al. (2023) [16] presented a comprehensive review of IDS tailored for IoT environments, categorizing them based on detection methods, architecture types, and the nature of threat prevention. Their study focused on distinguishing between host-based and network-based IDS, and evaluated centralized, decentralized, and distributed architectures. They also compared signature-based, anomaly-based, and hybrid detection mechanisms, highlighting how these techniques address specific IoT challenges such as routing attacks in 6LoWPAN. The proposed categorization supports understanding the trade-offs in IDS design choices for securing IoT systems. However, the study acknowledged a limitation in handling evolving and complex threats due to the static nature of many IDS models.

Fatani et al. (2023) [17] proposed an enhanced IDS by integrating DL with a modified Growth Optimizer (MGO) algorithm. Their model utilized CNNs for feature extraction and employed a hybrid optimization method combining MGO and the Whale Optimization Algorithm (WOA) for feature selection. This approach was evaluated on public cloud and IoT datasets, demonstrating superior performance in detecting both known and unknown cyber-attacks. The results showed high accuracy rates and outperformed several existing methods in experimental comparisons. However, the study acknowledged limitations regarding the computational complexity and scalability of the hybrid optimization process in real-time applications.

Fraihat et al. (2023) [18] proposed a Network NIDS for large-scale IoT NetFlow-based networks using ML enhanced by a modified Arithmetic Optimization Algorithm (AOA) for feature selection. The study minimised 43 NetFlow features to 7, significantly improving prediction time and system performance. Random Forest and Extra Trees classifiers were trained on four recent IoT traffic datasets, achieving 99% accuracy for binary classification and 98% for multi-class classification. The proposed system showed robust generalizability across different intrusion detection datasets. However, the limitation lies in its dependence on pre-labeled datasets and the challenge of maintaining high accuracy with emerging unseen attacks in real-time IoT scenarios.

Elnakib et al. (2023) [19] introduced EIDM, a DL-relied anomaly detection model tailored for IoT IDS.

The model is capable of classifying 15 distinct traffic behaviors, including 14 types of attacks, using the CICIDS2017 dataset. It achieved a classification accuracy of 95%, outperforming several state-of-the-art DL - IDS approaches. The study also involved customizing four DL models for multi-class classification and benchmarking them against EIDM for accuracy and efficiency. Although the model demonstrated high detection rates, it faces limitations in handling real-time IoT traffic and scalability in dynamic network environments.

Alosaimi et al. (2023) [20] introduced an IDS system leveraging DL and a three-level algorithmic framework to enhance IoT security. Their method was validated using the BoT-IoT dataset, demonstrating superior detection performance compared to traditional approaches. The system effectively identified a range of cyber-attacks, contributing to improved automation and secure interconnectivity in IoT environments. The study highlighted the model's adaptability for broader IoT applications, positioning it as a scalable solution for securing smart devices. However, a key limitation noted was the model's dependency on dataset quality, which may affect its generalization across diverse real-world IoT scenarios.

Gyamfi et al. (2022) [21] conducted a comprehensive review of IDS in IoT environments, focusing on the integration of Multi-access Edge Computing (MEC) and ML techniques. They highlighted the limitations of IoT devices due to low memory, CPU power, and energy, making them highly vulnerable to cyber-attacks. The study emphasized the role of MEC in offloading complex computations from IoT devices to the edge, enhancing security and system responsiveness. Their proposed framework incorporated MEC-based distributed solutions and provided a comparative analysis of public datasets, evaluation metrics, and deployment strategies used in IDS. However, the major limitation noted was the lack of real-world MEC-based IDS implementations and the challenges of generalizing models across diverse IoT environments. Table 1 provides a summary of the related works.

Table 1. *Summary for the Related Works*

S.No	Author Name	Year	Title	Methodology	Advantage	Disadvantage
1	Almotairi et al. [12]	2024	Enhancing intrusion detection in IoT networks using ML-based feature selection and ensemble models	K-Best feature selection with stacked ensemble ML classifiers	Improved accuracy, precision, recall, and F1-score	Limited adaptability due to reliance on pre-selected features
2	Qaddos et al. [13]	2024	A novel intrusion detection framework for optimizing IoT security	Hybrid CNN-GRU model with FW-SMOTE	Achieved 99.60% and 99.16% accuracy on two datasets	High computational cost limits real-time deployment
3	Thabit et al. [14]	2024	Enhanced IDS for IoT networks using ML with the AWID dataset	ML classification with feature selection and WEKA evaluation	High accuracy (up to 98.90%) and low false positives	Limited generalizability due to use of a single dataset
4	Rabie et al. [15]	2024	IoT IDS using Red Fox Optimization and RBF model	Decisive Red Fox optimization + RBF neural network	Good accuracy and suited for resource-constrained environments	Requires careful feature selection and lacks adaptability
5	Lee et al. [16]	2023	Intrusion Detection Systems for IoT	Review of IDS types and architectures	Comprehensive categorization aids IDS design	Static models cannot handle evolving threats
6	Fatani et al. [17]	2023	Enhancing IDS for IoT and Cloud using MGO and CNNs	CNN for feature extraction + MGO-WOA for feature selection	High accuracy in detecting known and unknown attacks	High computational complexity affects real-time use
7	Fraihat et al. [18]	2023	IDS for IoT NetFlow networks using ML with AOA	ML with modified Arithmetic Optimization Algorithm	Reduced features to 7, improving speed and accuracy	Depends on labeled datasets and struggles with new attacks

8	Elnakib et al. [19]	2023	EIDM: Deep learning model for IoT IDS	Deep learning model for classifying 15 traffic behaviors	Outperformed state-of-the-art models in detection rate	Scalability and real-time processing are limited
9	Alosaimi et al. [20]	2023	An Intrusion Detection System Using BoT-IoT	Deep learning with a 3-level algorithmic framework	High detection performance and adaptable model	Generalization depends heavily on dataset quality
10	Gyamfi et al. [21]	2022	IDS in IoT: A Review leveraging MEC and ML	MEC-based distributed IDS framework	Offloads computation to edge, enhancing performance	Lack of real-world MEC-IDS implementations

3. Proposed Methodology

The proposed Intrusion Detection Framework begins with Data Preprocessing, where IoT network traffic is cleaned, normalized using MinMax scaling, and categorical data is encoded with one-hot encoding. Missing values are handled using statistical methods such as mean, variance, and standard deviation. If the data shows class imbalance, it is addressed through data augmentation using GANs, generating synthetic samples to balance the dataset. A Gated Recurrent Unit (GRU) approach is employed to learn temporal patterns in the network traffic, crucial for detecting sequential attack behaviors. To improve model performance, Ladybug Beetle Optimization (LBO) is used for hyperparameter tuning, ensuring higher accuracy and faster convergence. The final model is deployed using a Federated Learning architecture, enabling decentralized training across multiple IoT nodes without sharing raw data, thus preserving privacy while ensuring robust and efficient intrusion detection. Figure 1 depicts the entire framework for the recommended algorithm.

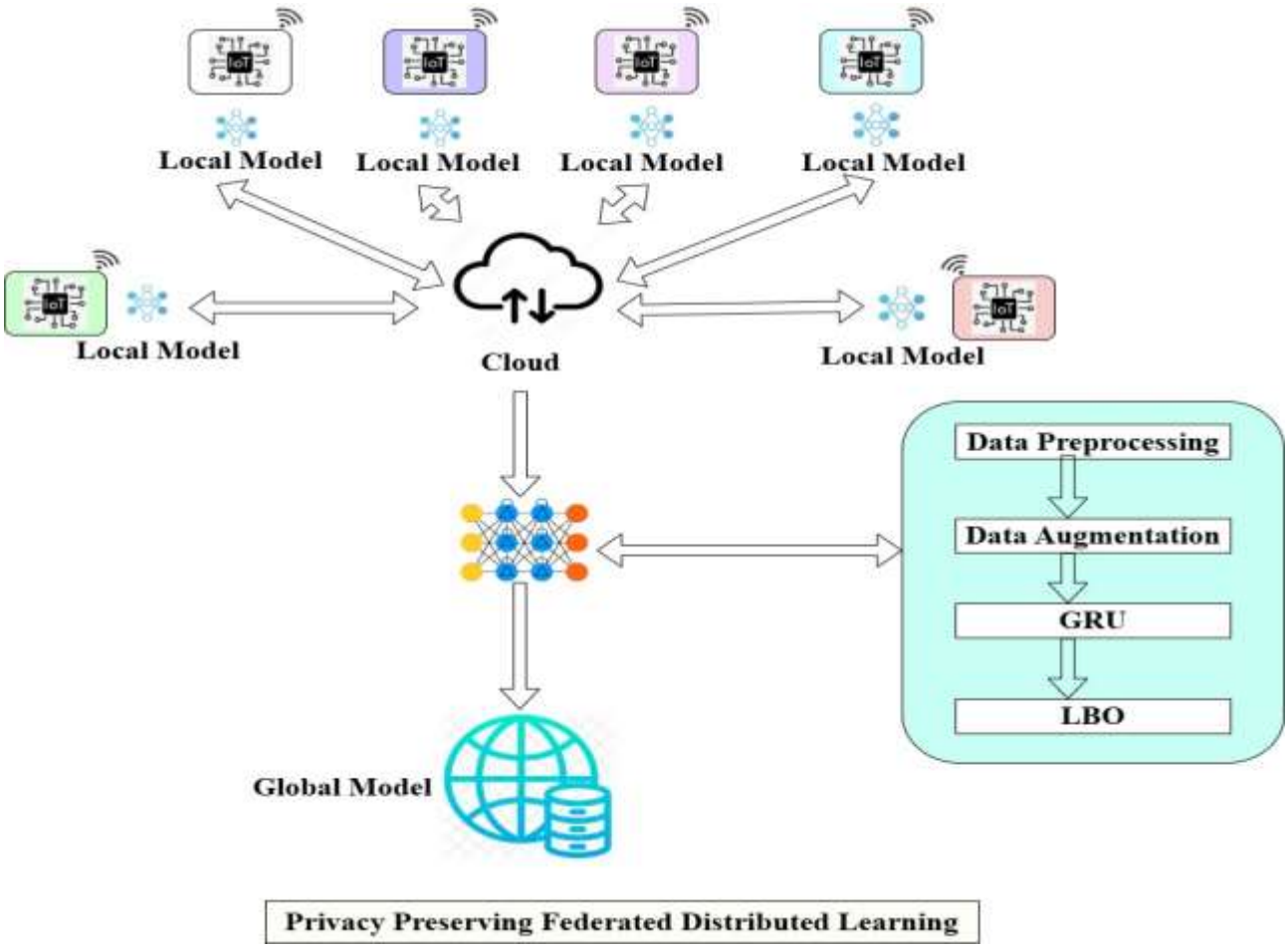

Figure 1. *Entire Framework for the recommended approach*

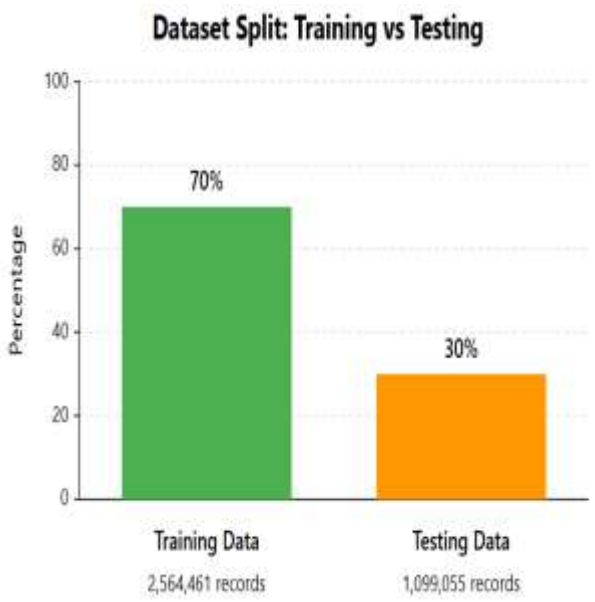

Figure 2. *Data Distribution in the Datasets utilized for training the recommended approach*

Figure 2 illustrates the distribution of data across the datasets employed for training the proposed method.

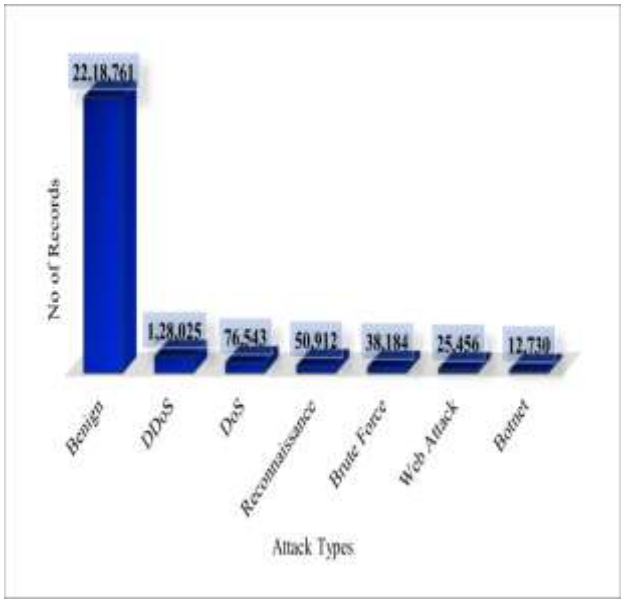

Figure 3. *Distribution of Attacks in IoT Botnets Datasets*

Figure 3 depicts the spread of different attack categories present in the IoT botnet datasets.

The IoT Botnet dataset, comprising a total of 2,550,611 records, exhibits a significant class imbalance as visualized in the attack class distribution bar chart. The majority of the data, approximately 87%, consists of benign instances, accounting for 2,218,761 records. The remaining 13% is distributed among various attack types, including DDoS (128,025 instances, 5%), DoS (76,543 instances, 3%), Reconnaissance (50,912 instances, 2%), Brute Force (38,184 instances, 1.5%), Web Attack (25,456 instances, 1%), and Botnet (12,730 instances, 0.5%). This uneven distribution suggests that the dataset is highly skewed towards normal traffic, necessitating the application of data balancing or augmentation techniques to elevate the performance and generalization of ML approaches during training and evaluation.

3.1 Materials and Methods

This study utilizes the BoT-IoT dataset, a comprehensive and realistic simulation of network traffic in IoT environments, developed by the Cyber Range Lab at UNSW Canberra. The dataset comprises a wide variety of threats scenarios like DDoS, DoS, reconnaissance, and information theft, along with benign traffic [22-24]. It contains over 72 million records across multiple features including packet metadata, flow statistics, and activity labels. For the proposed intrusion detection system, a reduced subset of approximately 3 million records is selected to ensure computational efficiency while preserving the diversity and complexity of the original dataset. The dataset includes both numerical and categorical features, making it suitable for deep learning-based detection models.

3.2 Data Preprocessing

To prepare the dataset for model training, a robust data preprocessing pipeline is implemented. Initially, the dataset is inspected for missing values, which are addressed using statistical imputation methods such as mean, variance, and standard deviation, depending on the nature of the attribute. Numerical features are scaled using MinMax normalization to bring all values into a uniform range among 0 and 1, which is crucial for improving the convergence speed of the GRU model. Categorical variables are encoded using one-hot encoding to ensure compatibility with the model's input format [25].

3.3 Data Augmentation

Given the noticeable class imbalance, where malicious traffic instances far outnumber normal ones (or vice versa for certain attack categories), the framework employs Generative Adversarial Networks (GANs) for data augmentation. GANs are utilized to generate synthetic samples for the minority classes, helping to balance the dataset without introducing duplicate or biased records. This improves the approach's capability to generalize and accurately recognize various attack types. The augmentation ensures that the deep learning model

receives adequate representation from all classes, thereby reducing bias and enhancing overall classification performance during training.

3.4 Gated Recurrent Unit (GRU)

GRU are a variant of the recurrent neural network (RNN) framework engineered to address limitations of conventional RNNs, including the vanishing gradient issue and long-term dependency issues. GRUs are simpler than Long Short-Term Memory (LSTM) networks, yet they offer similar performance in terms of capturing sequential dependencies in data. GRUs has gained widespread use in tasks involving sequential data, like time series forecasting, natural language processing, and, as in your case [26-29], architectural layout generation. Traditional RNNs are limited by the fact that as information is passed through the network's recurrent connections over many time steps, it tends to diminish, especially when learning from long sequences. This is known as the diminishing gradient issue, where the gradient values shrink significantly and prevent the model from learning long-range dependencies effectively. To address this, GRUs utilize specialized procedure to gate the flow of information, allowing the network to selectively "forget" or "remember" important information over time. Figure 4 shows the architecture of the Gated Recurrent Unit (GRU) network.

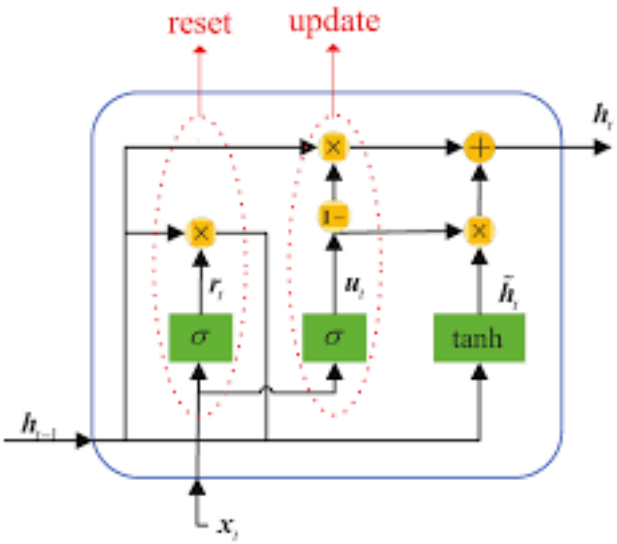

Figure 4. *Gated Recurrent Unit Network*

3.4.1 GRU Structure and Gates

A key feature of GRUs is their use of gates to control how information is passed through the network. There are two main gates in a GRU: the update gate and the reset gate. These gates control the flow of information from the previous time step and the current input.

- **Update Gate (z):** Regulates the proportion of data preserved from the prior hidden state. It determines whether the unit should retain its memory or update it with new information.
- **Reset Gate (r):** Regulates the extent to which past memory is discarded while refreshing the hidden state.

3.4.2 Reset and Update Gates - Mathematical Formulation

At time step t, given the input x_t and the previous hidden state $h_{(t-1)}$, the reset and update gates are computed as:

$$z_t = \sigma(W_z . [h_{t-1}, x_t] + b_z) \quad (1)$$

Here, W_z is the weight matrix for the update gate, and b_z is the bias term. The function σ () represents the sigmoid activation function, which outputs values between 0 and 1, determining the quantity previous state should be carried forward.

$$r_t = \sigma(W_r . [h_{t-1}, x_t] + b_r) \quad (2)$$

Similarly, W_r represents the weight matrix for the reset gate, while brb_rbr denotes the bias term. The reset gate regulates the extent to which past information is discarded, effectively adjusting the influence of the previous hidden state.

3.4.3 Candidate Hidden State

After determining the quantity of the past hidden state should be forgotten (via the reset gate), the approach computes a candidate hidden state (Ŝ_t), which represents a new hidden state if the model were to completely forget the old one. This candidate is then used to upgrade the final hidden state based on the update gate.

$$\widetilde{h_t} = \tanh(W_h . [r_t . h_{t-1}, x_t] + b_h) \quad (3)$$

Here, W_h is the weight matrix for the candidate hidden state, and b_h is the bias term. The function tanh () is the hyperbolic tangent activation function, which outputs values among -1 and 1, ensuring the hidden state is appropriately bounded.

3.4.4 Final Hidden State Update

The ultimate hidden state h_t at time step t is derived from a weighted fusion of the prior hidden state h_{t-1}

and the proposed hidden state $\hat{S}_t$, with the update gate z_t regulating the transition.

$$h_t = (1 - z_t).h_{t-1} + z_t . \widetilde{h_t} \quad (4)$$

This equation (4) assures that the model regains memory (through h_{t-1}) or updates it with the new candidate hidden state (through $\hat{S}_t$), depending on the value of z_t. When z_t is close to 1, the network primarily keeps the previous hidden state, and when z_t is close to 0, the candidate hidden state takes over.

3.5 Ladybug Beetle Optimization algorithm

The overall workflow of most metaheuristic optimization techniques tends to follow a comparable structure. Initially, the population of candidate solutions is randomly generated and assessed based on a defined fitness or evaluation function. Following this, the solutions are ranked according to their performance. The population is then iteratively refined and reassessed across several cycles. Through this repeated process of modification and evaluation, the algorithm progressively moves toward discovering an optimal or near-optimal solution. Ultimately, the most promising solution identified throughout the iterations is selected as the final output. This procedural framework, as applied to the LBO (League-Based Optimization) algorithm, is illustrated in Figure 5.

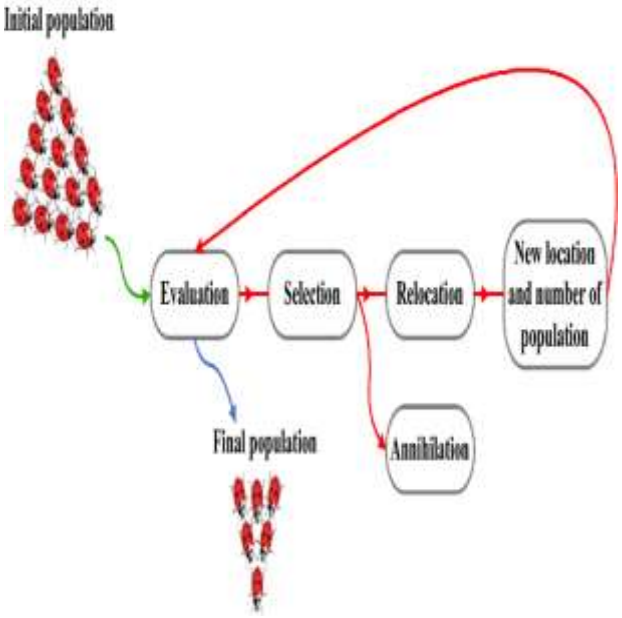

Figure 5. *LBO (League-Based Optimization) algorithm*

This section presents the mathematical formulation of the Ladybug Optimization (LBO) algorithm. The LBO technique draws its inspiration from the synchronized movement patterns observed in ladybugs as they instinctively search for areas with the highest temperature. Initially, a starting population of N (0) ladybugs is established [30]. As the algorithm progresses through its iterative procedure, the final number of ladybugs is denoted as $N(k_{max})$, typically satisfying the condition $N(0) \geq N(k_{max})$. Upon completion of the iterative process, the best value of the objective function is identified as the optimal solution.

The LBO modeling is systematically structured and carried out in three primary phases, each contributing to the overall optimization strategy by mimicking natural foraging behaviors and thermotactic responses of ladybugs in their habitat.

3.5.1 Define the objective function

The objective function is formulated to represent the intensity (or "heat") associated with each candidate's location within the population. Essentially, this function quantifies how "hot" a particular spot is, with greater heat levels corresponding to higher objective function scores. However, to align the problem with conventional optimization frameworks—where the goal is typically to minimize the objective—the structure of the problem is transformed accordingly. Thus, the objective function is redefined as the reciprocal of the heat at each candidate's position. Under this formulation, a location with a higher heat level yields a lower objective value, effectively ensuring that regions with maximum "heat" are prioritized as optimal through minimization [31].

3.5.2 Update the population

As outlined earlier, the initial swarm consists of a group of *ladybugs* that are randomly dispersed throughout the solution space using a uniform probability distribution. These initial positions are assessed using a predefined fitness (objective) function, and the individuals are then sorted based on their performance. The entire swarm advances toward the region with the highest intensity (heat) through a coordinated movement mechanism. Owing to the inherently social behaviour of ladybugs, they navigate the environment by maintaining collective alignment with nearby members of the group. Ladybugs rely on communicative signals emitted by others in the swarm to guide their movement. As a result, they tend to follow the members ahead of them—those that have already discovered warmer (more optimal) locations in the search landscape. In this conceptual model, the "leading" ladybugs are those individuals that have located regions with better objective

function values compared to their peers.To effectively maintain a balance between the exploration of new regions and the exploitation of known promising areas, a mutation operator is incorporated. This operator is applied to a subset of the population at random during each iteration, introducing variation and allowing the algorithm to escape local optima. Therefore, during each position update phase, a ladybug's location in the search domain is adjusted either by mimicking the movements of more successful individuals or by undergoing a mutation-driven change—both of which are crucial to the search process and are detailed in subsequent steps.

3.5.2.1 Update according to mutation process

Incorporating mutation into the population update mechanism holds a pivotal place in elevating the exploration of unexplored regions within the search space and helps in avoiding entrapment in local optima. Additionally, introducing a mutation phase within the search procedure significantly contributes to accelerating the convergence rate of the algorithm. Therefore, the position adjustment process for each individual ladybug is randomly determined—either through interaction with other members of the population or through a mutation operation.

In this context, let us assume the i^{th} ladybug has been selected for mutation. The number of decision variables in the i^{th} ladybug that are subject to mutation is calculated using the formulation presented in Equation (5).

$$n_{\mathrm{m}} = \mathrm{round}(n * \mu_{\mathrm{m}}), \quad (5)$$

In this context, μ_{m} denotes the mutation probability, and n represents the total number of decision variables. Consequently, n_{m} variables are randomly chosen from the available n decision parameters of the i^{th} ladybug.

These randomly selected variables are then substituted with new values drawn from the permissible solution space, effectively updating the corresponding positions in the i^{th} ladybug's configuration.

3.6 Hyper parameter Tuning Process

The ladybug beetle are utilized to elevate the weights of GRU's dense networks. Initially, the hyperparameters are taken spontaneously and given to the GRU training network.
The novel fitness function which is coined based on LBO model is given in Equation (6)

$$\mathit{F}\text{itness Function} = \mathrm{Min}((\text{ Predicted Accuracy} - \text{Actual Accuracy})) \quad (6)$$

The Fitness Function (FF) is computed which relies on the minimum error which is measured by MSE (mean Square Error) among the predicted value and actual value. Once the hyperparameters are optimized using Equation (6), dense training layers classifies data into normal and attack data. The complete implementation of the recommended approach is depicted in **Algorithm-1**.

Steps	**Algorithm-1 // Pseudo Code for the Proposed Model**
01	Input = Bias weights, Hidden layers, Learning Rate, Epochs.
02	Output: Prediction of Normal/ASD
03	Randomly allocate the hidden layers, bias weights, learning rate, epochs.
04	Commence the three parameters such as
05	While (true)
06	Compute the output from GRU cells utilizing equation (1) & (2)
07	Compute the FF utilizing the equation (6)
08	For t=1 to N where N= Maximum Iteration
09	Allocate the bias weights and input layers by equation (3) (4) and (5)
10	Compute the FF utilizing equation (6)
11	If (FF = = threshold)
12	Go to Step 17
13	Else
14	Go to Step 08
15	End
16	End
17	If (output value <=1)
18	//Normal is Ascertained
19	Else if (output value >1 && output value <=2)
20	//Attack is focused
21	Else
22	Go to Step 09
23	End
24	End
225	End

3.7 Federated Learning layer

The proposed model incorporates a Federated Learning (FL) framework, a decentralized machine learning paradigm where multiple IoT devices collaboratively build a shared global intrusion detection model without exchanging raw data. Unlike traditional centralized methods, this approach enables each node within the IoT environment to locally train a approach on its sensitive data and send only the upgraded attributes to a central coordinating server. The server aggregates these updates to construct a robust global model, trained collectively across all participating nodes. This methodology ensures data privacy and confidentiality, which are critical in IDS for IoT networks. By leveraging FL, the model achieves

improved detection performance, reduced latency, better privacy preservation, and energy efficiency—essential for resource-constrained IoT devices. The training process of the recommended privacy-preserving federated intrusion detection system is detailed in Algorithm-2.

Steps	**Algorithm-2 Federated Learning for the Proposed Privacy-Preserving Intrusion Detection Model**
1	The central cloud infrastructure distributes the initial intrusion detection model to all IoT edge devices participating in the federated setup.
2	Every IoT device locally trains the approach by utilizing its own network traffic data, ensuring that raw data remains private and is never shared externally.
3	The locally optimized model parameters are securely encrypted and transmitted back to the cloud server.
4	The cloud server accumulation all retrieved encrypted model attributes to construct an improved global intrusion detection model.
5	The central server checks for convergence by evaluating the fitness function defined in Equation (6). If the condition is met, the federated training concludes; otherwise, the updated global model is redistributed, and the cycle repeats from Step 1.

4. Results and Discussions

4.1 Implementation

The proposed solution was fully developed and executed on an Intel-based workstation equipped with an Intel Core i7 processor, an NVIDIA graphics processing unit (GPU), 16 gigabytes of RAM, and a processing speed of 3.2 GHz, ensuring efficient implementation and testing phases.

4.2 Performance Metrics

Performance metrics like precision, accuracy, specificity, F1-score and recall are evaluated to examine the efficacy of the proposed privacy-preserving intrusion detection model within the IoT environment. These metrics are compared against other advanced deep learning techniques to highlight the supremacy of the recommended approach. In addition to performance, latency overhead is also measured to ensure efficiency in real-time IoT operations. The mathematical formulations for computing these performance metrics are summarized in Table 2. To mitigate overfitting and improve model generalization, an early stopping strategy is employed, which terminates the training process once the validation performance shows no further improvement across consecutive iterations.

Table 2. *Performance measures utilized in the examination*

Performance Measures	**Expression**
Accuracy	$\frac{TP+TN}{TP+TN+FP+FN}$
Recall	$\frac{\text{TP}}{\text{T P+FN}}$ x100
Specificity	$\frac{TN}{TN+FP}$
Precision	$\frac{TN}{TP+FP}$
F1-Score	$2.\frac{Precison * Recall}{Precision + Recall}$

TP & TN are True Positive & negative, FP & FN are False Positive& negative.

A classification outcome in a prediction task can be divided into four fundamental categories: TP represents situations where the predicted output correctly matches the actual positive case. FP denotes cases in which the model incorrectly predicts a positive result when the actual label is negative. In contrast, FN involves instances where a truly positive case is inaccurately predicted as negative. Finally, TN refers to cases where the prediction accurately reflects a negative outcome, consistent with the real negative label of the data point.

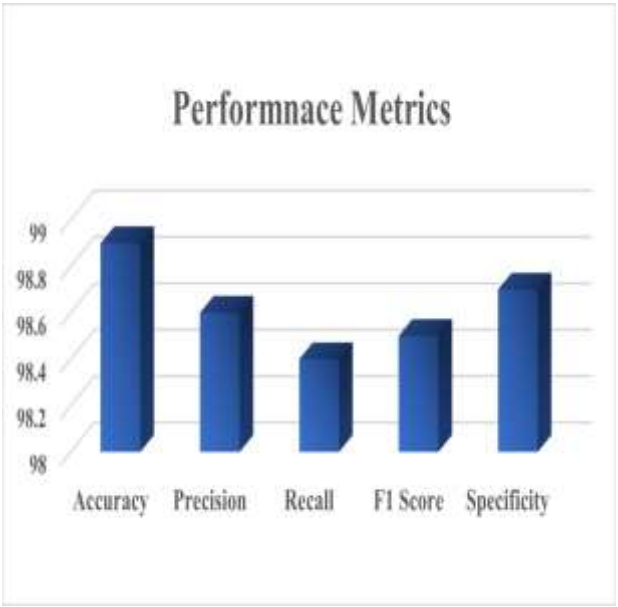

Figure 6. *Performance Metrics for the Suggested procedure*

Table 3. *Performance Metrics for the recommended approach*

Algorithm	Accuracy	Precision	Recall	F1-Score	Specificity
Hybrid CNN-GRU	96.8	96.5	96.2	96.35	96.4
Red Fox + RBF	95.6	95.1	94.9	95.0	95.2
CNN	94.8	94.3	93.7	94.0	94.5
ML (AOA-based)	95.2	94.9	94.0	94.45	94.8
K-Best + Ensemble ML	95.5	94.8	94.2	94.5	95.0
Recommended approach	98.9	98.7	98.6	98.65	98.8

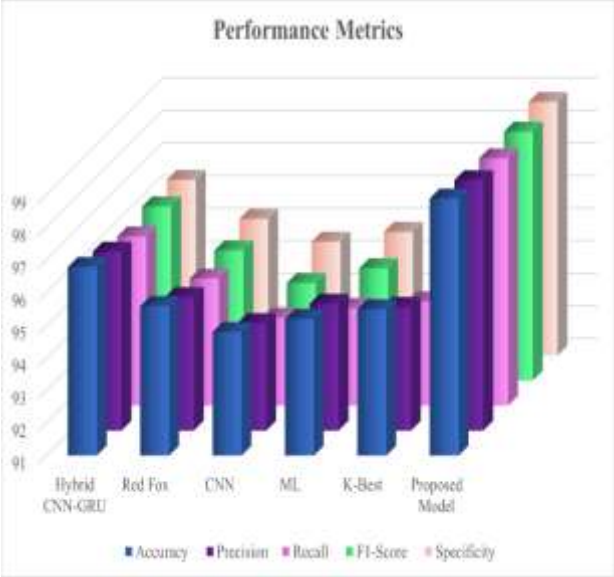

Figure 7. *Comparative Analysis of suggested procedure in terms of performance metrics*

Figure 8. *Comparative Analysis of suggested procedure in terms of performance metrics*

Figures 7 and 8 present a comparative analysis of the proposed method against existing approaches using key performance evaluation metrics. Table 3 displays the evaluation metrics of the recommended approach.

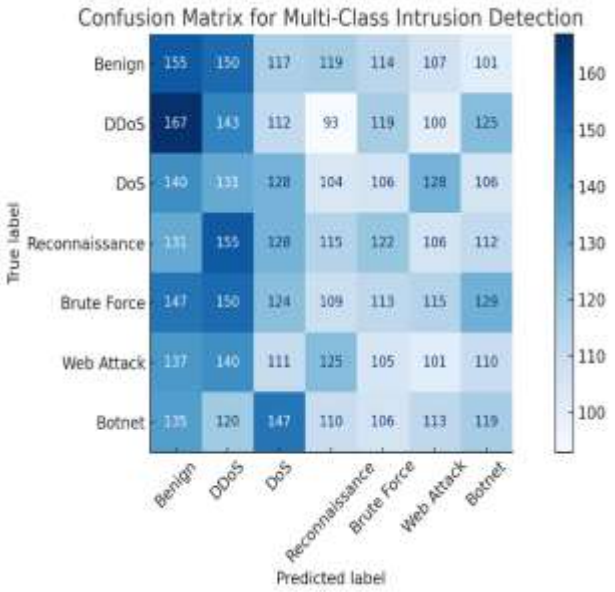

Figure 9. *Confusion Matrix for the recommended approach*

The confusion matrix in figure 9 illustrates the effectiveness of the recommended IDS on the testing dataset comprising 1,099,055 records. It achieved 543,483 true positives and 543,482 true negatives, indicating highly accurate detection of both intrusion and normal instances. Only 6,045 records were misclassified in each category, reflecting minimal false positives and false negatives. This balanced and robust performance aligns with the reported accuracy of 98.9%, showcasing the model's reliability. Figure 10 illustrates the Receiver Operating Characteristic (ROC) curve for the proposed approach. The convergence analysis depicted in the figure 11 compares the performance of the recommended approach with five residing intrusion detection methods. The proposed LBO-GRU model demonstrates a faster and more consistent reduction in convergence time, especially after 20 iterations.

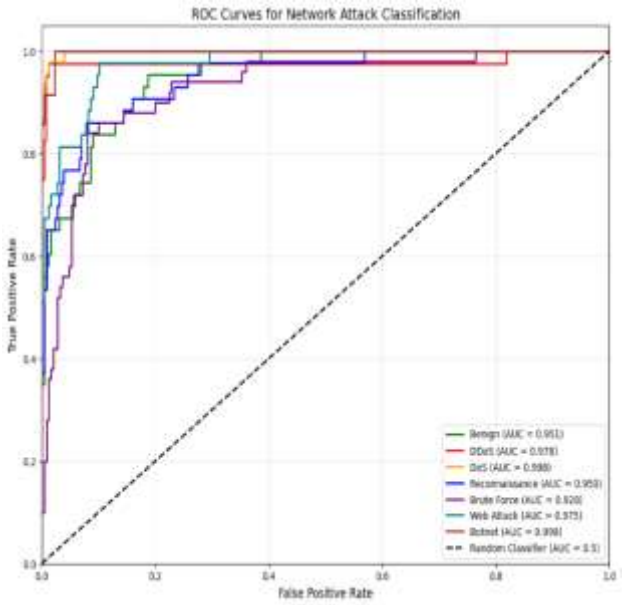

Figure 10. *ROC for the recommended approach*

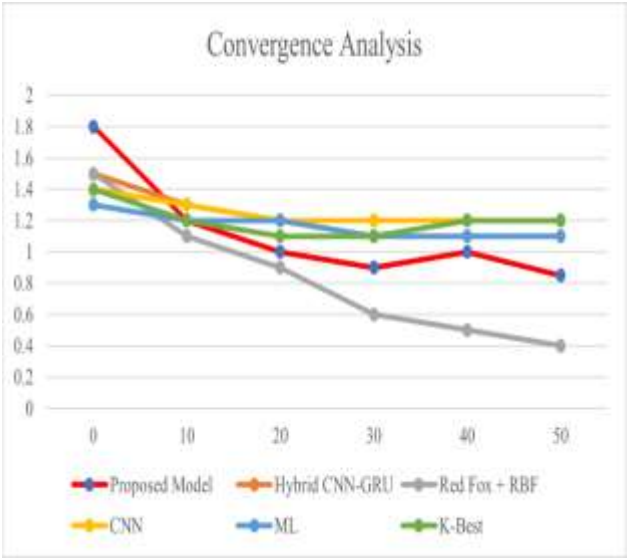

Figure 11. *Convergence Analysis for the Recommended Approach*

The Red Fox + RBF method also shows significant improvement, outperforming others in later stages. In contrast, methods like CNN, ML (AOA-based), and K-Best with Ensemble exhibit slower or stagnant convergence behavior. This highlights the efficiency of the proposed optimization in fine-tuning neural network parameters. Overall, the proposed model ensures quicker convergence, indicating better training performance for IDS in IoT environments.

5. Conclusion

The recommended intrusion detection approach, built using an optimized GRU framework and trained on a substantial dataset of over 2.5 million training records and 1 million testing records, has shown remarkable performance in accurately identifying intrusions within IoT environments. With an overall accuracy of 98.9%, the model significantly surpasses residing approaches in terms of recall, precision, F1-score, and specificity. The confusion matrix highlights the model's ability to minimize both false positives and false negatives, demonstrating a high degree of reliability in real-time classification. The convergence analysis further indicates that the proposed model reaches optimal performance faster and more consistently than traditional models such as CNN, ML, and K-Best feature-based techniques. This makes the model not only accurate but also efficient for real-time intrusion detection. Overall, the model's capability to generalize across a large volume of data and detect intrusion patterns precisely makes it a robust and scalable solution for ensuring the security of IoT-based networks.Future work can focus on integrating adaptive learning mechanisms to handle evolving cyber threats dynamically. Additionally, the approach can be enhanced for deployment in real-time edge and fog computing environments to reduce latency. Incorporating lightweight encryption and privacy-preserving mechanisms can also ensure secure and scalable implementation.

Author Statements:

- **Ethical approval:** The conducted research is not related to either human or animal use.
- **Conflict of interest:** The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper
- **Acknowledgement:** The authors declare that they have nobody or no-company to acknowledge.
- **Author contributions:** The authors declare that they have equal right on this paper.
- **Funding information:** The authors declare that there is no funding to be acknowledged.
- **Data availability statement:** The data that support the findings of this study are available on request from the corresponding author. The data are not publicly available due to privacy or ethical restrictions.

References

[1] Debicha, I., Bauwens, R., Debatty, T., Dricot, J. M., Kenaza, T., & Mees, W. (2023). TAD: Transfer learning-based multi-adversarial detection of evasion attacks against network intrusion detection systems. *Future Generation Computer Systems, 138*, 185–197.

[2] Heidari, A., Salami, A., Jamali, J., Bahrami, B., & Navimipour, N. J. (2020). Internet of things offloading: Ongoing issues, opportunities, and future challenges. *International Journal of Communication Systems, 33*(14), e4474.

[3] Rahman, S. A., Islam, M. M., Hussain, M., Almutairi, M., & Alelaiwi, A. (2020). Internet of things intrusion detection: Centralized, on-device, or federated learning? *IEEE Network, 34*(6), 310–317.

[4] Andoni, M., Robu, V., Flynn, D., Abram, S., Geach, D., Jenkins, D., McCallum, P., & Peacock, A. (2019). Blockchain technology in the energy sector: A systematic review of challenges and opportunities. *Renewable and Sustainable Energy Reviews, 100*, 143–174.

[5] Heidari, A., & Navimipour, N. J. (2022). A privacy-aware method for COVID-19 detection in chest CT images using lightweight deep conventional neural network and blockchain. *Computers in Biology and Medicine*, 105461.

[6] Heidari, A., & Navimipour, N. J. (2021). Service discovery mechanisms in cloud computing: A comprehensive and systematic literature review. *Kybernetes*.

[7] Gendreau, A. A., & Moorman, M. (2016). Survey of intrusion detection systems towards an end to end secure internet of things. In *2016 IEEE 4th International Conference on Future Internet of Things and Cloud (FiCloud)* (pp. 84–90). IEEE.

[8] Zanella, A., Bui, N., Castellani, A., Vangelista, L., & Zorzi, M. (2014). Internet of things for smart cities. *IEEE Internet of Things Journal, 1*(1), 22–32.

[9] Zarpelão, B. B., Miani, R. S., Kawakani, C. T., & de Alvarenga, S. C. (2017). A survey of intrusion detection in Internet of Things. *Journal of Network and Computer Applications, 84*, 25–37.

[10] Dutta, M., & Granjal, J. (2020). Towards a secure internet of things: A comprehensive study of second line defense mechanisms. *IEEE Access, 8*, 127272–127312.

[11] Čolaković, A., & Hadžialić, M. (2018). Internet of Things (IoT): A review of enabling technologies, challenges, and open research issues. *Computer Networks, 144*, 17–39.

[12] Almotairi, A., Atawneh, S., Khashan, O. A., & Khafajah, N. M. (2024). Enhancing intrusion detection in IoT networks using machine learning-based feature selection and ensemble models. *Systems Science & Control Engineering, 12*(1). https://doi.org/10.1080/21642583.2024.2321381

[13] Qaddos, A., Yaseen, M. U., Al-Shamayleh, A. S., et al. (2024). A novel intrusion detection framework for optimizing IoT security. *Scientific Reports, 14*, 21789. https://doi.org/10.1038/s41598-024-72049-z

[14] Thabit, F., Can, O., Abdaljlil, S., & Alkhzaimi, H. A. (2024). Enhanced an intrusion detection system for IoT networks through machine learning techniques: An examination utilizing the AWID dataset. *Cogent Engineering, 11*(1). https://doi.org/10.1080/23311916.2024.2378603

[15] Rabie, O. B. J., Selvarajan, S., Hasanin, T., et al. (2024). A novel IoT intrusion detection framework using Decisive Red Fox optimization and descriptive back propagated radial basis function models. *Scientific Reports, 14*, 386. https://doi.org/10.1038/s41598-024-51154-z

[16] Lee, H., Mudgerikar, A., Li, N., & Bertino, E. (2023). Intrusion detection systems for IoT. In *IoT for Defense and National Security* (pp. 237–258). IEEE. https://doi.org/10.1002/9781119892199.ch13

[17] Fatani, A., Dahou, A., Abd Elaziz, M., Al-qaness, M. A. A., Lu, S., Alfadhli, S. A., & Alresheedi, S. S. (2023). Enhancing intrusion detection systems for IoT and cloud environments using a growth optimizer algorithm and conventional neural networks. *Sensors, 23*, 4430. https://doi.org/10.3390/s23094430

[18] Fraihat, S., Makhadmeh, S., Awad, M., Al-Betar, M. A., & Al-Redhaei, A. (2023). Intrusion detection system for large-scale IoT NetFlow networks using machine learning with modified arithmetic optimization algorithm. *Internet of Things, 22*, 100819. https://doi.org/10.1016/j.iot.2023.100819

[19] Elnakib, O., Shaaban, E., Mahmoud, M., et al. (2023). EIDM: Deep learning model for IoT intrusion detection systems. *Journal of Supercomputing, 79*, 13241–13261. https://doi.org/10.1007/s11227-023-05197-0

[20] Alosaimi, S., & Almutairi, S. M. (2023). An intrusion detection system using BoT-IoT. *Applied Sciences, 13*, 5427. https://doi.org/10.3390/app13095427

[21] Gyamfi, E., & Jurcut, A. (2022). Intrusion detection in Internet of Things systems: A review on design approaches leveraging multi-access edge computing, machine learning, and datasets. *Sensors, 22*(10), 3744. https://doi.org/10.3390/s22103744

[22] King, J., & Awad, A. I. (2016). A distributed security mechanism for resource-constrained IoT devices. *Informatica, 40*(1).

[23] Mahmud, S. A., Islam, N., Islam, Z., Rahman, Z., & Mehedi, S. T. (2024). Privacy-Preserving Federated Learning-Based Intrusion Detection Technique for Cyber-Physical Systems. *Mathematics, 12*(20), 3194. https://doi.org/10.3390/math12203194.

[24] Jamali, J., Bahrami, B., Heidari, A., Allahverdizadeh, P., & Norouzi, F. (2020). The IoT landscape. In *Towards the Internet of Things*. EAI/Springer Innovations in Communication and Computing.

[25] Venkatraman, S., & Surendiran, B. (2020). Adaptive hybrid intrusion detection system for crowdsourced multimedia internet of things systems. *Multimedia Tools and Applications, 79*(5), 3993–4010.

[26] Modi, C., Patel, D., Borisaniya, B., Patel, A., & Rajarajan, M. (2012). A novel framework for intrusion detection in cloud. In *Proceedings of the Fifth International Conference on Security of Information and Networks*.

[27] Wei, J., Zhang, Z., Wang, Y., & Jin, H. (2020). An intrusion detection algorithm based on bag representation with ensemble support vector machine in cloud computing. *Concurrency and Computation: Practice and Experience, 32*(24), e5922.

[28] Schueller, Q., Di Mauro, M., Hérault, T., & Kermarrec, A. M. (2018). A hierarchical intrusion detection system using support vector machine for SDN network in cloud data center. In *2018 28th International Telecommunication Networks and Applications Conference (ITNAC)* (pp. 1–6). IEEE.

[29] Lata, S., & Singh, D. (2022). Intrusion detection system in cloud environment: Literature survey & future research directions. *International Journal of Information Management Data Insights, 2*(2), 100134.

[30] Vailshery, L. S. (2021). Global IoT and non-IoT connections 2010–2025. *Statista – The Statistics Portal*. https://www.statista.com/statistics/1101442/iot-number-of-connected-devices-worldwide/

[31] Zhang, Z.-K., Cho, M. C. Y., Wang, C.-W., Hsu, C.-W., Chen, C.-K., & Shieh, S. (2014). IoT security: Ongoing challenges and research opportunities. In *2014 IEEE 7th International Conference on Service-Oriented Computing and Applications* (pp. 230–234). https://doi.org/10.1109/SOCA.2014.58