Journal of Computer Science

Research Article

A Hybrid Grey Wolf Optimizer and Deep Transfer Learning-Based Intrusion Detection System for IoT

Kapil Shrivastava[1,2], Manish Tiwari[1] and Prasun Chakrabarti[1]

[1]*Faculty of Computing and Informatics, Sir Padampat Singhania University, Udaipur, India*
[2]*Department of CEA, GLA University, Mathura, India*

Article history
Received: 11-04-2025
Revised: 04-08-2025
Accepted: 26-08-2025

Corresponding Author:
Kapil Shrivastava
Faculty of Computing and Informatics, Sir Padampat Singhania University, Udaipur, India
Email: kapil1411@gmail.com

Abstract: The rapid proliferation of Internet of Things (IoT) devices has driven significant advancements in connectivity and automation, but it has also introduced substantial cybersecurity risks, including intrusions and cyberattacks. To address these challenges, this study proposes a hybrid intrusion detection framework that combines advanced optimization techniques with ensemble deep learning to enhance detection accuracy and computational efficiency. The proposed framework integrates Grey Wolf Optimizer (GWO) with Tabu Search for feature selection, effectively eliminating irrelevant and redundant features. These refined features are processed using an ensemble deep learning model comprising stacked Long Short-Term Memory (LSTM) networks and Gated Recurrent Units (GRU), which are adept at capturing complex temporal patterns in network traffic data. The framework was rigorously evaluated on benchmark datasets, including NSL-KDD, UNSW-NB15, and Edge-IIoT. The model achieved results, with accuracies of 99.55, 98.57 and 95.08%, respectively. Additionally, the system demonstrated a superior sensitivity of 99.21% and specificity of 99.45% on the NSL-KDD dataset while achieving high precision rates across all datasets. Comparative analyses showed that the framework consistently outperformed models such as GWO-GRU, GWO-LSTM, and GWOTB-LSTM in terms of accuracy, detection rate, and false alarm reduction. This robust and adaptable framework addresses the unique challenges of intrusion detection in IoT networks, supporting secure deployment in industrial and consumer IoT systems.

Keywords: Cyberattack, Deep Learning, Grey Wolf Optimizer, Intrusion Detection, Internet of Things

Introduction

The rapid evolution of IoT technology has led to its increasing utilization in recent years. IoT technology enables the connection and interaction of various objects through a network, thereby driving advancements in corporate processes (Li et al., 2018). As cybersecurity threats have rapidly increased, several problems have arisen in different areas, such as finance, credibility verification, enforcement, and company operations (Madakam et al., 2015). Given the limited storage capacity of IoT devices, cloud computing often serves as a model-structured with scalable data storage solution. IoT generally reduces the level of human involvement required in the interaction between consumers and providers (Botta et al., 2016). It has garnered significant interest from both organizations and users because of its notable attributes. Nevertheless, the transition from the current system to the IoT environment may present many challenges in terms of operational mechanisms and data security. The risk of cloud computing and IoT arises from the storage of precious data and information on remote servers. Security threats make IoT devices vulnerable to hackers and intruders, deterring a significant number of people from adopting or transitioning to this technology.

There are multiple factors contributing to the significant increase in recent cyberattacks (Prasad and Chandra, 2024). One of the primary factors contributing to this phenomenon is the availability of user-friendly hacking tools, enabling inexperienced hackers to launch attacks against IoT storage systems without requiring advanced expertise or specialized knowledge (Louvieris et al., 2013; Man and Sun, 2021), Some challenges remain insufficiently addressed in existing research in addressing

various difficulties within the realm of cyberattacks, specifically focusing on Intrusion Detection Systems (IDS) (Mohammadpour et al., 2020; Liu and Dong, 2012) over the last few years. Moreover, researchers have employed various Machine Learning (ML) algorithms to tackle the challenges posed by cyberattacks. For instance, Support Vector Machine (SVM) has been implemented in previous studies (Aslahi-Shahri et al., 2016; Jaber and Rehman, 2020).

Additionally, various ML-based approaches have been explored by authors in this domain, including the k-means algorithm (Best et al., 2022), which has been proposed as a hybrid approach addressing the growing need for flexible algorithms capable of operating across diverse devices in IoT. Similarly, K-Nearest Neighbor (KNN) (Ma and Kaban, 2013) is an effective approach that calculates distances, often using the Euclidean metric, between a query point and its neighbors to identify the closest match. The Genetic Algorithm (GA) (Desale and Ade, 2015), on the other hand, has been employed as a feature selection method for the NSL-KDD dataset. By applying an intersection principle, the approach retains only consistently identified features, enhancing the Naïve Bayes classifier's accuracy. In recent years, the utilization of deep learning-based solutions has increased significantly. However, prior methods face limitations in real-time adaptability and scalability when applied to high-dimensional IoT traffic. Many existing solutions struggle to efficiently process diverse and rapidly evolving data streams, leading to delayed and sometimes inaccurate detection. This highlights the necessity for a robust, low-latency intrusion detection framework capable of handling the complex and dynamic nature of IoT network environments.

To address these challenges and bridge the identified research gaps, this article makes the following key contributions:

- A novel hybrid of the Grey Wolf Optimizer with Tabu Search as a feature selection technique to select the most relevant and informative characteristics
- A stacked LSTM-GRU deep learning model that effectively captures temporal dependencies in IoT network traffic for accurate intrusion detection
- Comprehensive evaluation on NSL-KDD, UNSW-NB15 and Edge-IIoT datasets to demonstrate improvement in accuracy and robustness over state-of-the-art methods
- A scalable and resource-efficient framework suitable for deployment in real world IoT environments

We structure the subsequent sections of this article as follows: First, we provide an overview of the existing research on IDS models. Next, we elaborate on the foundational principles and fundamental concepts of GWO, tabu search, LSTM, and GRU. A detailed outline of the proposed IoT security approach follows this. Then, we present the study's findings and analysis. Finally, we conclude the study with closing remarks and offer recommendations for future research.

Related Work

Edge computing, cloud computing, grid computing, and other IoT applications require IDS for security monitoring (Hernandez-Jaimes et al., 2023). The of Convolutional Neural Networks (CNN)-based approach presents an IDS that involves detection models using advanced deep learning techniques. The framework analyzes two forms of input: Raw network traffic, which provides unprocessed, comprehensive information about network activity, and the features derived from this data, which highlight key patterns and attributes essential for intrusion detection. By incorporating both raw and processed inputs, the approach ensures a more robust and accurate detection mechanism; this study proposes two artificial neural network models for IDS using the NSL-KDD dataset (Nguyen et al., 2018). The Recurrent Neural Network (RNN) was implemented for IDS (Zarai et al., 2020), and various other models were suggested (Kaushik et al., 2023), which have been well recognized in the field. Feature selection is an important component of IDS, and this selection issue has been effectively taken care of by a range of conventional classifiers (Thakkar and Lohiya, 2022). Verma and Ranga (2020) tested different ML classifiers including CART, XGBoost, and Random Forest (RF). Their study emphasized the importance of optimizing classifier parameters using random search algorithms with statistical validation through Nemenyi post-hoc tests. In the same way, Khatib et al. (2021) looked at classifiers such as RF, LDA, and Decision Trees and discovered that oversampling methods like SMOTE significantly improves performance, especially in tasks that require binary classification. These classifiers are noted for their scalability and real-time applicability.

Brun et al. (2018) utilized dense RNNs to detect anomalies in real-time within IoT environments. Their research demonstrates particularly high detection accuracy, with notable success in identifying complex and sophisticated attack patterns. Researchers found that Dense RNNs address IoT security challenges by effectively modeling complex temporal dependencies. Tyagi and Kumar (2021) concentrate on IoT networks, attaining 99% accuracy using decision trees and RF classifiers. They underscore the interpretability and efficacy of these models, especially in managing extensive, real-time datasets.

Thamilarasu and Chawla (2019) further contribute to IoT security by proposing a deep learning-based IDS capable of identifying attacks such as DDoS, blackholes, and wormholes. Their system achieves a 97% true

positive rate, indicating deep learning's potential for detecting IoT-specific threats. On the other hand, (Anthi et al., 2018) report challenges in identifying Denial of Service (DoS) attacks, despite using an ensemble approach. Their framework faces issues with precision and recall, particularly in distinguishing more sophisticated IoT attacks.

Ye et al. (2018) utilized SVM to identify DDoS attacks in a network environment, attaining an accuracy of 95.24%. However, their methodology encounters constraints in extending to various forms of cyberattacks, illustrating a prevalent challenge observed in models tailored for attack-specific identification.

Lopez-Martin et al. (2017) suggested using Conditional Variational Auto Encoders (CVAE) to find intrusions because they work better than conventional classifiers like SVM and RF. CVAE's ability to reconstruct missing features proves particularly useful in handling incomplete datasets. Yihunie et al. (2019) evaluated multiple classifiers and found that RF excelled at minimizing false negatives.

Researchers widely use meta-heuristic optimization methods to address a diverse set of intricate optimization problems, leading to a revolution in this field. Bekri and Diouri (2019) IDS widely employ Meta-Heuristics (MH) algorithms, the author proposed Particle Swarm Optimization (PSO) algorithm and an RF classifier for IDS to detect an attack. The PSO algorithm selects the relevant features from the dataset, while the RF classifier handles detection and classification tasks.

Aljawarneh et al. (2018) proposed a hybrid approach for intrusion detection, which performs data filtering using the gained information to select important features that enhance the model's accuracy. Classifiers such as J48, Meta Pagging, and Random Tree are utilized. Genetic Algorithms (GA), Ant Colony Optimization (ACO), and the GWO algorithm (Safaldin et al., 2021) also utilized in IDS.

The Neuro-Fuzzy Inference System (NFIS) model was improved by Manimurugan et al. (2020) they used the Crow Search Optimization (CSO) algorithm to make it better at finding intrusions. They applied the CSO algorithm to optimize the NFIS parameters. The combination of CSO and NFIS shows how advanced optimization methods can improve the performance of IDS. Combining ML techniques like the harmony search algorithm method and the LSTM classification model, Saba et al. (2022) came up with a new hybrid meta-heuristic approach that makes cloud-based IDS work better. RM et al. (2020) suggest a mix of PCA-GWO-driven DNN classification models that can use the benchmark Kaggle dataset to find intrusions.

Chaganti (2025) proposed a scalable and adaptive IoT security framework that combines AI-driven intrusion detection, blockchain-based trust management and edge computing for real-time, resource-efficient threat response. By leveraging optimization techniques, game theory and differential equations. The framework dynamically balances detection accuracy, energy consumption and response time.

Current IoT IDS face challenges in detecting multi-dimensional threats and generalizing across datasets like Edge-IIoT. While models like SVMs and dense RNNs excel in specific cases, they struggle with real-time anomaly detection and high-volume IoT data. Meta-heuristic algorithms like PSO enhance feature selection but lack integration with advanced deep-learning models. Issues such as precision-recall imbalances, limited robustness to noisy data, and the absence of lightweight, dynamic frameworks hinder the effectiveness of existing IDS solutions.

This article presents a novel and robust IDS model that incorporates a fusion of Deep Learning (DL) and a novel hybrid optimization algorithm. The feature selection was initially accomplished with efficiency through the implementation of the proposed GWOTB algorithm, and intrusion detection is done using an ensemble model of stacked LSTM-GRU.

Materials and Methods

In this study, we selected and implemented three algorithms for feature selection: GWO, Tabu Search, and a hybrid of both. We have implemented a stacked version of LSTM and GRU for classification. Figure 1 illustrates the layout of the proposed work. In a multi-layered IoT security system, the proposed hybrid GWO-based feature selection method can be used to secure both the edge and cloud layers.

GWOTB optimizes feature selection to process only the most critical features, enabling lightweight and efficient intrusion detection in resource-constrained environments. This real-time detection helps mitigate localized threats and reduces the need for constant communication with the cloud while also preserving bandwidth. However, at the cloud layer, the model leverages its deep learning capabilities to perform more complex and large-scale analysis using vast amounts of aggregated data from multiple edge devices. It can keep adapting to new attack patterns by retraining datasets like NSL-KDD and UNSW-NB15 that are both old and new. Figure 2 illustrates the layout of the proposed IDS based Security system for IoT.

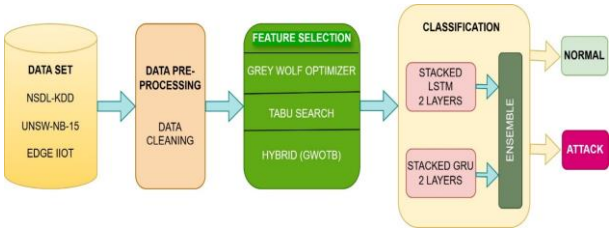

Fig. 1: Proposed Approach

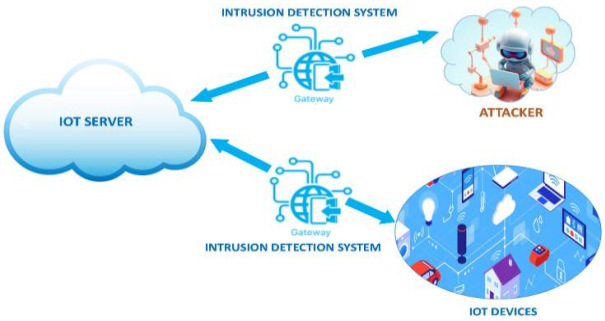

Fig. 2: Proposed IDS based Security system for IoT

GWO

GWO (Mirjalili et al., 2014) is a nature-inspired algorithm that comes from the class of meta-heuristic algorithms which draws its inspiration from the social framework and hunting practices of grey wolves.

Dominance levels determine hierarchical positions within a social group of wolves. These positions are represented by the alpha (α), beta (β), delta (δ) and omega (ω) members. In wolf pack, the alpha holds the top position of dominance, whereas the level of domination and leadership diminishes gradually from the alpha to the omega.

The implementation of this method involves the categorization of a population of potential solutions for a certain optimization problem into distinct types, denoted as α, β and δ. The remaining solutions are encompassed within the set of ω wolves. To execute this process, it is necessary to modify the hierarchy at each iteration before altering the previous results. The subsequent mathematical models update the position of the solutions after the division process is complete. The act of encircling prey is a common predatory behavior observed in various animal species. Wolves typically engage in group hunting behavior. This implies that they engage strategically to capture and consume a target. The Grey Wolf species employs a cooperative hunting strategy wherein they initially pursue their prey collectively, intending to surround it and updating its trajectory and therefore enhancing the likelihood of a successful hunt. By establishing a hierarchical structure, formulating an equation to determine the encirclement process, and identifying the location of the prey, it becomes possible to update the position of each wolf using the subsequent equations. Each iteration updates the positions of α, β and δ wolves by using Eq. (1):

$$\vec{D} = C.\overrightarrow{PW}_p(i) - \overrightarrow{PW}(i)$$
$$\overrightarrow{PW}(i+1) = \overrightarrow{PW}_p(i) - A.\vec{D} \quad (1)$$

Where (i) represents the iteration number, $\overrightarrow{PW}_p(i)$ represents the position of prey $\overrightarrow{PW}$ represents the position of the wolf, C and A are the coefficients of vectors as computed by the Eq. (2):

$$A = 2a.r_1 - a, C = 2.r_2 \quad (2)$$

$$\vec{D}_\alpha = \left|C_1.PW_\alpha - \overrightarrow{PW}\right|, \overrightarrow{PW}_1\,\overrightarrow{PW}_\alpha - A_1.\vec{D}_\alpha \quad (3)$$

Where $\vec{D}_\alpha$ represents the distance between alpha and omega wolves, C_1 is the coefficient of vector:

$$\vec{D}_\beta = \left|C_2.\overrightarrow{PW}_\beta - \overrightarrow{PW}\right|, \overrightarrow{PW}_2 = \overrightarrow{PW}_\beta - A_2.\vec{D}_\beta \quad (4)$$

Where $\vec{D}_\beta$ represents the distance between beta and omega wolves, C_2 represents the coefficient of vector:

$$\vec{D}_\delta = \left|C_3.\overrightarrow{PW}_\delta - \overrightarrow{PW}\right|, PW = \overrightarrow{PW}_\delta - A_3.\vec{D}_\delta \quad (5)$$

Where $\vec{D}_\delta$ represents the distance between delta and omega wolves, C_3 is the coefficient of vector.

The Equation (6) specifies that the top three wolves from the previous iteration will be considered when upgrading the wolf's position:

$$\overrightarrow{PW} = \frac{\left(\overrightarrow{PW}_1 + \overrightarrow{PW}_2 + \overrightarrow{PW}_3\right)}{3} \quad (6)$$

Tabu Search

Tabu Search is considered an extension of iterative optimization techniques like Simulated Annealing (SA). The Tabu Search algorithm is founded on the principle that intelligent problem-solving necessitates the integration of adaptive memory and responsive exploration. It is recognized that the procedure in adaptive approach employs several techniques, such as linear programming algorithms and specialized heuristics, to effectively address the constraints associated with local optimality.

The adaptive memory functionality of Tabu Search enables the execution of algorithm that possess the ability to efficiently explore the solution space. The focus on adaptive exploration, especially in the context of tabu search, is based on the idea that a strategic decision that isn't the best can often lead to more useful insights than a random decision that leads to favourable results. Tabu Search implements limitation to direct the exploration towards varied geographical areas. These limitations relate to memory structures, which we conceptualize as cognitive qualifiers. According to Gendreau and Potvin (2006) Intelligence is contingent upon the presence of adaptive memory and responsive exploration. For example, when ascending a mountain, an individual engages in adaptive memory, recalling qualities of previously traversed trails, and then employs responsive exploration to make strategic decisions regarding the optimal route to reach the summit or initiate the descent.

Therefore, Tabu Search incorporates responsive exploration: It understands that a less-than-ideal strategic choice could lead to more useful information than a randomly chosen choice, ultimately creating beneficial solutions. The Tabu Search algorithm uses memory to store information about previously explored solutions, avoiding revisiting them.

LSTM

LSTM (Sherstinsky, 2020) networks are employed in our intrusion detection system to effectively capture long-range temporal dependencies within network traffic data.

Each LSTM unit consists of a memory cell and three primary gates, input, forget, and output, that control the flow of information. The input gate regulates which portions of the current input influence the cell state, the forget gate selectively removes outdated information, and the output gate determines the data passed to the next layer. This gating mechanism allows LSTM to dynamically learn and remember important patterns over time, making it particularly suitable for detecting persistent or evolving attack behavior in IoT environments. Figure 3a illustrates the key components of a typical LSTM block. This includes the gating mechanisms, the input signal denoted as $x^{(t)}$, activation functions, the output signal represented as $y^{(t)}$, and peephole connections.

In our architecture, a stacked LSTM configuration is used, comprising two hidden layers with 128 and 64 cells respectively, followed by a dense SoftMax layer to enable classification of network traffic instances. This design enhances the model's ability to generalize across diverse attack types while maintaining robustness to temporal variations. The input activation is computed as:

$$S^t = G(W_z.X^t + R_z.y^{t-1} + b_z) \quad (7)$$

Where we used two sets of weights, *Wz* and *Rz*. We associate these weights with the current input, represented as $x^{(t)}$, and the output from the previous time step, represented as $y^{(t-1)}$. The bias term, symbolized as *bz*, functions as a vector of weight values.

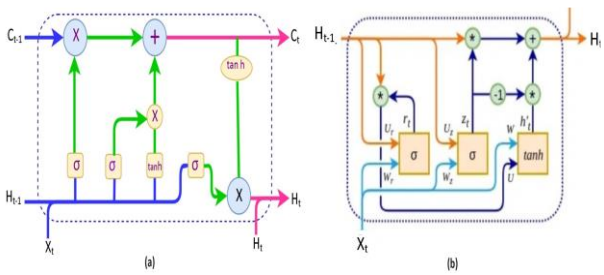

Fig. 3: (a) LSTM Architecture. (b) GRU Architecture

The input gate in an LSTM is a crucial component. An LSTM updates its input gate using three component values: The current input, $x^{(t)}$, represents the input data that we want to process and incorporate into the LSTM's memory. The cell value $c^{(t-1)}$ represents the internal memory state of the LSTM from the previous time step. $Y^{(t-1)}$ represents the output of the preceding run or time step. The input gate is computed as:

$$IG^t = \sigma(b + wx^t + r.y^{t-1} + p \odot c^{t-1}) \quad (8)$$

w, *r*, and p represent the weights corresponding to the current input, the prior output $y^{(t-1)}$, and the preceding cell state $c^{(t-1)}$, respectively. *b* denotes the bias vector.

At a certain time step t, the forget gate calculates an activation value, referred to as $f^{(t)}$. We use this activation value to identify the information that we remove from the previous cell state. Eq. (9) computes the activation value, $f^{(t)}$. The forget gate utilizes the current input, prior output, previous cell state, and a bias function specific to the forget gate:

$$f^t = \sigma\left(W_f.x^t + p_f \odot .c^{t-1} + R_f.y^{t-1} + b_f\right) \quad (9)$$

Where *bf* represents the bias weight vector used in the forget gate, and W_f , P_f, and R_f represent weights that are associated with different components of the forget gate.

In Equation (10), we have calculated the cell value by blending several components, which are given by the equations mentioned below:

$$c^t = c^{t-1} \odot f^t + z^t \odot i^t \quad (10)$$

Where c^{t-1} denotes cell value in the last time step, *zt* represents block input, it denotes input gate, and *ft* denotes forget gate.

In this step, we calculate the output gate's value by combining the current input with the cell value output $c^{(t-1)}$ from the previous step, using Eq. (11):

$$o^t = \sigma(V_o.x^t + S_o.y^{t-1} + Q_o \odot c^t + b_o) \quad (11)$$

Where V_o, S_o and Q_o represent associated weights and b_o denotes bias.

The block input and output activation functions are computed in Eq. (12) and (13), *g* and *h*, are hyperbolic tangents that represent the logistic sigmoid activation function for gate activation:

$$\sigma(x) = 1/(e^{1-x} + 1) \quad (12)$$

$$tanh(x) = g(x) = h(x) \quad (13)$$

GRU

The inclusion of Gated Recurrent Units (GRUs) in proposed ensemble framework is driven by their computational efficiency and effectiveness in modeling sequential dependencies, particularly in scenarios constrained by latency and resource limitations. Chung et al. (2014) suggested GRU as a straightforward iteration of the LSTM cell, employing a single concealed state that functions as an update gate and mirrors the operation of both the input and output gates. Figure 3b depicts the flow of data and control within a GRU block, including its key components: The update gate z_t, reset gate r_t, candidate activation , and the final hidden state h_t. The update gate and the reset gate, two gating approaches, form the foundation of the GRU architecture. This gating mechanism allows the model to control the transfer of information between consecutive time steps. The reset gate controls the degree of discard for previous state information, enabling the model to reset and focus on fresh input data.

The update gate, on the other hand, regulates the integration of new input and previous state information into the current state. These gates work together to allow the GRU to dynamically adjust and discard information as needed, which in turn makes it a highly valuable tool for tasks involving sequential data analysis and modelling. Equation (14) represents the hidden gate of GRU:

$$h_t = h_t * z_t + h_t * (1 - z_t) \tag{14}$$

Update gate is represented by the Eq. (15):

$$z_t = \sigma([h_{t-1}, x_t] * W_z) \tag{15}$$

The reset gate is represented by the Eq. (16):

$$r_t = \sigma([h_{t-1}, x_t] * W_r) \tag{16}$$

In reset gate the hyperbolic tangent function also known as remember gate is represented by the Eq. (17):

$$h_t = tanh([r_t * h_{t-1}, x_t] * W) \tag{17}$$

Intrusion Detection

The proposed model comprises five fundamental processes: Normalization, encoding, Stratified Sampling, feature selection, and classification using a stacked LSTM GRU network.

Frequency Encoding

We have utilized frequency encoding to convert categorical data into significant numerical representations. Unlike basic label encoding or one-hot encoding, frequency encoding replaces each category with how often it appears in the dataset. This is a more informative transformation. For a categorical variable X with categories x1, x2,…,xn, the frequency of each category xi is determined using Eq. (18):

$$\text{frequency(xi)} = \frac{\text{count(xi)}}{\text{N}} \tag{18}$$

Normalization

In this step, normalization operation is performed on the dataset using min-max normalization using the Eq. (19). Figures 4, 5, and 6 display the correlation heat maps for the normalized NSL-KDD, UNSW-NB15, and Edge-IIoT datasets, respectively. These heatmaps reveal inter-feature relationships, where red indicates high positive correlation and blue denotes high negative correlation. This analysis informed the feature selection process by identifying and eliminating redundant features, ensuring the retained attributes contribute unique and meaningful information to the classification task:

$$\text{Z}_{\text{norm}} = \text{Z} - \text{Zmin(x)/max(z)} - \text{min(z)} \tag{19}$$

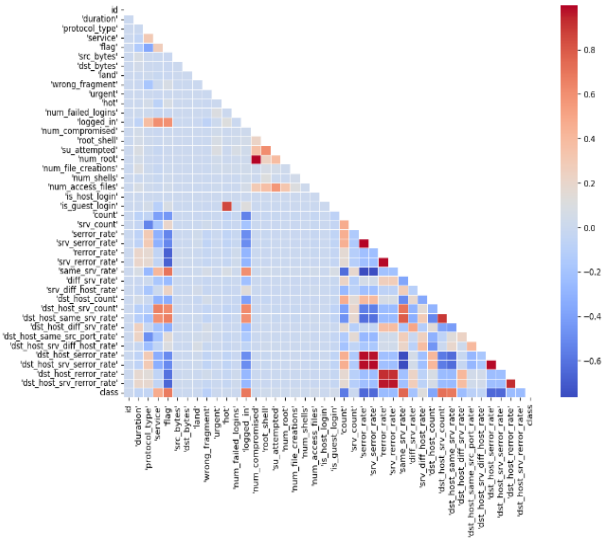

Fig. 4: Heatmap normalized NSL-KDD dataset

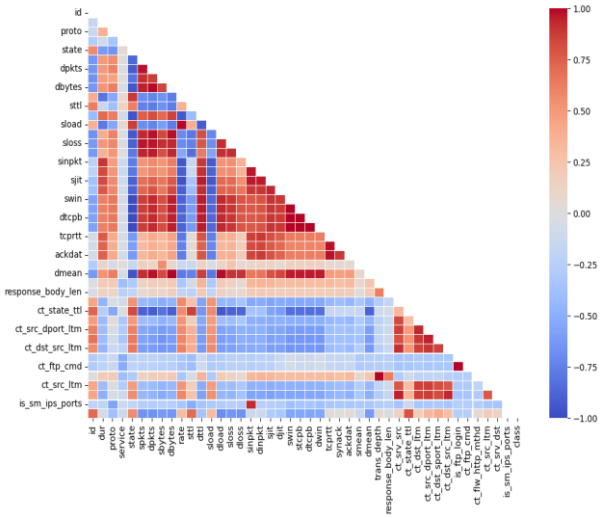

Fig. 5: Heatmap normalized, UNSW-NB15 dataset

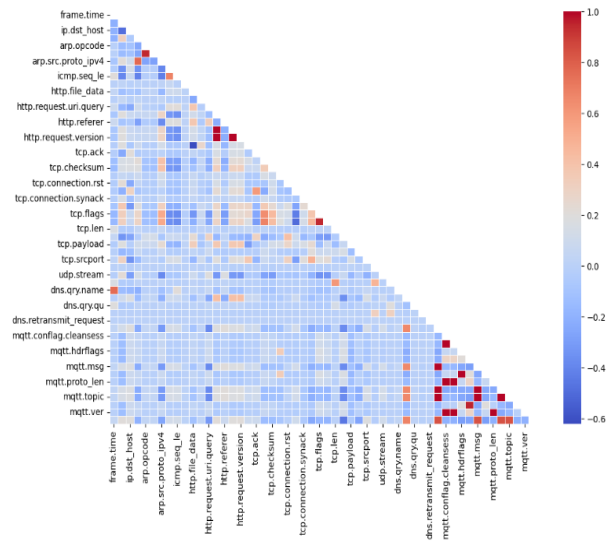

Fig. 6: Heatmap normalized, Edge-IIoT dataset

Sampling

Stratified sampling is used to split data into training 70%, validation 15% and testing 15% sets. To address class imbalance mainly in Edge-IIoT and UNSW-NB15, we applied random under sampling and SMOTE on the training set.

Feature Selection

In this step, we perform the process of selecting specific features from the given dataset. Each feature in the dataset holds some significance, but not all features are necessary for optimal performance. Showing how well it can find network intrusions. Many times, selecting a set of features will lead to an increase in the accuracy of the model.

Proposed Algorithm for feature selection is mentioned.

Selected Features from Different Datasets Using the Proposed Algorithm

NSL-KDD Dataset

'Wrong fragment', 'dst host srv rerror rate', 'srv rerror rate', 'srv serror rate', 'dst host srv diff host rate', 'protocol type', 'dst host diff srv rate', 'diff srv rate', 'dst host same src port rate', 'dst host same srv rate', 'land', 'root shell', 'dst host serror rate', 'is guest login', 'same srv rate', 'dst host srv serror rate'.

UNSW-NB15 Dataset

'State', 'is sm ips ports', 'dpkts', 'dttl', 'is ftp login', 'sload', 'sloss','djit', 'swin', 'dwin', 'synack"ct srv dst', 'ct state ttl'.

Edge-IIoT Dataset

'Tcp.flags', 'mqtt.msg', 'ip.srchost', 'ip.dsthost', 'http.request.method', 'tcp.connection.syn', 'dns.qry.name', 'icmp.transmit timestamp', 'udp.port', 'tcp.ack', 'tcp.srcport', 'tcp.dstport', 'tcp.payload', 'arp.dst.proto ipv4', frame.time', 'tcp.seq', 'udp.time delta, 'tcp.len'.

Classification

In this step, we implemented two distinct neural network architectures: A stacked LSTM and a stacked GRU. For the LSTM model, we utilized two layers, consisting of 128 and 64 cells, respectively. We stacked these layers to enhance the model's learning capacity. Similarly, for the GRU model, we constructed a two-layer network, with the first layer comprising 32 cells and the second layer containing 16 cells.

Proposed Algorithm

Initialize the Grey Wolf pack
Determine the population size N
Generate N initial wolf positions randomly within the search space
Evaluate the fitness of each wolf using the objective function.
Set the maximum number of iterations (MaxIter) and initialize the iteration counter (iter) to 1
Initialize the Tabu List as an empty list
while iter ≤ MaxIter do
Calculate fitness values for all wolves.
Identify the best wolf (alpha) with the highest fitness.
Identify the second-best wolf (beta) and the third-best wolf (gamma)
for each wolf do
Update its position using the following formulas
For alpha: newPosition = wolfPosition + A (2 · r1 − 1) · |C · alphaPosition − wolfPosition|
For beta: newPosition = wolfPosition + A (2 · r2 − 1) · |C · betaPosition − wolfPosition|
For gamma: newPosition = wolfPosition + A (2 · r3 − 1) · |C · gammaPosition − wolfPosition|
end for
Apply Tabu Search to refine the positions:
for each wolf do
if a new position is in the Tabu List then
Ignore it and move to the next position.
else
Perform fitness evaluation of the new position and update the Tabu List.
end if
end for
amend the Tabu List:
if the Tabu List exceeds its maximum allowed size then
Remove the oldest element from the Tabu List.
end if
Add the newly evaluated positions to the Tabu List.
Increment the iteration counter (iter) by 1.
end while
return the best wolf (alpha) as the solution

We concluded both models with a dense layer, incorporating a SoftMax activation function to facilitate classification tasks. Additionally, the Adam optimizer was employed. We executed each model independently to evaluate its individual performance. We then devised an ensemble based on soft voting that integrated the predictions from both the LSTM and GRU models.

Experimental Setup

We implemented the proposed model on a machine running 64-bit Windows 11 Pro, equipped with a 12-generation i7 processor with 12 cores operating at 4.9 GHz and 16 GB of RAM. We implement the LSTM-GRU stacked classifier using the deep learning library TensorFlow. The hyperparameters used are listed in Table 1.

Dataset

In this study, we used three datasets, i.e., UNSW-NB15 (Moustafa and Slay, 2015), Edge-IIoT (Ferrag et al., 2022) and NSL-KDD (Tavallaee et al., 2009) as they are recently generated datasets from real-time traffic. The edge IIOT dataset has 62 features; it has a total of 11223940 records in the normal class and 9728708 in the attack class; and the dataset has 11 classes for attack. The UNSW-NB15 dataset contains 175431 records and nine attack classes. The NSL-KDD dataset comprises a total of 148517 records and four attack classes.

Results

Performance Matrices

For evaluation of proposed model, we used accuracy, precision, sensitivity and specificity they are mentioned in the Eq. (20), (21), (22) and (23) respectively:

$$\text{Accuracy} = \frac{(TP+TN)}{(TP+TN+FP+FN)} \quad (20)$$

$$\text{Precision} = \frac{TP}{(TP+FP)} \quad (21)$$

$$\text{Sensitivity} = \frac{TP}{(TP+FN)} \quad (22)$$

$$\text{Specificity} = \frac{TN}{(TN+FP)} \quad (23)$$

Where TP is True positive, TN is True Negative, FP is False Positive and FN is False Negative.

Performance Evaluation

A remarkable 99.50% accuracy is achieved by the proposed method on the NSL-KDD dataset, showing how well it can find network intrusions. This performance highlights its superiority compared to other approaches. Additionally, Figure 7 shows accuracy of the proposed work on benchmark dataset and it inherently represents key statistical indicators, including Interquartile Ranges (IQR), medians, and outliers, effectively capturing variance and confidence levels in model performance.

To see how well GWOTB-LSTM-GRU-STACK (proposed) works, we compare it to GWO-GRU, GWO-LSTM, GWOTB-GRU, and GWOTB-LSTM using the three datasets mentioned in Table 2.

Table 1: Hyper Parameters and Their Values.

Model	Hyper Parameter	Value
LSTM	LSTM Cells	[128, 64]
GRU	GRU Cells	[64, 32]
GRU and LSTM	Input Shape	1D
GRU and LSTM	Optimizer	Adam
GRU and LSTM	Activation Function	Softmax
GRU and LSTM	Learning Rate	0.001
GRU and LSTM	Dropout Rate	0.2
GWOTB	Population Size	100
GWOTB	Max Iteration	50
GWOTB	Lower Bound	0
GWOTB	Upper Bound	100
GWOTB	r1, r2, r3	[0,1]
GWOTB	a	2

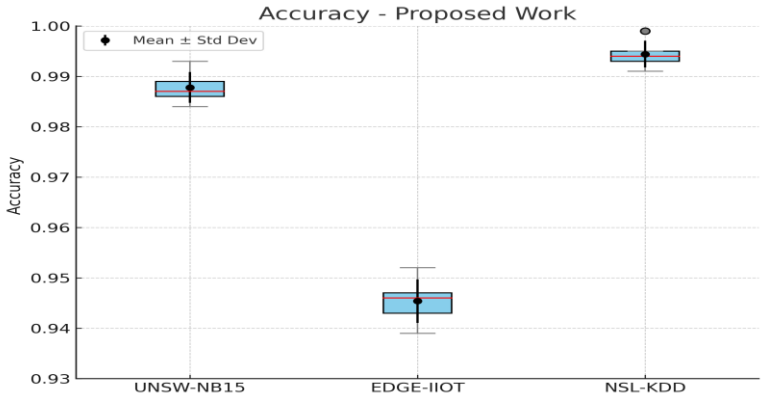

Fig. 7: Accuracy of the Proposed IDS across Benchmark Datasets

Table 2: Performance of the proposed techniques

`Dataset	Technique	Accuracy%	Sensitivity%	Specificity%	Precision%
	GWO-GRU	98.7	98.91	98.46	98.68
	GWO-LSTM	98.42	97.11	98.41	97.75
NSL- KDD	GWOTB-GRU	98.71	98.95	99.01	98.98
	GWOTB-LSTM	98.53	98.70	98.10	98.39
	Proposed	99.55	99.21	99.45	98.33
	GWO-GRU	97.51	98.13	98.41	98.27
	GWO-LSTM	97.15	98.5	98.95	98.72
UNSW- NB-15	GWOTB-GRU	97.71	98.64	98.12	98.38
	GWOTB-LSTM	97.12	98.57	97.94	98.25
	Proposed	98.57	98.48	98.71	98.59
	GWO-GRU	92.80	84.54	98.12	90.72
	GWO-LSTM	92.38	86.21	97.51	91.43
EDGE-IIOT	GWOTB-GRU	93.21	90.12	98.45	94.12
	GWOTB-LSTM	93.11	89.87	97.11	93.33
	Proposed	95.08	94.13	98.25	96.12

The proposed approach also exhibits exceptional sensitivity of 99.21% and specificity of 99.45%. In the UNSWNB-15 dataset, the proposed method continues to perform well, with an accuracy of 98.57%, displaying a good sensitivity of 98.48% and specificity of 98.71%. On the Edge-IIoT dataset, the proposed method achieves an accuracy of 95.08%, underscoring its effectiveness over GWO-GRU, GWO-LSTM, GWOTB-GRU, and Figure 8 highlights the confusion matrices across three datasets, comparing classification performance.

The confusion matrices illustrate strong class-wise performance across all datasets. For NSL-KDD (Figure 8c), the model achieves near-balanced accuracy with 49.56% true negatives and 49.76% true positives. In UNSW-NB15 (Figure 8a), attack detection stands at 51.45% with low misclassification. In Edge-IIoT Figure (8b), the model detects 57.14% tire positive, though with a slightly higher false positive rate 3.76%. These results affirm the model's effectiveness in distinguishing between normal and malicious traffic in diverse IoT environments.

The confusion matrices for each dataset present the values for TN, TP, FN, and FP, providing a detailed view of the model's performance. These metrics help to evaluate the system's ability to correctly classify instances, including detecting true positives and true negatives, as well as identifying false negatives and false positives. GWOTB-LSTM demonstrate solid results, showcasing its potential to enhance network security by accurately identifying anomalies and intrusions. These results underscore the significance of advanced techniques in safeguarding critical systems and data from cyberattacks.

Discussion

Table 4 displays eleven effective techniques, all published in reputable academic journals, to conduct a comprehensive assessment of the proposed system, all performance values in Table 4 were directly extracted from the respective referenced studies and evaluated under similar dataset conditions. Across all datasets, the models generally exhibit high accuracy values, ranging from 92.38 to 99.49%. This indicates that these models can make accurate predictions on intrusion detection tasks. The proposed technique, consistently achieves the highest accuracy on all datasets, indicating its effectiveness in uncovering patterns and associations within the data.

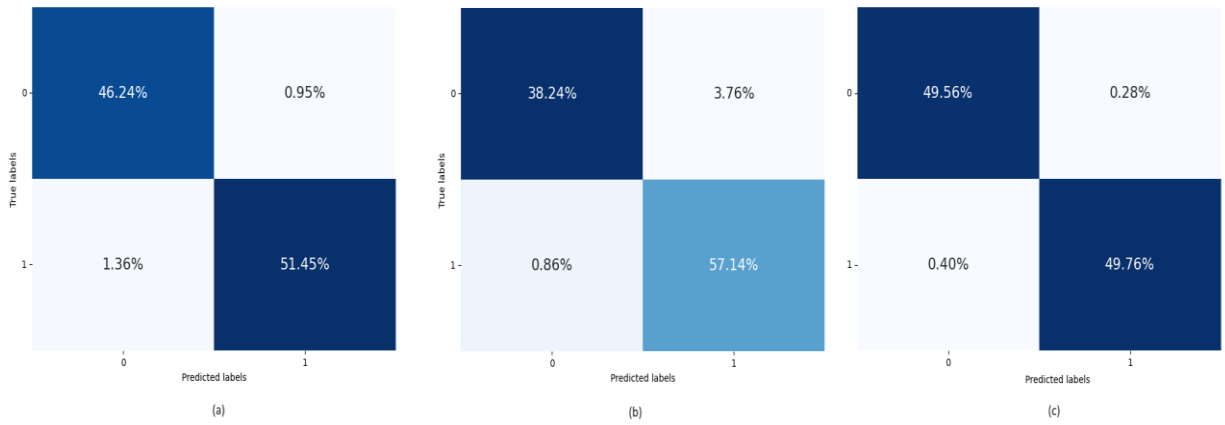

Fig. 8: (a) Confusion Matrix with UNSW-NB15 (b) Confusion Matrix with Edge-IIoT (c) Confusion Matrix with NSL-KDD

Diro and Chilamkurti (2018) have obtained an accuracy of 98.27% using the shallow neural network and deep neural network, which proved their effectiveness on the NSL KDD dataset. The gradual increase from shallow to deep models shows deep learning architecture's benefits.

Alhowaide et al. (2021) proposed an Edge-ENClf approach that has shown promising accuracy on the NSL-KDD and UNSW-NB15 datasets. (Hassan et al., 2022) have employed the RF algorithm in conjunction with the Modified Manta Ray Foraging Optimization Algorithm. This technique performed well on the CICIDS and NSL-KDD datasets. The research's creative use of a modified optimization method contributed to its success.

Mehedi et al. (2023) suggested deep learning and deep transfer learning as methods to identify network intrusions on the UNSW-NB15 dataset. The network had 87.3% accuracy, 86% precision, and 86% recall. The accuracy may seem lower than in prior investigations, but the precision and recall rates show a fair trade-off between spotting intrusions and minimizing false positives. Sharma et al. (2023) utilized both Deep Neural Networks (DNN) and Generative Adversarial Networks (GAN). On the UNSW-NB15 dataset, deep neural networks and GANs achieved 91% accuracy.

Ding et al. (2023) make use of generative models, which may aid in learning the underlying data distributions for better detection. This suggested technique achieved high accuracy on the Edge-IIoT dataset and UNSW-NB15 datasets at 94.96 and 98.41%, respectively. In the context of Edge-IIoT deployment, computational trade-offs and false alarm rates are crucial considerations. Our proposed model maintains a high accuracy of 95.08% on the Edge-IIoT dataset while achieving a sensitivity of 94.13% and a precision of 96.12%, which reflects the system's strong ability to correctly identify threats. We observed a slightly elevated false positive rate when compared to NSL-KDD and UNSW-NB15 datasets. This trade-off is partially attributable to the high-dimensional nature and real-time variability of edge traffic.

To validate the observed gains, we conducted paired two-tailed t-tests across five independent experimental runs using different random seeds for each model. These tests were applied to compare the ensemble model with each relevant baseline, including GWO-GRU, GWOTB-GRU, GWOTB-LSTM and the no-feature-selection variant.

As shown in Table 3, the resulting p-values were consistently < 0.01 and the 95% confidence intervals were tight, indicating statistically significant improvements in accuracy. These results robustly confirm that the proposed ensemble model outperforms other configurations.

We have conducted a comprehensive evaluation over four configurations: No feature selection, GWO alone, Tabu Search alone and the proposed hybrid method. Figure 9 depicts the results which demonstrate the progressive improvement in classification accuracy when employing more advanced feature selection strategies. From a computational standpoint, while the hybrid feature selection introduces additional preprocessing time at the same time it significantly reduces model complexity during training and inference by eliminating redundant

Features, this gain in efficiency is beneficial in resource-constrained edge environments. The proposed feature selection has a worst-case complexity of $O(N \cdot F \cdot T)$, the stacked LSTM-GRU model exhibits a training complexity of $O(E \cdot n \cdot h^2)$ while inference is reduced to $O(Fs \cdot h)$ with $Fs \ll F$ due to prior feature reduction. These analyses demonstrate a balanced trade-off between performance and efficiency which is suitable for edge and cloud-based IoT environments.

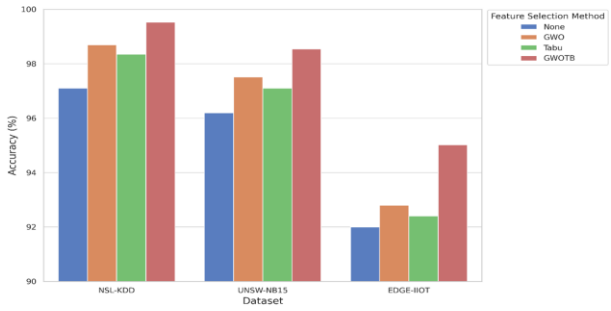

Fig. 9: Effects of feature selection on Accuracy

Table 3: Statistical comparison of the proposed model with baseline models

Dataset	Compared Models	95% CI (±)	p-value	Statistically Significant
NSL-KDD	Proposed vs GWO-GRU	±0.18	0.0031	Yes
NSL-KDD	Proposed vs GWOTB-GRU	±0.19	0.0048	Yes
NSL-KDD	Proposed vs No Feature Selection (FS)	±0.18	0.0009	Yes
UNSW-NB15	Proposed vs GWO-GRU	±0.21	0.0056	Yes
UNSW-NB15	Proposed vs GWOTB-LSTM	±0.21	0.0023	Yes
Edge-IIoT	Proposed vs GWO-LSTM	±0.23	0.0042	Yes
Edge-IIoT	Proposed vs No FS	±0.24	0.0015	Yes

Table 4: Comparison with past studies

Reference	Algorithm	Results	Dataset
Verma and Ranga (2020)	RF, GBM, XGB AND MLP	Accuracy = 94.99%	CIDDS-001, UNSW-NB15
Diro and Chilamkurti (2018)	Shallow NN & DNN	Accuracy = 98.27%	NSL-KDD
Kozik et al.	ELM	Accuracy = 83%	Self-generated
Pajouh et al. (2019)	LDA, NB, CF-KNN	Accuracy = 84.82%	NSL-KDD
Chen et al. (2020)	Decision Tree	Accuracy = 99.98%, Precision = 97.38%	CIC-IDS2017
Kushwah and Ranga (2021)	BH-ELM	Accuracy 99.23% 99.5% (CICIDS 2017)	NSL-KDD, CICIDS 2017
Alhowaide et al. (2021)	Edge-ENClf	Accuracy = 98% (NSL-KDD), 95% (UNSW-NB15)	NSL-KDD, UNSW-NB15
Hassan et al. (2022)	Modified MRFO algorithm and random forest	Accuracy = 98.8% 99.3% (CIC-IDS2017)	NSL-KDD, CIC-IDS2017
Mehedi et al. (2023)	DL and DTL-based residual neural network	Accuracy = 87.3%, Precision = 86%, Recall = 86%	Self-generated
Sharma et al. (2023)	DNN and GAN	Accuracy = 91%	UNSW-NB15
Ding et al. (2023)	eRSR block, TRB, and DB	Accuracy = 94.96% (Edge-IIoT), 98.41% (UNSW-NB15)	Edge-IIoT, UNSW-NB15

Conclusion

The proposed framework for intrusion detection in IoT environments demonstrates significant advancements in accuracy, efficiency, and reliability. Through a carefully structured approach comprising data pre-processing, a novel hybrid feature selection method, and a deep transfer learning model utilizing stacked LSTM and GRU networks, the system addresses critical challenges in IoT security. Experimental results on benchmark datasets, including NSL-KDD, UNSW-NB15, and Edge-IIoT, confirm the framework's superior performance compared to existing methods such as GWO, CNN, SVM-Bagging, Decision Tree, and ELM. While the current framework has shown promising results, opportunities for further improvement remain. Future research can explore optimizing the framework for energy efficiency, particularly in resource-constrained IoT environments. Additionally, adapting the model to incorporate dynamic updates in response to evolving cyber threats could enhance its resilience. By addressing these aspects, the proposed solution can contribute to more robust and scalable intrusion detection systems, ensuring enhanced security for the rapidly expanding IoT ecosystem.

Acknowledgment

The Authors gratefully acknowledge the support of SPSU faculty.

Funding Information

This study received no external funding.

Author's Contributions

Kapil Shrivastava: Responsible for Conceptualization, methodology, original draft preparation.

Manish Tiwari: Responsible for data analysis, experimental setup, and result validation.

Prasun Chakrabarti: Responsible for Manuscript refinement and final proofreading.

Ethics

This study did not involve human or animal subjects, and no ethical approval was required.

Data Availability

The datasets used in this study are publicly available.

Conflict of Interest

The authors declare no conflict of interest.

References

Alhowaide, A., Alsmadi, I., & Tang, J. (2021). Ensemble Detection Model for IoT IDS. *Internet of Things*, *16*, 100435. https://doi.org/10.1016/j.iot.2021.100435

Aljawarneh, S., Aldwairi, M., & Yassein, M. B. (2018). Anomaly-based intrusion detection system through feature selection analysis and building hybrid efficient model. *Journal of Computational Science*, *25*, 152–160. https://doi.org/10.1016/j.jocs.2017.03.006

Anthi, E., Williams, L., & Burnap, P. (2018). Pulse: an adaptive intrusion detection for the internet of things. *Proceeding of the Living in the Internet of Things: Cybersecurity of the IoT*, 4–35. https://doi.org/10.1049/cp.2018.0035

Aslahi-Shahri, B. M., Rahmani, R., Chizari, M., Maralani, A., Eslami, M., Golkar, M. J., & Ebrahimi, A. (2016). A hybrid method consisting of GA and SVM for intrusion detection system. *Neural Computing and Applications*, *27*(6), 1669–1676. https://doi.org/10.1007/s00521-015-1964-2

Bekri, M. E., & Diouri, O. (2019). Pso Based Intrusion Detection: A Pre-Implementation Discussion. *Procedia Computer Science*, *160*, 837–842. https://doi.org/10.1016/j.procs.2019.11.002

Best, L., Foo, E., & Tian, H. (2022). Utilising K-Means Clustering and Naive Bayes for IoT Anomaly Detection: A Hybrid Approach. *Measurement and Instrumentation*, *43*, 177–214. https://doi.org/10.1007/978-3-031-08270-2_7

Botta, A., de Donato, W., Persico, V., & Pescapé, A. (2016). Integration of Cloud computing and Internet of Things: A survey. *Future Generation Computer Systems*, *56*, 684–700. https://doi.org/10.1016/j.future.2015.09.021

Brun, O., Yin, Y., Gelenbe, E., Kadioglu, Y. M., Augusto-Gonzalez, J., & Ramos, M. (2018). Deep Learning with Dense Random Neural Networks for Detecting Attacks Against IoT-Connected Home Environments. *Communications in Computer and Information Science*, 79–89. https://doi.org/10.1007/978-3-319-95189-8_8

Chaganti, K. C. (2025). A Scalable, Lightweight AI-Driven Security Framework for IoT Ecosystems: Optimization and Game Theory Approaches. *IEEE Access*, *13*, 72235–72247. https://doi.org/10.1109/access.2025.3558623

Chen, Y.-W., Sheu, J.-P., Kuo, Y.-C., & Van Cuong, N. (2020). Design and Implementation of IoT DDoS Attacks Detection System based on Machine Learning. *Proceeding of the European Conference on Networks and Communications (EuCNC)*, 122–127. https://doi.org/10.1109/eucnc48522.2020.9200909

Chung, J., Gülçehre, Ç., Cho, K., & Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. *ArXiv*, 1–8. https://doi.org/10.48550/arxiv.1412.3555

Desale, K. S., & Ade, R. (2015). Genetic algorithm based feature selection approach for effective intrusion detection system. *Proceeding of the International Conference on Computer Communication and Informatics (ICCCI)*, 1–6. https://doi.org/10.1109/iccci.2015.7218109

Ding, W., Abdel-Basset, M., & Mohamed, R. (2023). DeepAK-IoT: An effective deep learning model for cyberattack detection in IoT networks. *Information Sciences*, *634*, 157–171. https://doi.org/10.1016/j.ins.2023.03.052

Diro, A. A., & Chilamkurti, N. (2018). Distributed attack detection scheme using deep learning approach for Internet of Things. *Future Generation Computer Systems*, *82*, 761–768. https://doi.org/10.1016/j.future.2017.08.043

Ferrag, M. A., Friha, O., Hamouda, D., Maglaras, L., & Janicke, H. (2022). Edge-IIoTset: A New Comprehensive Realistic Cyber Security Dataset of IoT and IIoT Applications for Centralized and Federated Learning. *IEEE Access*, *10*, 40281–40306. https://doi.org/10.1109/access.2022.3165809

Gendreau, M., & Potvin, J.-Y. (2006). *Tabu search*. https://doi.org/10.1007/0-387-28356-0_6

Hassan, I. H., Abdullahi, M., Aliyu, M. M., Yusuf, S. A., & Abdulrahim, A. (2022). An improved binary manta ray foraging optimization algorithm based feature selection and random forest classifier for network intrusion detection. *Intelligent Systems with Applications*, *16*, 200114. https://doi.org/10.1016/j.iswa.2022.200114

Hernandez-Jaimes, M. L., Martinez-Cruz, A., Ramírez-Gutiérrez, K. A., & Feregrino-Uribe, C. (2023). Artificial intelligence for IoMT security: A review of intrusion detection systems, attacks, datasets and Cloud–Fog–Edge architectures. *Internet of Things*, *23*, 100887. https://doi.org/10.1016/j.iot.2023.100887

Jaber, A. N., & Rehman, S. U. (2020). FCM–SVM based intrusion detection system for cloud computing environment. *Cluster Computing*, *23*(4), 3221–3231. https://doi.org/10.1007/s10586-020-03082-6

Kaushik, B., Sharma, R., Dhama, K., Chadha, A., & Sharma, S. (2023). Performance evaluation of learning models for intrusion detection system using feature selection. *Journal of Computer Virology and Hacking Techniques*, *19*(4), 529–548. https://doi.org/10.1007/s11416-022-00460-z

Khatib, A., Hamlich, M., & Hamad, D. (2021). Machine Learning based Intrusion Detection for Cyber-Security in IoT Networks. *E3S Web of Conferences*, *297*, 01057. https://doi.org/10.1051/e3sconf/202129701057

Kozik, R., Choraś, M., Ficco, M., & Palmieri, F. (2018). A scalable distributed machine learning approach for attack detection in edge computing environments. *Journal of Parallel and Distributed Computing*, *119*, 18–26. https://doi.org/10.1016/j.jpdc.2018.03.006

Kushwah, G. S., & Ranga, V. (2021). Distributed denial of service attack detection in cloud computing using hybrid extreme learning machine. *TURKISH JOURNAL OF ELECTRICAL ENGINEERING & COMPUTER SCIENCES*, *29*(4), 1852–1870. https://doi.org/10.3906/elk-1908-87

Li, S., Xu, L. D., & Zhao, S. (2018). 5G Internet of Things: A survey. *Journal of Industrial Information Integration*, *10*, 1–9. https://doi.org/10.1016/j.jii.2018.01.005

Liu, K., & Dong, L. (2012). Research on Cloud Data Storage Technology and Its Architecture Implementation. *Procedia Engineering*, *29*, 133–137. https://doi.org/10.1016/j.proeng.2011.12.682

Lopez-Martin, M., Carro, B., Sanchez-Esguevillas, A., & Lloret, J. (2017). Conditional Variational Autoencoder for Prediction and Feature Recovery Applied to Intrusion Detection in IoT. *Sensors*, *17*(9), 1967. https://doi.org/10.3390/s17091967

Louvieris, P., Clewley, N., & Liu, X. (2013). Effects-based feature identification for network intrusion detection. *Neurocomputing*, *121*, 265–273. https://doi.org/10.1016/j.neucom.2013.04.038

Ma, Z., & Kaban, A. (2013). K-Nearest-Neighbours with a novel similarity measure for intrusion detection. *Proceeding of the UK Workshop on Computational Intelligence (UKCI)*, 266–271. https://doi.org/10.1109/ukci.2013.6651315

Madakam, S., Ramaswamy, R., & Tripathi, S. (2015). Internet of Things (IoT): A Literature Review. *Journal of Computer and Communications*, *03*(05), 164–173. https://doi.org/10.4236/jcc.2015.35021

Man, J., & Sun, G. (2021). A Residual Learning-Based Network Intrusion Detection System. *Security and Communication Networks*, *2021*, 1–9. https://doi.org/10.1155/2021/5593435

Manimurugan, S., Majdi, A., Mohmmed, M., Narmatha, C., & Varatharajan, R. (2020). Intrusion detection in networks using crow search optimization algorithm with adaptive neuro-fuzzy inference system. *Microprocessors and Microsystems*, *79*, 103261. https://doi.org/10.1016/j.micpro.2020.103261

Mehedi, Sk. T., Anwar, A., Rahman, Z., Ahmed, K., & Islam, R. (2023). Dependable Intrusion Detection System for IoT: A Deep Transfer Learning Based Approach. *IEEE Transactions on Industrial Informatics*, *19*(1), 1006–1017. https://doi.org/10.1109/tii.2022.3164770

Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey Wolf Optimizer. *Advances in Engineering Software*, *69*, 46–61. https://doi.org/10.1016/j.advengsoft.2013.12.007

Mohammadpour, L., Ling, T. C., Liew, C. S., & Aryanfar, A. (2020). A Mean Convolutional Layer for Intrusion Detection System. *Security and Communication Networks*, *2020*, 1–16. https://doi.org/10.1155/2020/8891185

Moustafa, N., & Slay, J. (2015). UNSW-NB15: A Comprehensive Data Set for Network Intrusion Detection Systems (UNSW-NB15 Network Data Set). *2015 Military Communications and Information Systems Conference (MilCIS)*, *8*(2), 1–6. https://doi.org/10.1109/milcis.2015.7348942

Nguyen, S.-N., Nguyen, V.-Q., Choi, J., & Kim, K. (2018). Design and implementation of intrusion detection system using convolutional neural network for DoS detection. *Proceedings of the 2nd International Conference on Machine Learning and Soft Computing*, 34–38. https://doi.org/10.1145/3184066.3184089

Prasad, A., & Chandra, S. (2024). BotDefender: A Collaborative Defense Framework Against Botnet Attacks using Network Traffic Analysis and Machine Learning. *Arabian Journal for Science and Engineering*, *49*(3), 3313–3329. https://doi.org/10.1007/s13369-023-08016-z

Pajouh, H. H., Javidan, R., Khayami, R., Dehghantanha, A., & Choo, K.-K. R. (2019). A Two-Layer Dimension Reduction and Two-Tier Classification Model for Anomaly-Based Intrusion Detection in IoT Backbone Networks. *IEEE Transactions on Emerging Topics in Computing*, *7*(2), 314–323. https://doi.org/10.1109/tetc.2016.2633228

RM, S. P., Maddikunta, P. K. R., M., P., Koppu, S., Gadekallu, T. R., Chowdhary, C. L., & Alazab, M. (2020). An effective feature engineering for DNN using hybrid PCA-GWO for intrusion detection in IoMT architecture. *Computer Communications*, *160*, 139–149. https://doi.org/10.1016/j.comcom.2020.05.048

Saba, A., Liu, X. M., & Mansourvar, M. (2022). Application of meta-heuristic algorithms in intrusion detection systems. *International Journal of Network and Application Security. Advance Online Publication*, 2517–2540. https://doi.org/10.22075/ijnaa.2022.27338.3563

Safaldin, M., Otair, M., & Abualigah, L. (2021). Improved Binary Gray Wolf Optimizer and SVM for Intrusion Detection System in Wireless Sensor Networks. *Journal of Ambient Intelligence and Humanized Computing*, *12*(2), 1559–1576. https://doi.org/10.1007/s12652-020-02228-z

Sharma, B., Sharma, L., Lal, C., & Roy, S. (2023). Anomaly based network intrusion detection for IoT attacks using deep learning technique. *Computers and Electrical Engineering*, *107*, 108626. https://doi.org/10.1016/j.compeleceng.2023.108626

Sherstinsky, A. (2020). Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) Network. *Physica D: Nonlinear Phenomena*, *404*, 132306. https://doi.org/10.1016/j.physd.2019.132306

Tavallaee, M., Bagheri, E., Lu, W., & Ghorbani, A. A. (2009). A Detailed Analysis of the KDD CUP 99 Data set. *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*, 1–6. https://doi.org/10.1109/cisda.2009.5356528

Thakkar, A., & Lohiya, R. (2022). A survey on intrusion detection system: feature selection, model, performance measures, application perspective, challenges, and future research directions. *Artificial Intelligence Review*, *55*(1), 453–563. https://doi.org/10.1007/s10462-021-10037-9

Thamilarasu, G., & Chawla, S. (2019). Towards Deep-Learning-Driven Intrusion Detection for the Internet of Things. *Sensors*, *19*(9), 1977. https://doi.org/10.3390/s19091977

Tyagi, H., & Kumar, R. (2021). Attack and Anomaly Detection in IoT Networks Using Supervised Machine Learning Approaches. *Revue d'Intelligence Artificielle*, *35*(1), 11–21. https://doi.org/10.18280/ria.350102

Verma, A., & Ranga, V. (2020). Machine Learning Based Intrusion Detection Systems for IoT Applications. *Wireless Personal Communications*, *111*(4), 2287–2310. https://doi.org/10.1007/s11277-019-06986-8

Ye, J., Cheng, X., Zhu, J., Feng, L., & Song, L. (2018). A DDoS Attack Detection Method Based on SVM in Software Defined Network. *Security and Communication Networks*, *2018*, 1–8. https://doi.org/10.1155/2018/9804061

Yihunie, F., Abdelfattah, E., & Regmi, A. (2019). Applying Machine Learning to Anomaly-Based Intrusion Detection Systems. *2019 IEEE Long Island Systems, Applications and Technology Conference (LISAT)*, 1–5. https://doi.org/10.1109/lisat.2019.8817340

Zarai, R., Kachout, M., Hazber, M. A. G., & Mahdi, M. A. (2020). Recurrent Neural Networks and Deep Neural Networks Based on Intrusion Detection System. *OALib*, *07*(03), 1–11. https://doi.org/10.4236/oalib.1106151