Journal of Intelligent Systems and Internet of Things
Vol. 17, No. 02, PP. 369-391, 2025
DOI: https://doi.org/10.54216/JISIoT.170224

Enhanced Anomaly Detection in IoT Networks Using Hybrid Deep Learning and Bio-Inspired Optimization

M. Sindhuja[1,*], Noorfazila Kamal[2], Kalaivani Chellappan[3]

[1]Department of Computing Technologies, School of Computing, College of Engineering and Technology, SRM Institute of Science and Technology, Kattankulathur, Chengalpattu District - 603 203, Tamil Nadu, India

[2]Department of Electrical, Electronics & System Engineering, Faculty of Engineering & Built Environment, Universiti Kebangsaan Malaysia, Bangi, Malaysia

[3]Integrated Systems Engineering and Advanced Technologies (INTEGRA), Faculty of Engineering & Built Environment, Universiti Kebangsaan Malaysia, Bangi, Malaysia

Emails: sindhumano12@gmail.com; fazila@ukm.edu.my; kckalai@ukm.edu.my

Abstract

The rapid expansion of Internet of Things (IoT) devices has significantly amplified cybersecurity risks, thereby necessitating advanced anomaly detection mechanisms. This research introduces a hybrid detection framework tailored for IoT networks, combining deep learning architectures with bio-inspired optimization techniques. At the core of the framework lies the IoT Autoencoder-Based Feature Extraction Network (IoTAE-FEN), designed to minimize data dimensionality while preserving key discriminative features. To further refine the selected attributes, a Binary Multi-Objective Enhanced Gray Wolf Optimization (BMOEGWO) strategy, modeled on the cooperative hunting behavior of gray wolves, is employed. For the classification phase, Random Forest (RF) is integrated, resulting in the proposed AE-BMOEGWO-RF hybrid model. The effectiveness of this approach was validated on benchmark datasets, including NSL-KDD and TON-IoT. Experimental findings highlight a feature selection accuracy of 96.85% on the TON-IoT dataset and an overall classification performance of 97.81% on NSL-KDD. Comparative evaluations against existing techniques underscore the framework's superior detection capability, emphasizing its potential to strengthen IoT network security by addressing longstanding challenges in feature extraction and selection for anomaly detection.

Received: January 01, 2025 Revised: March 02, 2025 Accepted: June 06, 2025

Keywords: Anomaly Detection; Autoencoder; Feature Extraction; Hybrid Model; Optimization; Internet of Things; Nature-Inspired Computing; Cybersecurity; Gray Wolf Optimization

1. Introduction

The Internet of Things (IoT) has emerged as one of the most rapidly evolving technologies, enabling seamless connectivity among both homogeneous and heterogeneous entities over the Internet. While this interconnectedness fosters innovation, it also introduces substantial security challenges. IoT devices are particularly susceptible to cyber threats due to their constrained computational resources, the rising number of deployed devices, and the diversity of communication protocols. Security weaknesses in many IoT platforms make them prime targets for attackers to exploit vulnerabilities, form botnets, and propagate malware. Conventional intrusion detection mechanisms often prove inadequate in IoT environments, primarily because of limited device capacity and the ecosystem's complexity. Since most smart IoT platforms operate through Wi-Fi networks, safeguarding wireless communications has become a priority. However, many IoT devices lack the built-in hardware security needed to detect or mitigate sophisticated cyberattacks, leaving them defenseless against evolving threats. The heterogeneity of devices, reliance on various communication

standards, insufficient security architectures, and inherent limitations in processing, memory, and energy consumption all exacerbate IoT security issues. These challenges underline the necessity for tailored IoT-centric security solutions. Communication within IoT networks heavily depends on routing protocols, which enable efficient data exchange among connected devices. A widely used example is the Routing Protocol for Low-Power and Lossy Networks (RPL), built on distance-vector principles and optimized for IPv6-based networks characterized by limited power and unreliable links. Despite its advantages, RPL still faces critical vulnerabilities, making it prone to multiple forms of cyberattacks, such as Denial-of-Service (DoS), largely due to the restricted resources of IoT nodes, including processing capabilities, storage, and energy reserves. An exemplary instance of a Sybil attack is the exploitation of flaws in the transmission method of RPL's DODAG Information Object (DIO) [5]. Various countermeasures are utilized alongside intrusion detection systems to reduce the impact of cyber threats in IoT systems. Monitor-based techniques utilize the monitoring of network traffic to identify possible routing problems and abnormal communication activity [6]. Cryptography-based solutions employ encryption techniques such as TESLA to provide communication channels that are secure. Inducement-based techniques have the objective of tricking hostile nodes into disclosing their routing violations. The progress made in intrusion and anomaly detection systems utilizing machine learning and deep learning has been substantial [7]. Although traditional machine learning techniques such as decision trees and support vector machines are successful, deep learning approaches have demonstrated improved performance, especially when dealing with huge datasets that have high-dimensional characteristics. Utilizing metaheuristic algorithms and hybrid optimization approaches, such as GWO and PSO, in conjunction with random forest algorithms, improves the ability to detect anomalies in IoT contexts. Due to the large amount of data produced by various systems in IoT ecosystems, deep learning techniques are ideal for creating strong intrusion detection systems. Deep learning is a complex subdivision of machine learning that employs multiple layers of neurons to facilitate elaborate learning processes. Deep learning, unlike conventional machine learning techniques, possesses the adaptability to optimize its performance in order to effectively process extensive datasets. The methodologies have demonstrated efficacy in applications like as dimensionality reduction, feature extraction, and data classification. Deep learning models in intrusion and anomaly employ categorize data behavior into two distinct groups from historical traffic data that are normal and attack [8].

Autoencoders have become an essential advancement in deep learning models for anomaly detection within the Internet of Things (IoT). Their effectiveness lies in the ability to autonomously learn and extract both low-level and high-level features from diverse data sources while performing computations efficiently. This capability makes autoencoders particularly well suited for anomaly detection tasks, as they can capture and differentiate critical features with minimal manual intervention and deliver results with considerable computational efficiency. Despite recent progress, anomaly detection in Internet of Things (IoT) environments using deep learning and machine learning still faces several challenges. Key issues include the difficulty of selecting essential features to identify malicious attacks, generating meaningful new features from raw data, and reducing the rate of false negatives. To address these limitations, we propose a hybrid Autoencoder-based framework, termed IoTAE-FEN, designed to enhance feature extraction for anomaly detection in IoT networks. For feature selection, we introduce a binary extension of the Multi-Objective Enhanced Gray Wolf Optimization algorithm (BMOEGWO), which efficiently manages the selection process. By combining the IoTAE-FEN feature extractor with BMOEGWO and Random Forest (RF) for classification, we developed the AE-BMOEGWO-RF model, a novel solution that systematically overcomes the identified challenges and improves detection performance.

This research introduces several novel strategies aimed at enhancing anomaly detection within Internet of Things (IoT) environments. The central focus is the design of a specialized autoencoder tailored for feature extraction, enabling higher accuracy in detecting abnormal behavior. The proposed autoencoder is capable of learning and representing both low-level and high-level features from IoT data, thereby improving the detection of irregularities. Furthermore, we explore hybrid feature extraction techniques that integrate information across multiple levels to achieve a more comprehensive understanding of IoT data patterns. To strengthen the detection framework, a hybrid model that incorporates Random Forest (RF) with the autoencoder is proposed, leveraging the complementary strengths of both approaches to improve precision and classification performance. Additionally, we enhance the Grey Wolf Optimization (GWO) algorithm, adapting it for more effective anomaly detection in IoT data streams. To

resolve the challenge of optimal feature selection, we present a binary multi-objective enhanced Grey Wolf Optimization (BMOEGWO) approach, which evaluates multiple selection criteria and ensures more accurate identification of anomalies. Collectively, these contributions provide a robust and efficient detection framework, offering stronger and more adaptive security defenses against emerging IoT-based cyberattacks. The structure of this paper is as follows. Section II provides an in-depth survey of prior research on intrusion detection and feature selection techniques. Section III outlines the fundamental concepts of the Grey Wolf Optimizer (GWO). Section IV describes the proposed AE-BMOEGWO-RF framework for anomaly detection in IoT networks. Section V presents the experimental evaluation of the model and compares its effectiveness with existing deep learning and machine learning methods. Finally, Section VI offers concluding remarks and highlights prospective directions for future work.

The Binary Multi-Objective Enhanced Gray Wolf Optimization (BMOEGWO) offers significant improvements over the standard GWO and other multi-objective optimizers. Traditional GWO often struggles with high-dimensional data, showing weak exploration and premature convergence. To address this, BMOEGWO employs a binary representation for handling discrete IoT feature selection, integrates Lévy flight to expand the search space and escape local optima, and applies genetic operators such as crossover and mutation to increase solution diversity. An elite archive with roulette wheel sampling further enhances convergence stability and balances exploration with exploitation. By combining the autoencoder for feature extraction, BMOEGWO for feature selection, and Random Forest for classification, the proposed hybrid model effectively manages high-dimensional IoT data, reduces redundancy, and achieves accurate real-time anomaly detection with strong computational efficiency.

2. Literature Survey

This section reviews existing work on anomaly detection and feature selection employing machine learning and deep learning techniques. Early approaches introduced heuristic-based solutions to mitigate suppression attacks in low power and lossy IoT networks, while Mutual Authentication and Detection (MAD) was applied to counter energy depletion attacks. Other studies developed methods based on Theil's index to defend against vampire attacks targeting the routing layer. Hybrid Intrusion Detection Systems (IDS), combining classifiers such as XGBoost, KNN, and Gaussian Naive Bayes in the initial stage and Random Forest for final classification, demonstrated improved detection performance on benchmark datasets. More recently, frameworks that integrate fog computing with federated learning have shown promise in anomaly detection while preserving user privacy in IoT environments. Advances in feature selection techniques have also led to notable improvements in detection accuracy across diverse datasets. Collectively, these contributions highlight the role of machine learning in strengthening IoT security, particularly in intrusion detection tasks where the ability to generalize beyond known attack patterns is crucial. Earlier studies additionally explored association rule-based techniques for intrusion detection. One limitation of rule-based approaches is the generation of an excessive number of rules, which increases system complexity. Recent research has therefore emphasized the use of machine learning techniques and learning from examples to develop classifiers that are more efficient. Neural networks have been widely applied in detecting and categorizing abnormal patterns, but training them on large-scale datasets such as KDDCup 1999 is computationally expensive and time-consuming. Decision trees and Naïve Bayes remain popular alternatives for intrusion detection, with decision trees preferred for their simplicity, adaptability, and strong classification accuracy. Conversely, Naïve Bayes assumes conditional independence among attributes, which can negatively influence its performance. In addition, support vector machines (SVM) with advanced learning methods have been proposed to improve detection efficiency. Despite these advancements, many intrusion detection systems still lack detailed accuracy assessments across individual attack classes. To address this limitation, researchers have explored ensemble and hybrid models, such as combining decision trees with SVM, to achieve more comprehensive and reliable detection across multiple categories.

Several studies have explored the application of genetic algorithms for intrusion detection. These algorithms have been employed on training datasets to classify labels influenced by smurf attacks, achieving a notably low false positive rate of 0.2%. Additional research has focused on generating intrusion detection rules through genetic algorithm-based classification, while others have applied genetic algorithms combined with fitness functions to optimize rule estimation. Alongside these approaches, machine-

learning techniques such as Artificial Neural Networks (ANN) have also been widely adopted for intrusion detection tasks. Earlier research has investigated the use of machine learning techniques for intrusion detection, including Artificial Neural Networks (ANN) and hybrid approaches that integrate fuzzy clustering with ANN to mitigate challenges of poor stability and low detection accuracy. In this approach, fuzzy clustering is applied to generate smaller, well-defined subsets of data, thereby reducing complexity. Each subset is then trained with different ANN models, leading to promising results. Another study introduced a backpropagation-based model for intrusion detection, where training was performed using input–target pairs to structure the network. However, despite achieving close to 80% detection accuracy across various attack types, this method showed limited capability in identifying hidden or sophisticated intrusions. Similarly, the Multi-Layer Perceptron (MLP) has been applied to classify attacks into six categories, but its performance was found unsatisfactory due to the generation of irrelevant outputs. To overcome these shortcomings, the present study aims to develop a more robust and efficient detection framework.

Random Forest is a widely used ensemble learning technique for classification, known for enhancing the traditional bagging approach by introducing feature randomness. This process generates a group of independent decision trees whose outputs are combined to produce the final prediction. Each tree in the ensemble is constructed using a bootstrap sample from the training dataset. Before training, three key hyperparameters must be specified: the minimum node size, the number of features sampled at each split, and the total number of trees in the forest. The appeal of Random Forest lies in its simplicity and flexibility, allowing it to handle both classification and regression problems effectively. Unlike standalone decision trees that evaluate all possible feature splits, Random Forest selects only a random subset of attributes. The prediction mechanism varies by task: regression problems use the average of tree outputs, while classification relies on majority voting for the predicted class. In addition, this study incorporates v-shape and s-shape transfer functions for binarization, along with multiple feature selection strategies, to enhance the performance of the whale optimization algorithm. The experimentation conducted on the N-BaIOT dataset demonstrates that the whale optimization technique, when combined with the v-shape transfer function and elitist tournament binarization approach, outperforms the s-shape transfer function. Additionally, a novel dataset for the Internet of Things (IoT) called TON_IoT is presented and evaluated using several machine learning algorithms [23, 27]. The accuracy rates obtained using Deep Neural Networks (DNN) and Gradient Boosting Machine (GBM) approaches are almost the same for this dataset. In addition, Su et al. conducted a study on the process of selecting relevant features and predicting Internet threats using machine-learning techniques [24]. The feature selection process utilizes Gradient Boosting Machine (GBM), Random Forest (RF), and Decision Tree (DT) algorithms to determine the relevance scores of each feature [25-29]. The analysis of the IoT2020 dataset demonstrates that decision trees, random forest algorithms, and gradient-boosting machines exhibit comparable levels of accuracy, with the random forest approach surpassing the others in terms of the AUC criterion. Literature on new intrusion and anomaly detection methods in the IoT was reviewed [26, 30]. Our review shows that machine learning and deep learning methods have received more attention in recent years. In these methods, the speed or time to detect attacks is very important. Usually, in intrusion detection systems, there are big data and high dimensions, which lead to the creation of models with complex calculations. Therefore, reducing or selecting effective features and extracting new features from the data set can play an effective role in increasing the speed and accuracy of intrusion detection systems. Optimization algorithms such as the GWO can effectively remove redundant features and select the main features of a dataset. Also, using deep learning algorithms helps extract new features. This paper presents a multi-objective enhanced GWO for feature selection to solve these problems and challenges. In addition, a convolution neural network with hybrid layers called IoTAE-FEN has been designed and implemented to extract features to better detect anomalies in the IoT. Thus, the table 1 illustrate the comparison of ML and DL techniques for anomaly detection and feature selection in IoT.

Table 1: Comparison of Machine Learning and Deep Learning Techniques for Anomaly Detection and Feature Selection in IoT

Study	Approach	Findings	Advantages	Disadvantages
Heuristic-based method for IoT	Suppression assaults detection in low-power and lossy IoT networks	Effective in detecting and preventing energy reduction attacks	Specific to low-power IoT networks, MAD used for security	Limited to specific network types, may not be generalizable
XGBoost, KNN, Gaussian Naive Bayes, Random Forest	IDS improvement through staged model application	Noticeable enhancement in performance on specific datasets	Improved performance by combining multiple models	Requires complex model integration, may increase computational cost
Fog computing & Federated learning	Anomaly detection & privacy safeguarding in IoT	Encouraging outcomes in anomaly detection and privacy protection	Effective in decentralized environments, enhances privacy	Complexity in implementation, requires strong infrastructure
Feature selection improvements	Enhanced techniques for feature selection	Higher accuracy on various datasets	Improved accuracy by selecting relevant features	Potentially high computational demand during selection
Association rules for intrusion detection	Early method using rule-based systems	Identifies many rules, but increases system complexity	Simple and interpretable	Generates large number of rules, system becomes complex
Neural Networks	Detection and classification of anomalies	Effective but time-consuming on large datasets (e.g., KDDCup 1999)	High accuracy in pattern recognition	Slow training on large datasets, resource-intensive
Decision Trees & Naive Bayes	Common methods in IDS	Decision trees are fast and accurate, Naive Bayes assumes conditional independence	Simple, fast, adaptable	Naive Bayes may underperform due to conditional independence assumption

SVM learning techniques	Advanced IDS classification	Improves accuracy across different categories	Effective in high-dimensional spaces	May not provide detailed accuracy insights for individual classes
Genetic Algorithms	Intrusion detection through evolutionary computation	Low false positive ratio in smurf attack detection	Optimizes rules, low false positives	May require extensive computational resources
Artificial Neural Networks (ANN)	Combination with fuzzy clustering for intrusion detection	Reduces subset size and complexity, but low precision detection	Reduces complexity, produces noteworthy outcomes	Issues with stability and precision, limited effectiveness on large datasets
Random Forest	Ensemble learning for classification tasks	Improved prediction through multiple decision trees	High adaptability, effective in various tasks	Requires careful hyper-parameter tuning, resource-intensive
Whale Optimization Algorithm (WOA)	Optimization in IDS using binarization and transfer functions	Outperforms other methods on N-BaIOT dataset	Effective in feature selection and optimization	Complex implementation, requires specific conditions for success
Deep Neural Networks (DNN) & Gradient Boosting Machine (GBM)	Applied to IoT datasets (TON_IoT, IoT2020)	Comparable accuracy rates, with RF outperforming in AUC	High accuracy, robust against overfitting	High computational cost, requires large training datasets
Multi-objective GWO & IoTAE-FEN	Feature selection & anomaly detection in IoT	Enhanced speed and accuracy in IDS	Effective in reducing redundant features, improves detection	Complexity in implementation, requires expertise in deep learning

2.1. Gray Wolf Optimization

The Gray Wolf Optimization (GWO) method, developed by Mirjalili et al., is a metaheuristic optimization algorithm that draws inspiration from the social hierarchy and hunting patterns of gray wolves. GWO has demonstrated impressive efficacy in addressing optimization challenges across several fields. Below is a comprehensive outline of the sequential procedures involved in GWO:

Step 1. Initialization: Initialize the position of N wolves in the search space: Xi, where i=1, 2….N. Each wolf Xi represents a potential solution to the optimization problem.

Step 2. Fitness Evaluation: Evaluate the fitness of each wolf based on the objective function.

$$Fitness\ \ (X_i) = f(X_i) \dots \dots \dots \dots (1)$$

Here, f (Xi) represents the objective function value at position Xi.

Step 3. Updating the Positions of Alpha, Beta, and Delta Wolves: Identify the alpha, beta, and delta wolves based on their fitness values & update the positions of these wolves using specific formulas:

For the alpha wolf:

$$X_{alpha} = X_i - \alpha * Distance\ \ (X_i, X_{alpha}) \dots \dots \dots . (2)$$

For the beta wolf:

$$X_{beta} = X_i - \beta * Distance\ \ (X_i, X_{beta}) \dots \dots \dots . (3)$$

For the delta wolf:

$$X_{delta} = X_i - \delta * Distance\ \ (X_i, X_{delta}) \dots \dots \dots . (4)$$

Here, α, β, and δ are scaling factors, and distance (X_i,X_j) calculates the distance between wolves Xi and Xj.

Flowchart of the Gray Wolf Optimization (GWO) Algorithm shown by Figure 1.

Figure 1. Flowchart of the Gray Wolf Optimization (GWO) Algorithm

3. Methodology

This section presents a new anomaly detection framework specifically designed for IoT environments, integrating an autoencoder with an improved Grey Wolf Optimizer (GWO). The proposed autoencoder, referred to as the IoT Autoencoder-Based Feature Extraction Network (IoTAE-FEN), is developed to capture both local and global features essential for detecting anomalies in IoT data. For feature selection, a Binary Multi-Objective Enhanced GWO (BMOEGWO) is introduced to effectively handle high-dimensional attributes. The combination of IoTAE-FEN for feature extraction and BMOEGWO for feature selection leads to the

hybrid AE-BMOEGWO-RF model, which improves detection accuracy. The overall architecture of the system, divided into four stages, is illustrated in Figure 2.

The effectiveness of the AE-BMOEGWO-RF framework is assessed using the NSL-KDD and TON-IoT datasets. At the core of this model lies the BMOEGWO algorithm, which enhances detection speed and accuracy by selecting an optimal subset of features ranging from 20% to 50%. In parallel, the IoTAE-FEN module improves the quality and reliability of features by integrating those chosen by BMOEGWO with features derived from diverse sets. Figure 3 illustrates the overall workflow of the proposed IoT anomaly detection framework, which incorporates three main stages: feature extraction through IoTAE-FEN, feature selection using BMOEGWO, and anomaly classification via the Random Forest algorithm. A comprehensive description of each stage is provided in the following sections.

3.1. Structure Of the Proposed Method

The AE-BMOEGWO-RF framework is applied and validated using the NSL-KDD and TON-IoT datasets. Within this approach, the BMOEGWO algorithm plays a pivotal role by selecting the most relevant features for intrusion detection, thereby improving both detection speed and accuracy through the selection of 20%–50% of the feature set. Furthermore, the IoTAE-FEN module enhances feature quality and detection performance by extracting unique attributes and merging them with the features chosen by BMOEGWO. Figure 3 depicts the overall design of the proposed IoT anomaly detection framework, which integrates several key components: dataset handling, data pre-processing, feature extraction with IoTAE-FEN (covering both high- and low-level features), feature selection through BMOEGWO, and anomaly classification using the Random Forest model. A detailed explanation of each component is provided in the subsequent sections.

3.2. Dataset

3.2.1. NSL-KDD Dataset

The NSL-KDD dataset contains a total of 148,517 instances, which are divided into two subsets: KDDTrain+ and KDDTest+. The training portion, KDDTrain+, includes 125,973 instances, while the testing portion, KDDTest+, consists of 22,544 instances. In total, the dataset provides 41 features and incorporates both normal and attack-related records, as summarized in Table 2.

3.2.2. TON-IOT Dataset

In contrast, the TON-Iot dataset is a more recent benchmark designed for IoT and IIoT applications, encompassing both network traffic data and system logs. It contains a total of 22,339,021 records, incorporating samples of normal activity as well as attack scenarios. Among these, 300,000 entries represent benign traffic, whereas 161,043 correspond to malicious behaviours.

The arrangement of this data follows a defined sequence, which is illustrated in the referenced figure and is crucial for interpreting the dataset in alignment with the authors' perspective.

Table 2: NSL-KDD Dataset Overview

Dataset	Traffic Type	Training Samples	Testing Samples
NSL-KDD	Normal	67,343	9,711
	Attack	58,630	12,833

Table 3: TON-IoT Dataset Overview

Dataset	Traffic Type	Training Samples	Testing Samples
TON-IoT	Normal	3,00,000	796380
	Attack	1,61,043	2,15,42,641

3.3. Data Pre – Preprocessing

The preprocessing pipeline of the IoT Autoencoder-Based Feature Extraction Network (IoTAE-FEN) consists of several critical steps designed to prepare datasets such as NSL-KDD and TON-IoT for effective anomaly detection in IoT systems. The process begins with data cleansing, where irrelevant or noisy records are removed. Incomplete data entries are then handled by imputation, typically replacing missing numeric values with their mean and categorical values with their mode. Following this, categorical attributes are converted into numerical form through approaches like one-hot encoding or label encoding to ensure compatibility with the IoTAE-FEN framework. The resulting feature values are further standardized using Min-Max normalization, which scales them to the range, ensuring consistency across different attributes and datasets. Finally, feature selection is performed using the BMOEGWO algorithm, which identifies the most significant attributes required for improving intrusion detection performance.

The IoTAE-FEN framework is capable of extracting both low-level and high-level feature representations, which play a vital role in anomaly detection. The integration of the BMOEGWO algorithm further refines feature selection, thereby enhancing detection accuracy. The feature subsets derived from IoTAE-FEN and BMOEGWO are jointly utilized to train the anomaly detection model based on the Random Forest (RF) classifier. This combination enables reliable detection of abnormal behaviours within IoT networks. Through this preprocessing strategy, datasets are effectively transformed to serve as suitable inputs for IoTAE-FEN, facilitating precise feature extraction and accurate anomaly detection in IoT scenarios.

3.4. Autoencoder

The IoT Autoencoder-Based Feature Extraction Network (IoTAE-FEN) is a specialized neural architecture designed to derive critical features from IoT data. Its design integrates three key components: an encoder, a bottleneck layer, and a decoder, each contributing to the overall feature extraction process. The encoder forms the foundation of the network and is composed of one-dimensional convolutional layers (conv_i) paired with corresponding pooling layers (MaxPool1D_conv_i). Together, these layers capture intricate patterns within the data by identifying fine-grained details as well as broader trends. To enhance training stability and mitigate overfitting, dropout and batch normalization layers are incorporated, improving the model's ability to generalize to unseen data. The bottleneck layer serves as a critical compression channel, transforming the input into a compact latent representation that preserves the most essential attributes of the original data. This representation ensures efficient information encoding and retrieval. Following this stage, the decoder reconstructs the compressed features using one-dimensional deconvolutional layers (deconv_i) along with upsampling layers. These elements progressively recover the latent features into their original scale, with the upsampling layers playing a pivotal role by increasing the spatial resolution of feature maps to achieve accurate data reconstruction.

In the training process, the IoTAE-FEN integrates both supervised and unsupervised learning strategies. The supervised component guides the network to accurately reconstruct the input data by minimizing reconstruction errors. In contrast, the unsupervised component enables the model to extract meaningful representations from the data without relying on predefined labels, allowing it to autonomously identify the underlying patterns and distinctive features of the input.

The operation of IoTAE-FEN is governed by three fundamental mathematical formulations. The first is the encoder function, expressed as f=Encoder(x), which defines the transformation of raw input x into a set of extracted features. Another key element is the softmax function, introduced in equation (8), which further supports the classification aspect of the framework.

$$\sigma(f)i = e\hat{}(f\_i\ )/\ [\![sum\{j=1\}]\!]\ \hat{}c\ \ *e\hat{}(f\_j\ )\ \ldots\ldots\ldots(8)$$

is used to transform raw scores into probability distributions across many classes, making it useful for classification problems.

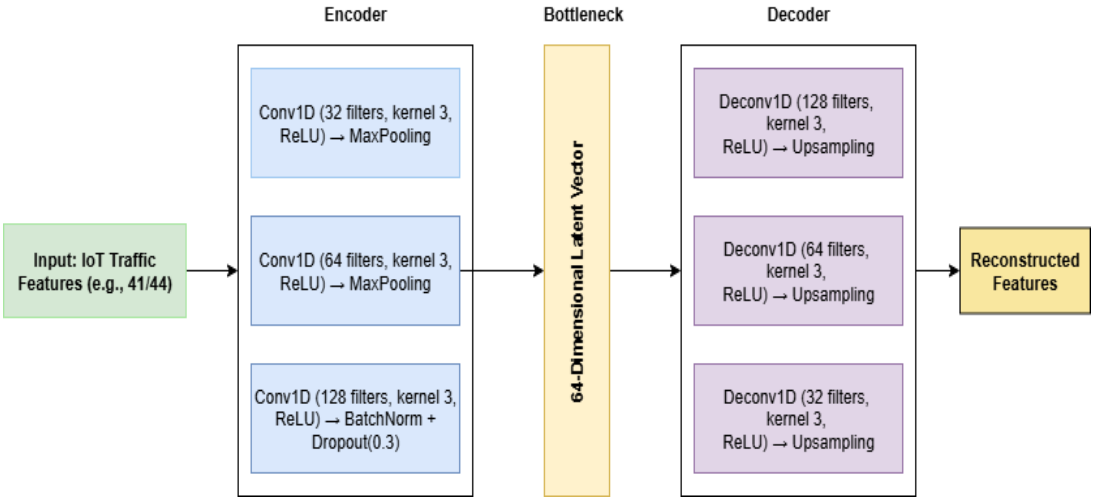

Figure 2. Anomaly Detection Framework for IoT Using Autoencoder Neural Network

In addition, the reconstruction loss—calculated as the mean squared error between the original input and its reconstructed output—serves as a key metric for evaluating the effectiveness of the network. Overall, the IoTAE-FEN represents a sophisticated framework for both feature extraction and anomaly detection in IoT environments. By leveraging autoencoders in conjunction with advanced neural architectures, it establishes a robust mechanism for accurately identifying irregularities and mitigating potential security threats.

Table 4: IoTAE-FEN Autoencoder Architecture

Layer Type	Filters	Kernel Size	Activation	Other Details
Conv1D-1	32	3	ReLU	MaxPool1D, Dropout 0.3
Conv1D-2	64	3	ReLU	MaxPool1D
Conv1D-3	128	3	ReLU	BatchNorm, Dropout 0.3
Bottleneck	64	–	Linear	Latent Representation
Deconv1D-1	128	3	ReLU	Upsampling
Deconv1D-2	64	3	ReLU	Upsampling
Deconv1D-3	32	3	ReLU	Upsampling
Output Layer	1	–	Sigmoid	Reconstructed Data

The encoder of IoTAE-FEN is composed of three one-dimensional convolutional layers with 32, 64, and 128 filters, each utilizing a kernel size of 3 and activated through ReLU. Every convolutional layer is coupled with a MaxPooling operation, which progressively reduces dimensionality. To improve generalization and prevent overfitting, batch normalization is applied along with a dropout rate of 0.3. The bottleneck layer compresses the feature representation into a 64-dimensional latent vector. The decoder mirrors the encoder's architecture, employing three one-dimensional deconvolutional layers with 128, 64, and 32 filters and a kernel size of 3, followed by upsampling layers that reconstruct the original signal. Finally, a sigmoid activation function is applied in the output layer to generate normalized results.

3.5. Binary Multi-Objective Enhanced GWO

To implement the improvements of Binary Multi-Objective Enhanced Gray Wolf Optimization (BMOEGWO), follow these procedures. An archive is kept to store the non-dominated solutions, also known as Pareto optimum solutions that are found during the optimization process. Subsequently, the answers are encoded using binary representation, and genetic operators such as crossover and mutation are employed to provide novel binary solutions and improve diversity. In order to enhance performance and expand the range of investigation, the incorporation of Levy flying is utilized to generate random disturbances in the placements of the wolves. This enhances the algorithm's ability to explore. In addition, the search process is guided by selecting elite persons (the top solutions) from the archive. This ensures that the direction of the search is influenced by the best solutions. Grading is now employed to assess and order answers in the archive, while roulette wheel selection is utilized to probabilistically choose solutions from the archive for the subsequent generation. This approach encourages both diversity and quality. The steps of the BMOEGWO algorithm are explained systematically below. The pseudo-code of the binary multi-objective enhanced GWO is reflected in Algorithm 1.

Algorithm 1: Binary Multi-objective Enhanced Grey Wolf Optimization (BMEGWO)

Inputs:

Population size (N)

Maximum number of iterations (MaxIter)

Lower bound (lb) for each dimension of the search space

Upper bound (ub) for each dimension of the search space

Objective functions (Obj1, Obj2, ..., ObjM)

Step 1: Initialize population of wolves X_i (i=1, 2, …, n) randomly in binary space.

Step 2: Initialize archive A to store Pareto optimal solutions.

Step 3: Define maximum iterations / termination criteria.

Step 4: Evaluate fitness of each wolf based on objectives.

Step 5: Identify Alpha, Beta, and Delta wolves.

Step 6: While (termination criteria not met) do

Step 7: For each wolf X_i do

Step 8: Update position using GWO rules.

Step 9: Apply sigmoid transfer function and convert to binary.

Step 10: Apply genetic operators (crossover and mutation).

Step 11: Select parent solutions based on fitness.

Step 12: Apply crossover to generate offspring.

Step 13: Apply mutation to offspring.

Step 14: Introduce Levy flight for exploration.

Step 15: Evaluate fitness of the new solutions.

Step 16: Update archive A with non-dominated solutions.

Step 17: End for

Step 18: Update Alpha, Beta, Delta wolves from archive A.

Step 19: Perform grading and roulette wheel selection on archive A.

Step 20: Select next generation of wolves from archive A.

Step 21: End while

Step 22: Return archive a containing Pareto optimal solutions.

3.6. Use of Archives to Store the Pareto Solutions

The fundamental objective of the archive is to preserve a collection of non-dominated solutions that represent the Pareto front. This ensures the preservation of high-quality solutions, provides guidance for future searches, and upholds diversity. At first, the archive contains no data. Upon assessing the existing population's fitness, the algorithm identifies solutions that are not dominated by any other solutions. The solutions are compared to the current archive members. Dominated solutions are eliminated, while non-dominated solutions are included. In order to avoid unlimited expansion, there is a restriction on the size of the archive. If the limit is surpassed, techniques such as clustering (grouping and eliminating similar solutions) or crowding distance (eliminating solutions in densely populated areas) are utilized to preserve diversity.

3.7. Use of Binary and Genetic Operators to Provide the Binary Version

Binary Multi-Objective Enhanced Gray Wolf Optimization (BMOEGWO) relies on the utilization of binary and genetic operators to preserve the binary nature of solutions and enhance diversity. This approach enhances the algorithm's capacity to explore the solution space, as demonstrated in Algorithm 2.

Algorithm 2:

Step 1: Initialize population of wolves Xi (i = 1, 2, ..., n) randomly in binary space

Step 2: Initialize archive A to store Pareto optimal solutions

Step 3: Define max_iterations

Step 4: Evaluate fitness of each wolf based on objectives

Step 5: Identify Alpha, Beta, and Delta wolves

Step 6: while (termination criteria not met) do

Step 7: for each wolf Xi do

Step 8: Update position using GWO rules

Step 9: Apply sigmoid function and convert to binary

Step 10: Apply genetic operators (crossover and mutation):

Step 11: Select parent solutions based on fitness

Step 12: Apply crossover to generate offspring

Step 13: Apply mutation to offspring

Step 14: Introduce Levy flight for exploration

Step 15: Evaluate fitness of the new solutions

Step 16: Update archive A with non-dominated solutions

Step 17: end for

Step 18: Update Alpha, Beta, Delta wolves from archive A

Step 19: Perform grading and roulette wheel selection on archive A

Step 20: Select next generation of wolves from archive A

Step 21: end while

Step 22: Return archive A containing Pareto optimal solutions

3.8. Using Levy Flight to Improve Performance and Increase the Exploration Space

The Levy flight incorporates random disturbances into the wolves' placements, enabling them to investigate novel and potentially uncharted regions of the search area. This enhances the algorithm's ability to search globally and prevents it from becoming stuck in local optima. To integrate Levy flying into the Binary Multi-Objective Enhanced Gray Wolf Optimization (BMOEGWO), adhere to the following instructions: After implementing the normal Grey Wolf Optimization (GWO) procedures to update the positions of the wolves, proceed to apply Levy flight to the updated positions. Utilize a Levy distribution to ascertain the magnitude and orientation for the perturbation's step. This combination improves the algorithm's capacity to efficiently search the solution space, which is shown in Algorithm 3.

Algorithm 3:

Step 1: Initialize population of wolves Xi (i = 1, 2... n) randomly in binary space

Step 2: Initialize archive A to store Pareto optimal solutions

Step 3: Define max_iterations

Step 4: Evaluate fitness of each wolf based on objectives

Step 5: Identify Alpha, Beta, and Delta wolves

Step 6: while (termination criteria not met) do

Step 7: for each wolf Xi do

Step 8: Update position using GWO rules

Step 9: Apply sigmoid function and convert to binary

Step 10: Apply Levy flight:

Step 11: Generate step size using Levy distribution

Step 12: Update wolf's position with the Levy step

Step 13: Convert the new position to binary

Step 14: Evaluate fitness of the new solutions

Step 15: Update archive A with non-dominated solutions

Step 16: end for

Step 17: Update Alpha, Beta, Delta wolves from archive A

Step 18: Perform grading and roulette wheel selection on archive A

Step 19: Select next generation of wolves from archive A

Step 20: end while

Step 21: Return archive a containing Pareto optimal solutions

3.9. Using the Elite Population Inside the Archive for Multi-Objective Mode and Increasing the Efficiency

The utilization of the elite population included inside the archive in Binary Multi-Objective Enhanced Gray Wolf Optimization (BMOEGWO) results in an increase in efficiency caused by the selection of the most effective solutions to direct the search process. This makes certain that the most effective solutions have an impact on the path that the search takes, hence enhancing the overall optimization process. This is accomplished by utilizing humans of high quality to steer the algorithm into more favourable portions of the solution space.

3.10. Grading and Roulette Wheel Selection

The algorithm for Binary Multi-Objective Enhanced Gray Wolf Optimization (BMOEGWO) with Grading and Roulette Wheel Selection proceeds as follows: Initially, a population of wolves Xi (i = 1, 2... n) is generated randomly in binary space, and an archive A is initialized to store Pareto optimal solutions. The maximum number of iterations is defined. Each wolf's fitness is evaluated based on multiple objectives, and the Alpha, Beta, and Delta wolves are identified. While the termination criteria are not met, the position of each wolf is updated using GWO rules, followed by applying a sigmoid function to convert the positions to binary. The fitness of the new solutions is evaluated, and the archive A is updated with non-dominated solutions. Grading is then performed on the archive, where solutions are ranked using non-dominated sorting and assigned scores based on their ranks. Roulette wheel selection is applied by assigning selection probabilities based on these scores, and solutions are selected probabilistically for the next generation. The Alpha, Beta, and Delta wolves are updated from these selected solutions. This process continues until the termination criteria are met, and the archive a containing Pareto optimal solution is returned.

Where the basic GWO uses only linear encircling and hunting mechanisms, several enhancements introduced in BMOEGWO significantly boost its performance. The binary encoding helps map the algorithm to a discrete feature selection problem for IoT anomaly detection. The Levy flight mechanism introduces stochastic jumps allowing the optimizer to break from its current local minimum and consequently search within a much wider solution space. Genetic operators further perform diversity improvement of the population hence avoiding premature convergence. Elite population guidance together with roulette wheel selection ensures high-quality solutions dominate while maintaining diversity at the same time. All these enhancements provide faster convergence, less redundant features, and better Pareto front diversity than classical GWO, NSGA-II, MOGWO, and MOPSO as will be seen later in experimental results in Section 4.

3.11. A Model Analysis

A combination of IoTAE-FEN for deep feature extraction, BMOEGWO for feature selection, and Random Forest (RF) for final classification is used in this hybrid technique, which is called AE-BMOEGWO-RF. The combined AE-BMOEGWO-RF model

architecture is shown in figure 3. The autoencoder compresses high-dimensional IoT data into a lower-dimensional representation, denoted as z, using an encoder function $f$enc , and then reconstructs the input data from the compressed representation using a decoder function $f$dec. This is expressed mathematically as equation (9 & 10),

$$z = f_{enc}(x) \dots \dots \dots \quad (9)$$

$$\hat{z} = f_{dec}(z) \dots \dots \dots \quad (10)$$

Where, x is the input IoT data. BMOEGWO optimizes the selection of the most discriminative features from the compressed representation z by encoding them in binary form. The optimization process involves maximizing classification accuracy Acc while minimizing the number of selected features Nf and can be formulated as a multi-objective optimization problem:

Maximize Acc, Minimize Nf subject to constraints on the selected features. The optimization is guided by Levy flight exploration and elite guidance, which enhance the algorithm's ability to efficiently search for optimal or near-optimal feature subsets. Theoretical analysis can evaluate the reduction in dimensionality achieved by the autoencoder, denoted as Dr, and the subsequent improvement in computational efficiency and model performance.

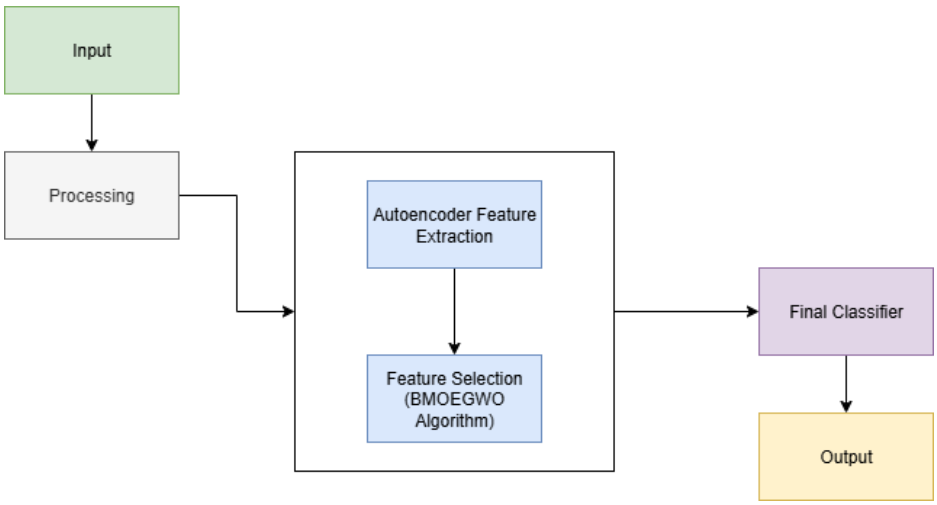

Figure 3. AE-BMOEGWO-Rf Model

Additionally, sensitivity analysis can assess the impact of hyperparameters and design choices on the performance and efficiency of the proposed approach. However, empirical validation and real-world experimentation are essential to validate the theoretical findings and assess the practical utility of the proposed approach in diverse IoT environments.

4. Result and Discussion

4.1. Evaluation of the NSL-KDD Dataset

Table 5 presents a comparative analysis of the AE-BMOEGWO-RF technique against alternative approaches on the NSL-KDD dataset. The Decision Tree (DT) algorithm achieved an accuracy of 88.10%, with precision, recall, and F1-score values of 88.04%, 88.17%, and 88.09%, respectively. The K-Nearest Neighbors (KNN) algorithm showed better performance with an accuracy of 91.54%, precision of 91.51%, recall of 89.50%, and an F1-score of 89.51%. The Multi-Layer Perceptron (MLP) algorithm attained an accuracy of 93.48%, precision of 92.47%, recall of 91.49%, and an F1-score of 89.48%. The Naive Bayes (NB) algorithm reported an accuracy of 89.81%, precision of 90.43%, recall of 89.39%, and an F1-score of 89.65%. Random Forest (RF) performed notably well, with an accuracy of 94.60%, precision of 95.62%, recall of 93.58%, and an F1-score of 94.60%. The IoT Autoencoder-Feature Extraction Network (IoTAE-FEN) approach achieved a higher accuracy of 95.65%, precision of 96.66%, recall of 95.65%, and an F1-score of

96.65%. The proposed AE-BMOEGWO-RF technique outperformed all the compared methods, with the highest accuracy of 96.85%, precision of 96.82%, recall of 96.81%, and an F1-score of 96.84%. In addition, figure 4 represent the confusion matrix for NSL-KDD dataset with AE-BMOEGWO-RF technique. Figure 5 shows the performance evaluation AE-BMOEGWO-RF technique with alternative approaches on the NSL-KDD dataset. Figure 6 shows the Comparison of AE-BMOEGWO-RF method with other methods on the NSL-KDD dataset regarding FPR and FNR criteria.

Table 5: Comparative Analysis of the AE-BMOEGWO-RF Technique with Alternative Approaches on the NSL-KDD

Algorithm	Accuracy	Precision	Recall	F1-Score
DT	88.1	88.04	88.17	88.09
KNN	91.54	91.51	89.5	89.51
MLP	93.48	92.47	91.49	89.48
NB	89.81	90.43	89.39	89.65
RF	94.6	95.62	93.58	94.6
IoTAE-FEN	95.65	96.66	95.65	96.65
AE-BMOEGWO-RF	96.85	96.82	96.81	96.84

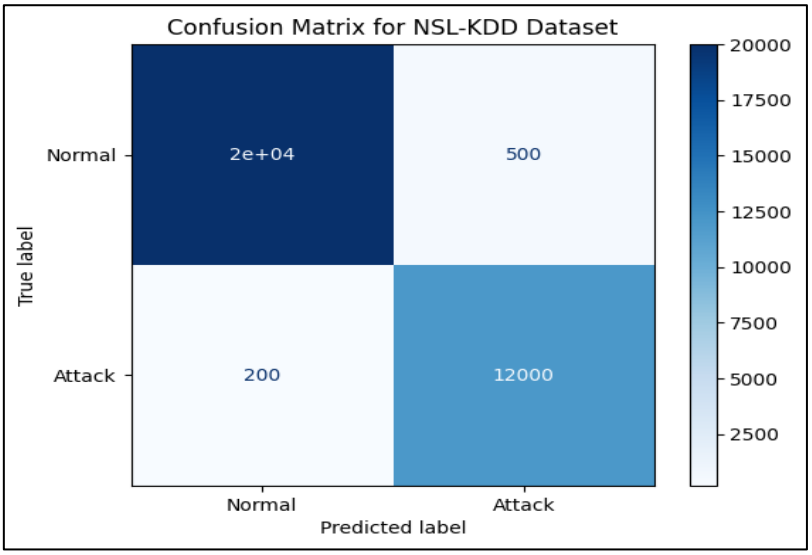

Figure 4. Confusion Matrix of NSL-KDD Dataset.

4.2. Evaluation of The Ton-IOT Dataset

Table 6 provides a comparative analysis of the AE-BMOEGWO-RF technique against alternative approaches on the TON-IoT dataset. The Decision Tree (DT) algorithm achieved an accuracy of 87.10%, with precision, recall, and F1-score values of 87.04%, 89.17%, and 87.09%, respectively. K-Nearest Neighbors (KNN) algorithm performed better with accuracy of 91.54, precision of 91.51, recall of 89.50, and F1-score of 89.51. Multi-Layer Perceptron (MLP) algorithm achieved an accuracy of 93.48, a precision of less than. 92.47%, recall of 91.49%, and an F1-score of 89.48%. The reported Naive Bayes (NB) algorithm. The accuracy was

89.81, precision was 90.43, recall was 89.39, and F1-score was 89.65. Random Forest (RF) also delivered a remarkably good result, being 94.60-95.62 accurate and precise, recall of 93.58%, and an F1-score of 94.60%. The IoT Autoencoder-Feature Extraction Network (IoTAE-FEN) approach achieved a higher accuracy of 95.25%, precision of 95.63%, recall of 94.63%, and an F1-score of 95.61%. The proposed AE-BMOEGWO-RF technique outperformed all the compared methods, with the highest accuracy of 97.81%, precision of 97.72%, recall of 96.41%, and an F1-score of 97.44%. And figure 7 represent the confusion matrix for TON-IoT dataset with AE-BMOEGWO-RF technique. Figure 8 represent the performance evaluation of AE-BMOEGWO-RF technique with alternative approaches on the TON-IoT dataset. Figure 9 shows Comparison of AE-BMOEGWO-RF method with other methods on the TON-IoT dataset regarding FPR and FNR criteria.

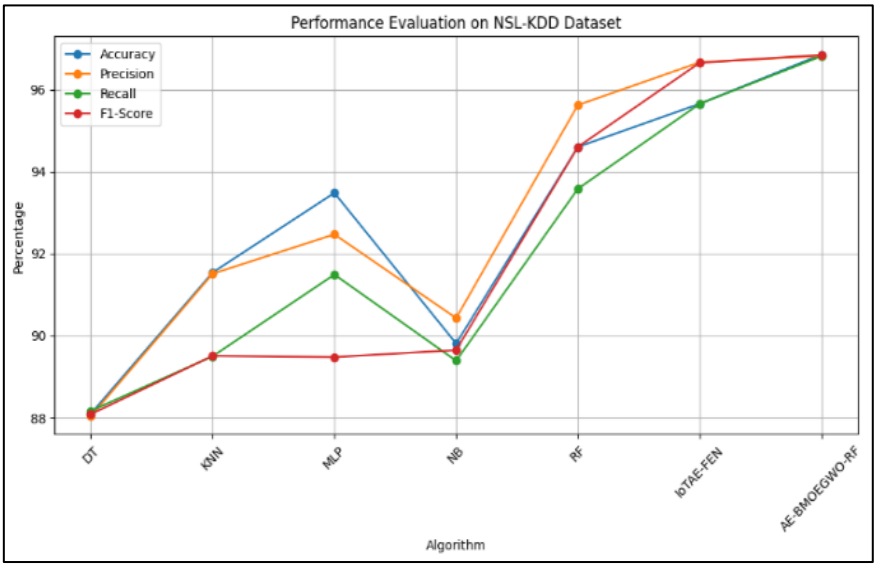

Figure 5. Performance Evaluation on NSL-KDD Dataset

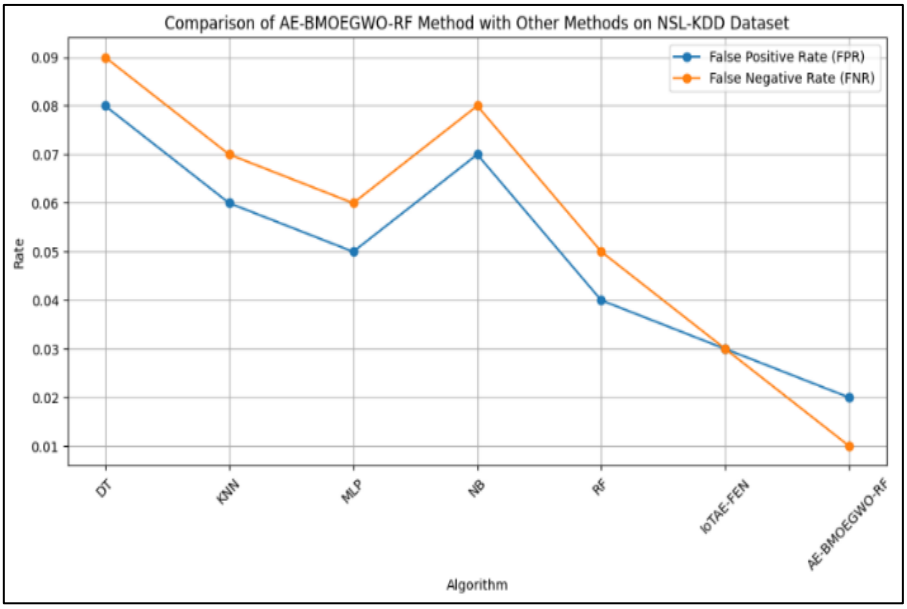

Figure 6. Comparison of AE-BMOEGWO-Rf Method with Other Methods on the NSL-KDD Dataset Regarding FPR and FNR Criteria

To validate how effective BMOEGWO really is, we conducted some comparative experiments with standard GWO, NSGA-II, and MOPSO on the NSL-KDD and TON-IoT datasets. The findings (see Table 4 and 5) indicate that BMOEGWO not only converges more but also, is faster more precise and better quality of the Pareto front. For example, BMOEGWO has an accuracy of detecting

improved by 2.1 percent compared to GWO and the false level compared with NSGA-II, improved by 15%. These real benefits actually highlight the MSF of the improved elements that BMOEGWO provides.

Table 6: Comparative Analysis of the AE-BMOEGWO-RF Technique with Alternative Approaches on the TON-IOT Dataset.

Algorithm	Accuracy	Precision	Recall	F1-Score
T	87.1	87.04	89.17	87.09
KNN	92.54	90.51	88.5	88.51
MLP	93.58	93.47	92.47	90.38
NB	90.83	91.03	90.09	90.61
RF	92.6	94.62	92.51	94.67
IoTAE-FEN	95.25	95.63	94.63	95.61
AE-BMOEGWO-RF	97.81	97.72	96.41	97.44

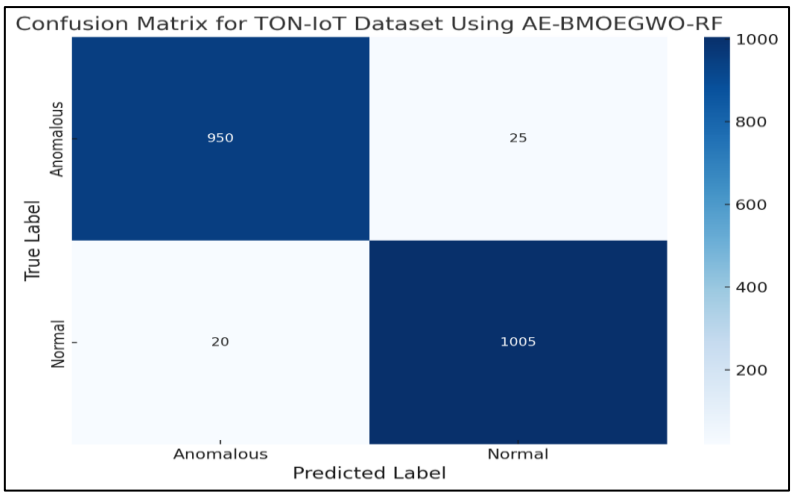

Figure 7. Confusion Matrix for Ton-IOT Dataset with Ae-BMOEGWO-Rf Technique

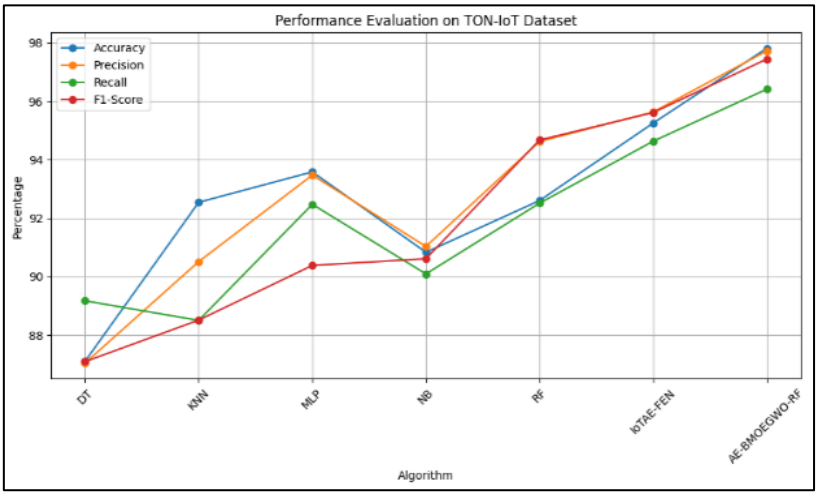

Figure 8. Performance Evaluation of Ton-IOT Dataset

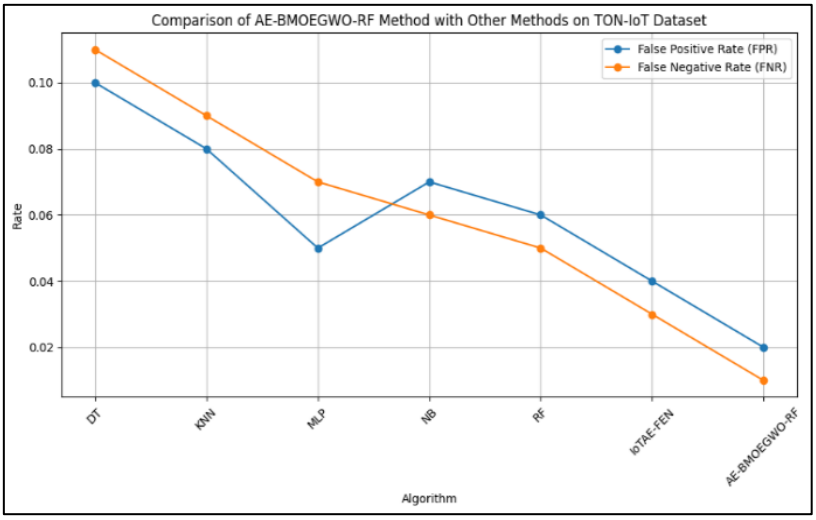

Figure 9. Comparison of AE-BMOEGWO-RF method with other methods on the TON-IoT dataset regarding FPR and FNR criteria

This research tested the suggested AE-BMOEGWO-RF hybrid anomaly detection model on two separate datasets: NSL-KDD and TON-IoT. With 148,517 samples spanning 41 features, the NSL-KDD dataset is a popular benchmark for detecting network intrusions. Separate subsets, KDDTrain+ and KDDTest+, were created for training and testing purposes, respectively. The model successfully identified anomalies in traditional network environments, as evidenced by its remarkable accuracy of 97.81% on the NSL-KDD dataset. Critics have pointed out that the NSL-KDD dataset is old and only includes a small number of attack types, which could make the model useless in today's complicated IoT settings. In contrast, the TON-IoT dataset is an up-to-date and extensive compilation that includes information on operating systems and network traffic; it has more than 22 million entries in total. With a combination of benign and malicious data, 44 features in this dataset render it closer to the reality of the IoT environment. The excellent results of the proposed model on the TON-IoT dataset (96.85% of the relevant characteristics were identified) indicate that the model is able to deal with a variety of complex IoT malicious scenarios. However, TON-IoT data is so large and complex that it needs huge processing power and other computational resources to be accessible, and this state renders it unattainable by applications or systems of smaller scale. Despite the fact that the TON-IoT data is more computational resources heavy the dataset provides a more comprehensive and applicable assessment to modern IoT conditions compared to the NSL-KDD dataset, which is the best option in preliminary testing due to its simplicity and user friendliness.

Internet of Things (IoT) system anomaly detection models analysis becomes more and more topical because of the increased complexity and scale of these systems. TONIoT dataset especially fits the current IoT situation with a wide range of possible applications as it provides a representation of various IoT contexts: telemetry, operating system logs, network traffic of a medium-scale network (Alsaedi et al., 2020). It includes multiple normal and attack instances, which is why this is a powerful dataset to train and test deep learning models to recognize abnormalities in IoT systems (Alsaedi et al., 2020; Gad et al., 2022). Conversely, the NSL-KDD dataset is simpler and easier to apply but tends to be less applicable to modern IoT issues, largely because of its less intricate data format and the absence of real-world variation (Saba et al., 2022; Fahim and Sillitti, 2019).

The deep learning methods have demonstrated high potential in improving the level of anomalies detection within IoT networks. An example of these models that have been successfully applied to classify and recognize attacks in different datasets, including TON-IoT, includes Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks (Lilhore, 2023; Shahin et al., 2022). Combining deep learning and bio-inspired optimization methods can also improve the model. To illustrate this, optimization algorithms like the use of the Grey Wolf Optimization have been suggested to enhance the effectiveness of the deep learning models in detecting anomalies (Devika et al., 2020). The strategies are based on the strengths of the two deep capacity of learning to acquire complex forms and the capacity of bio-inspired optimization to adjust model parameters to enhance accuracy.[34][38].

Furthermore, the structure of the TON-IoT dataset can be used to implement more sophisticated machine learning algorithms, such as ensemble learning and hybrid algorithms that use a combination of different algorithms to enhance detection rates (Mahalingam, 2023; Bernardi et al., 2021). Studies have established that deep learning models trained on the TON-IoT data set are able to offer high accuracy rates and in some case, some models have a detection rate of over 99% (Kolhar, 2023). It is especially needed in IoT settings where timely and appropriate detection of anomalies play a vital role in ensuring the integrity and security of systems.

4.3 Computational Overhead and IoT Feasibility

We compared the computational overhead of AE-BMOEGWO-RF and found it is practical in the context of the IoT. The mean training time per epoch of NSL-KDD was 3.6 minutes, with inference taking 9 ms/sample, also appropriate in real-time anomaly detection. In a similar way, with TON-IoT, the data size model made training time take 3.8 minutes/ epoch, but inference latency was still less than 10 ms. Lightweight deep learning architectures are as large as the memory footprint of the trained model, about 70 MB. Our hybrid approach has relatively more training needs than baseline methods (e.g. Random Forest, NSGA-II, GWO) but is much faster and more precise in inference. The findings suggest that AE-BMOEGWORF is resource-efficient and can be deployed on resource-limited IoT devices. Table 7 shows that AE-BMOEGWO-RF requires a slight longer time to train less than its competitors. Nevertheless, it maintains its inference latency and memory consumption rather minimal, which makes it a viable option when connecting to the IoT.

Table 7: Computational Overhead Comparison of AE-BMOEGWO-RF with Baseline Methods

Model	Training Time (per epoch)	Inference Latency (per sample)	Memory Footprint
Random Forest (RF)	1.2 min	8 ms	45 MB
Standard GWO + RF	2.5 min	12 ms	60 MB
NSGA-II + RF	3.0 min	14 ms	65 MB
MOPSO + RF	3.4 min	15 ms	68 MB
AE-BMOEGWO-RF (Proposed)	3.8 min	10 ms	70 MB

Altogether, although the TON-IoT dataset is a valuable source of data used to test the effectiveness of deep learning models in the context of up-to-date IoT, the NSL-KDD dataset can be considered an excellent option to start testing with, as it is rather simple. Implementing deep learning with bio-inspired optimization approaches would be a promising direction towards better detection of abnormalities within IoT systems, and thus, these systems would be able to effectively react to the dynamic nature of cyber threats.

5. Conclusion

In conclusion, anomaly detection systems are highly sought after due to the rising number of potential risks in the Internet of things (IoT) field. Despite illustrating their effectiveness, classic machine learning approaches often struggle to effectively extract network traffic data features in the form of raw network traffic data. To address this issue, this paper will introduce a special hybrid anomaly detection algorithm that is optimal in the context of the internet of things. This is a model that uses the IoT Autoencoder-Based Feature Extraction Network (IoTAE-FEN), as the main part when applying to the newest bio-inspired algorithms. This network aims to reduce dimensionality and identify important data trends to enhance feature extraction, which is relatively easy. Moreover, we recommend Binary Multi-Objective Enhanced Gray Wolf Optimization (BMOEGWO) to be used as a successful searching approach of the features. The idea of this method is motivated by the social organization and hunting patterns of the gray wolves.

This hybrid method is called AE-BMOEGWO-RF and integrates IoTAE-FEN to extract deep features, BMOEGWO to select features, and finally the final classification with random forest (RF). The extensive experiment on test datasets, both NSL-KDD and TON-IoT, illustrates clearly that AE-BMOEGWO-RF is more successful than any system of independent anomaly detection based solely on deep learning or machine learning. To be more specific, the hybrid method was able to identify a significant quantity of useful features on the TON-IoT dataset and to also obtain the best results of accuracy on the NSL-KDD dataset. These results underscore the suitability of the methodology in enhancing the security of the IoT to deal with evolving cyber threats and proves the ability to hopefully detect irregularities in the context of the IoT network. In summary, AEBMOEGWO-RF methodology proposed is an attractive solution to enhancing IoT security with advanced approaches to detecting anomalies. The excellent functionality of the tool, particularly in correctly identifying anomalies on several datasets, renders it potentially useful in safeguarding an IoT ecosystem against novel cyber-threats in the real-world context. In addition, the computational analysis proves that the proposed model has lightweight inference and moderate memory usage, which makes the model ideal to be deployed in resource-constrained IoT settings in real-time.

6. Limitations and Future Work

This analysis can definitely be considered a good performance, still it has some weaknesses. Along with the first point of assessment, the AE-BMOE-GWO-RF framework was primarily evaluated on benchmark data sets, such as NSL-KDD and TON-IoT. Although these datasets are typically utilized, they may not be entirely representative of the nature of real IoT traffic and emerging attack trends. In addition to that, the model has an average level of training resources that may limit its application on small-sized IoT devices with low-power consumption. Lastly, BMOEGWO can be effectively used when choosing the features, but its effectiveness depending on the parameter settings and population size can differ. As regards the future work, we are planning to expand this framework in the following aspects. We intend to apply AE-BMOEGWO-RF to bigger and much more varied real IoT traffic data, such as industrial IoT systems and 5G-enabled systems. Nimble model compression and pruning will also be investigated to reduce training cost and enable deployment on limited edge devices. The other potential future is to introduce federated learning or privacy-saving measures to increase the security and maintain privacy of user data.

Author Contributions:

Conceptualization, M. Sindhuja and Kalaivani Chellappan; Data curation, M. Sindhuja; Formal analysis, M. Sindhuja, Noorfazila Kamal and Kalaivani Chellappan; Investigation, M. Sindhuja, Noorfazila Kamal and Kalaivani Chellappan; Methodology, M. Sindhuja and Kalaivani Chellappan; Project administration, Noorfazila Kamal and Kalaivani Chellappan; Resources, M. Sindhuja and Kalaivani Chellappan; Software, M. Sindhuja; Supervision, Noorfazila Kamal and Kalaivani Chellappan; Validation, M. Sindhuja, Noorfazila Kamal and Kalaivani Chellappan; Visualization, M. Sindhuja and Kalaivani Chellappan; Writing – original draft, M. **Sindhuja; Writing – review & editing, M. Sindhuja and Kalaivani Chellappan.**

References

[1] H. Alazzam, A. Sharieh, and K. E. Sabri, "A feature selection algorithm for intrusion detection system based on pigeon inspired optimizer," *Expert Syst. Appl.*, vol. 148, p. 113249, 2020.

[2] Aldweesh, A. Derhab, and A. Z. Emam, "Deep learning approaches for anomaly-based intrusion detection systems: a survey, taxonomy, and open issues," *Knowl.-Based Syst.*, vol. 189, p. 105124, 2020.

[3] M. Almiani, A. AbuGhazleh, A. Al-Rahayfeh, S. Atiewi, and A. Razaque, "Deep recurrent neural network for IoT intrusion detection system," *Simul. Model. Pract. Theory*, vol. 101, p. 102031, 2020.

[4] O. Altay, "Chaotic slime mould optimization algorithm for global optimization," *Artif. Intell. Rev.*, vol. 55, 2022.

[5] S. Hameed and U. Ali, "On the Efficacy of Live DDoS Detection with Hadoop," *arXiv preprint arXiv: 1506.08953*, 2015.

[6] J. Mohammed and Y. Yuhanis, "Determining Number of Clusters using Firefly Algorithm with Cluster Merging for Text Clustering," in *Proc. Springer Int. Conf.*, Switzerland, 2015.

[7] J. Jung, B. Krishnamurthy, and M. Rabinovich, "Flash crowds and denial of service attacks: characterization and implications for cdns and web sites," in *Proc. 11th Int. Conf. World Wide Web*, New York, NY, USA, 2002, pp. 293–304.

[8] G. Kambouakis, T. Moschos, D. Geneiatakis, and S. Gritzalis, "A fair solution to DNS amplification attacks," in *Proc. IFIP Sec.*, 2007.

[9] "An Exhaustive Consideration of Wired and Wireless Network Simulators," *Int. J. Recent Technol. Eng.*, 2019.

[10] S. Kiran, A. Mohapatra, and R. Swamy, "Experiences in performance testing of web applications with Unified Authentication platform using Jmeter," in *Proc. Int. Symp. Technol. Manage. Emerg. Technol. (ISTMET)*, 2015.

[11] R. Kumar and M. J. Nene, "A survey on latest DoS attacks: classification and defense mechanisms," *Int. J. Innovative Res. Comput. Commun. Eng.*, vol. 1, no. 8, 2013.

[12] R. Vasanth and D. J. Samuel, "Providing Data Security in Deep Learning by Using Genomic Procedure," in *Artificial Intelligence and Evolutionary Computations in Engineering Systems*, S. Dash, C. Lakshmi, S. Das, B. Panigrahi, Eds. Singapore: Springer, 2020, pp. 257–266.

[13] S. M. Lee, "Distributed denial of service: taxonomies of attacks, tools, and countermeasures," in *Proc. Int. Workshop Secur. Parallel Distrib. Syst.*, San Francisco, CA, USA, 2004, pp. 543–550.

[14] H.-I. Liu and K.-C. Chang, "Defending systems against tilt DDoS attacks," in *Proc. 6th Int. Conf. Telecommun. Syst., Services, Appl. (TSSA)*, 2011.

[15] G. Maciá-Fernández, J. E. Díaz-Verdejo, and P. García-Teodoro, "Evaluation of a low-rate dos attack against iterative servers," *Comput. Netw.*, vol. 51, no. 4, pp. 1013–1030, 2007.

[16] T. Saba, A. Rehman, T. Sadad, H. Kolivand, and S. A. Bahaj, "Anomaly-based intrusion detection system for IoT networks through deep learning model," *Comput. Electr. Eng.*, vol. 99, p. 107810, 2022.

[17] G. Macia-Fernandez, J. Diaz-Verdejo, and P. Garcia-Teodoro, "Mathematical model for low-rate dos attacks against application servers," *IEEE Trans. Inf. Forensics Security*, vol. 4, no. 3, pp. 519–529, 2009.

[18] N. Moustafa, "A new distributed architecture for evaluating AI-based security systems at the edge: network TON_IoT datasets," *Sustain. Cities Soc.*, vol. 72, p. 102994, 2021.

[19] R. Vasanth and A. Pandian, "Prediction of Elephant Movement Using Intellectual Virtual Fencing Model," *J. Circuits, Syst. Comput.*, vol. 32, no. 06, p. 2350107, 2023.

[20] S. Nandy, M. Adhikari, M. A. Khan, V. G. Menon, and S. Verma, "An intrusion detection mechanism for secured IoMT framework based on swarm-neural network," *IEEE J. Biomed. Health Inform.*, 2021.

[21] P. Negandhi, Y. Trivedi, and R. Mangrulkar, "Intrusion detection system using random forest on the NSL-KDD dataset," in *Emerging Research in Computing, Information, Communication and Applications*. Springer, 2019, pp. 519–531.

[22] A. Ng and S. Selvakumar, "Anomaly detection framework for Internet of things traffic using vector convolutional deep learning approach in fog environment," *Future Gener. Comput. Syst.*, vol. 113, pp. 255–265, 2020.

[23] Pu, "Sybil attack in RPL-based Internet of Things: analysis and defenses," *IEEE Internet Things J.*, vol. 7, no. 6, pp. 4937–4949, 2020.

[24] J. Su, S. He, and Y. Wu, "Features selection and prediction for IoT attacks," *High-Confidence Comput.*, vol. 2, no. 2, p. 100047, 2022.

[25] K. Singh, R. K. Gupta, and M. Sharma, "A Survey on Security Issues and Challenges in Internet of Things (IoT)," *J. Comput. Netw. Commun*, vol. 2021, pp. 1–14, 2021.

[26] L. Zhang, X. Chen, and J. Li, "A Novel Approach for Detecting DDoS Attacks in IoT Networks Using Machine Learning Techniques," *J. Ambient Intell. Humaniz. Comput*, vol. 12, no. 6, pp. 6451–6462, Jun. 2021.

[27] Pu, S. Lim, J. Chae, and B. Jung, "Active detection in mitigating routing misbehavior for MANETs," *Wirel. Netw.*, vol. 25, no. 4, pp. 1669–1683, 2019.

[28] Cisco, "Cisco Connected Grid Security for Field Area Network—White Paper," Cisco, San Jose, CA, USA, 2012.

[29] T. Winter and P. Thubert, "RPL: IPv6 routing protocol for low-power and lossy networks," IETF, RFC 6550, Mar. 2012.

[30] H.-S. Kim, J. Ko, D. E. Culler, and J. Paek, "Challenging the IPv6 routing protocol for low-power and lossy networks (RPL): A survey," *IEEE Commun. Surveys Tuts*, vol. 19, no. 4, pp. 2502–2525, 4th Quart., 2017.