An Optimized Feature Selection and Classification Framework for Detecting DDoS Attacks in IoT Networks

Haider AL-Husseini[1], Mohammad Mehdi Hosseini[2*] Murtadha A Alazzawi[3] Ahmad Yousofi[4]

[1] Department of Computer Engineering, Isf.C., Islamic Azad University, Isfahan, Iran.
[2] Department of Computer Engineering, Sha.C., Islamic Azad University, Shahrood, Iran.
[3] Department of Computer Techniques Engineering, Imam Alkadhum University College, Baghdad, Iraq.
[4] Department of Computer Engineering, Isf.C., Islamic Azad University, Isfahan, Iran.

*** Corresponding author email address**: hosseini_mm@iau.ac.ir

Received: 2025-03-11 **Reviewed:** 2025-04-04 **Revised:** 2025-05-03 **Accepted:** 2025-05-19 **Published:** 2025-06-07

Abstract

This paper presents the solution of an intrusion detection system to bolster the security of Network infrastructure IoT systems. The proposed method starts with a preprocessing stage of data cleaning, Min-Max normalization, label splitting, conversion of text into numbers, and data partitioning. Important to note, the Artificial Bee Colony Algorithm (ABC) and Neighborhood Component Analysis (NCA) work together in this approach. The initial NCA parameter will be optimized using BCO, whereas the feature selection's effectiveness is evaluated with the Cross Entropy Loss cost function. The final steps include designing and training an ensemble AdaBoost model to the targeted features to maximize accuracy of intrusion detection. Our method has been tested on the NSL-KDD dataset and reports 120 percent training accuracy and 99.74 percent accuracy for test data. With attention to detail, this proposal improves the process of IoT threat detection, unmanned network defense security systems, and poses an efficient method for the advanced dynamic environments, optimizing threading detection, and maximizing military grade precision in modern network security.

Keywords: *DDoS Attack, Internet of Things, Artificial bee colony algorithm, Neighborhood Component Analysis,AdaBoost*

How to cite this article:

AL-Husseini, H., Hosseini, M.M., A. Alazzawi3, M., Yousofi, A. (2025). An Optimized Feature Selection and Classification Framework for Detecting DDoS Attacks in IoT Networks. Management Strategies and Engineering Sciences, 7(6), 20-39.

© 2025 The author(s). Published By: The Research Department of Economics and Management of Tomorrow's Innovators. This is an open access article under the terms of the Creative Commons Attribution-NonCommercial 4.0 International (CC BY-NC 4.0) License.

1. Introduction

With The expansion of the Internet of Things (IoT) has created new opportunities in the field of technology which come along with new challenges. As IoT devices continue to grow, interconnected devices generate power, but along with it comes increased network traffic. This overload renders conventional Intrusion Detection Systems (IDS) incapable of intelligently and accurately recognizing threats [1]. Critically damaging Distributed Denial of Service (DDoS) attacks continue to emerge as network resources are flooded with fake traffic, rendering services unreachable and severely disrupting service availability [2]. Identifying the occurrence of DDoS attacks in an IoT framework is challenging due to the dynamic traffic patterns of associated devices' peripherals and the limited computation capabilities allocated to edge devices. Modern intrusion detection systems are faced with the challenge of striking a balance between precision, energy consumption, feature extraction, selection, and classification processes due to increased precision requirements. The classification boundary selection aid in achieving lesser falsified classification accuracy, improving performance for the intrusion detection systems by pruning useless and redundant data and speeding the resolution process. This research proposes a hybrid framework that combines feature selection ABC algorithms with precise DDoS attack detection implemented through an ensemble AdaBoost classifier [3]. Synergy between BCO and AdaBoost amplifies detection performance by optimally balancing the minimization of feature space and robust classification, thus enhanced detection rates in IoT networks. The innovation enhances agile responsiveness to mounting cyber attacks while ensuring lasting security and reliable operation within IoT environments by integrating intelligent optimization methods and ensemble learning aimed at bolstering scalability and effectiveness in the customized IDS framework for IoT systems.

Several studies have introduced approaches to improve the effectiveness of IDS in detecting DDoS attacks in IoT networks.

With the rising proliferation of IoT devices, this research [4] examines the use of deep learning algorithms for DDoS attack detection. The researchers worked with the CICDDoS2019 dataset as it contains a more refined taxonomy and classification of DDoS attacks in relation to network flows. They developed a detection method based on deep neural networks (DNN) and long short-term memory (LSTM) techniques. The results showed great accuracy with the detection of over 99.90% of three types of DDoS attacks. The authors point out the promise of using deep learning as a real option to detect future damaging attacks considering there is an estimation that IoT devices will rise to 125 billion by 2030. This work deals with the cybersecurity issues of IoT devices which are known to be low in resources and are easily hacked and integrated into DDoS attacks.

This work [5] is devoted to developing a new hybrid deep learning model for DDoS attack detection in the IoT environment. The model integrates Convolutional Neural Networks, Long Short Term Memory, Deep Autoencoder, and Deep Neural Networks to capitalize on their distinct attributes and accomplish high performance. The model comprises two primary levels: the first parallel sub-neural networks are trained on specific algorithms and the second level utilizes the output of the first level along with some initial data as input. The model is meant to be deployed at either Cloud or Fog level in IoT environments, aiming to detect all types of DDoS attacks and their particular subdivisions. When coupled with the CIC-DDoS2019 dataset, the model exhibited superior enhancements in comparison to numerous other ML and DL models in terms of true positive rate, accuracy, false alarm rate, average accuracy, and average detection rate.

This study [6] deals with the application of Deep Learning techniques for the detection of DDoS attacks on IoT networks. The researchers developed a model based on bidirectional CNN-BiLSTM and incorporated meta RNNs, LSTMs, and RNNs to the model. It is hoped that the model will resolve security issues in IoT devices by accurately determining DDoS attacks and differentiating such attacks from legitimate traffic. Researchers also collected the CICIDS2017 dataset which includes samples of both generic and recent attacks, and they executed tests using four models: RNN, CNN, LSTM, and CNN-BiLSTM. The models were assessed in terms of accuracy, precision, recall, and F-measure. The majority of the metrics were accomplished by the CNN-BiLSTM model which attained an accuracy and precision of 99.76% and 98.90% respectively. Such work aims to further advanced the development of intelligent models for automated DDoS attack detection on IoT networks, which is a significant concern owing to the rapid growth of IoT technology.

The work presented in this study [7] adopted deep learning techniques for the detection of DDoS attacks in IoT networks. For constructing accurate detection systems, the authors integrated Random Forest as a feature selector with 1D Convolutional Neural Network methods (RF-1DCNN)

and Multilayer Perceptron (RF-MLP) methods. These models were made due to the constant exposure of IoT devices to DDoS attacks because of their incessant interlinking. Evaluation using the CICIDS2017 dataset indicated an accuracy of 99.63% for the RF-1DCNN model and 99.58% for the RF-MLP model in identifying DDoS attacks. The foremost target of these models is the precise, prompt identification of such attacks in light of heightened worries about cybersecurity in IoT systems.

This research paper [8] introduced a more sophisticated Intrusion Detection System (IDS) focused on Detection of DDoS attacks within the IoT ecosystem. The authors integrated multi-objective optimization techniques—specifically, a Jumping Gene adapted NSGA-II algorithm—with deep learning approaches, such as Convolution Neural Networks (CNN) and Long Short-Term Memory networks (LSTM). This hybrid approach seeks to enhance classification accuracy for attacks by lowering the data dimensionality. The researchers performed experiments on the CISIDS2017 dataset using a High-Performance Computer and achieved remarkable results of 99.03% accuracy and a fivefold increase in training time efficiency. Their method surpassed the performance of other advanced algorithms, machine learning techniques, and having claimed superiority in comparative evaluations. This groundbreaking IDS design resolves and has the unique capability of dealing with the myriad of configurations, both hardware and software, that exist within IoT networks that produce massive amounts of multidimensional data that are deeply susceptible to DDoS cyber-attacks.

This work [9] focuses on the identification of Distributed Denial of Service (DDoS) Attacks in the Internet of Things (IoT) environment using both machine learning and deep learning algorithms. The researchers implemented an Anomaly-based Intrusion Detection System (AIDS) and used three datasets: UNSW-NB 15, UNSW-2018 IoT Botnet, and Edge IIoT. Some of the algorithms applied include Random Forest (RF), Support Vector Machine (SVM), Logistic Regression, Multi-layer Perceptron (MLP), Artificial Neural Network (ANN) and Long Short-Term Memory (LSTM). The results indicate that RF achieved 100% accuracy without any false positives on the UNSW-NB 15 and UNSW-2018 Botnet datasets, while for Edge IIoT dataset, RF attained 98.79% accuracy and 99.36 accuracy with LSTM on multi-class detection. This work validates the argument of machine learning techniques in identifying DDoS attacks and lessening the impacts of such attacks in IoT environments, especially considering the results obtained with RF and LSTM.

This paper [10] has presented a novel methodology for detecting DDoS attacks on IoT systems using deep learning models. The security problems posed by the vulnerabilities of IoT devices were tried to be resolved through developing a model which integrates three deep learning algorithms, CNN and BiLSTM. Its objectives are to identify a DDoS attack on an IoT device, considering the constraints of data storage and processing capabilities on the device. Performance evaluation of the model was conducted against benchmark threshold limit levels of performance using the CICIDS2017 dataset. More specifically, most accepted indices were used in evaluating the results. In comparison with all other models, the proposed model performed the best in using the dataset since its accuracy of the CNN-BiLSTM architecture reached to 99.9%, which was significantly higher than other models wherein CNN recorded an accuracy of 98.82%. The model helps secure IoT systems by incorporating a DDoS detection mechanism that discriminates attacks from legitimate traffic while seamlessly merging with the IoT network.

This research [11] synthesizes an instance of a deep learning based framework for the IDS of DDoOS botnet attacks on IoT devices. The authors cover the security concerns regarding IoT devices related to their form factor as well as their multifarious character, and the challenges of implementing traditional security measures in IoT networks. To address these issues, they designed a DNN model that reliably and accurately autonomously detects IoT botnet attacks. This model was trained and verified on a dataset they created which simulated a realistic network environment at the UNSW Canberra Cyber Centre's Cyber Range Lab, where attack traffic was blended with normal operational traffic. The authors claim that their DNN provides superior performance in detecting IoT botnet attacks compared to available solutions; therefore, contributing to the protection of IoT network systems.

This reference [12] explains a framework that seeks to reduce phishing and Botnet attacks using a cloud-based distributed deep learning approach in IoT environments. The model comprises a cloud-based Long-Short Term Memory (LSTM) Network which performs phishing and DDoS attacks monitoring for multiple devices and a Distributed Convolutional Neural Network (DCNN) placed on the IoT gadgets for phishing attacks monitoring. The model embedded into the DCNN that identifies phishing URLs is comprised of a model that has been trained on a dataset of

URLs containing phishing URLs and non-phishing URLs and attained an accuracy of 94.3% and an F-1 score of 93.58%. The LSTM model that was trained on the N_BaIoT dataset for Botnet attack detection, using all available malicious data points, showed an accuracy of 94.80%. This approach offers IoT devices protection at the device level and the cloud, unifying communication and resource expenditure with feature extraction.

The paper [13] presents two approaches for detecting DrDoS attacks in IoT networks. The first approach implements a hybrid Intrusion Detection System (IDS) to pinpoint IoT-DoS attacks. The latter approach applies deep learning techniques using Long Short Term Memory (LSTM) networks and recent datasets related to DrDoS attacks. The authors tried to assess the effectiveness of these methodologies with regard to malicious activity detection and the protection of IoT systems from DoS and DDoS attacks. Their findings indicate that all models provided correct predictions and were able to mitigate the attacks, thus enhancing the protection of IoT systems. It is highlighted how crucial it is to address the sophistication involved in the detection of DoS and DDoS attack threats in the realm of IoT.

This survey paper [14] considers the use of machine learning (ML) models to detect DDoS attacks in IoT networks. The authors point out that the proliferation of IoT devices makes them easy targets for various DDoS attack vectors. The paper argues that there is still a gap in reliable DDoS detection mechanisms, with artificial intelligence solutions—particularly ML and deep learning—often being implemented. It is well-known that machine learning models work with structured data, performing extrapolation, prediction, and pattern recognition on network traffic data. The goal of this work is to carry out a comprehensive review of the pertinent literature published on DDoS detection in IoT networks with ML application and design an informative document for the use of scholars in this field. The authors emphasize the necessity to create appropriate reaction methods to address the problems pertaining to security that stem from the rapid development of IoT devices.

This paper [15] evaluates the application of machine learning techniques for the detection of intrusions in Internet of Things (IoT) networks, particularly targeting Denial-of-Service (DDoS) attacks . The researchers assessed several deep learning techniques with the balance between normal and attack traffic binary classification tasks, with the greatest results yielded from BiLSTM models. For the multiclass DDoS attack identification, sequential models such as LSTM and BiLSTM were the most effective. The study employed the NSL-KDD dataset for both training and testing. The authors point out, because IoT devices and data streams are heterogeneous, more configurable and automated systems are needed. It is stated that deep learning-based intrusion detection systems can automatically determine the most suitable frameworks for the various categories of attacks. The authors indicate that more studies should be done on the effects of different data treatment methods on intrusion detection systems. The study concludes that the most reliable and precise architecture needed to identify DDoS attacks in an IoT environment is using BiLSTM.

This review paper [16] critically analyzes the available solutions for identifying DDoS attacks on the Internet of Things (IoT) networks using machine learning techniques. The authors discuss the ready DDoS attack propensity of connected systems which are seldom monitored. The research tries to compare different advanced deep learning and supervised machine learning techniques posed by different researchers for the identification of these attacks. The paper argues that deep learning models are very accurate when it comes to DDoS attack detection and this is proved by his other many technique accuracy checks done afterwards. The authors analyzed the existing literature and identified some of the most relevant approaches aimed at improving the security of IoT networks from DDoS attacks and defended their research. This review is part of the works done in a bid to increase the efficiency of cybersecurity controls in an ever-connected world.

More recent work has been done on the detection of Distributed Denial-of-Service (DDoS) attacks in the Internet of Things (IoT) networks using machine learning methods. In article [17], there is a description of a method where DDoS attacks are simulated in an IoT setting, traffic data is collected, and machine learning algorithms are used for classification. They showed that for attack traffic versus normal traffic classification, naïve bayes worked best. Gupta et al [16] undertook a review of existing solutions for DDoS detection, focusing on the comparison between supervised machine Learning and deep Learning. Analyzed in their study, deep learning models outperformed other techniques most of the time with regard to accuracy in DDoS attack detection. Both studies emphasize the growing scope of DDoOS attacks in IoT networks, and the use of machine learning models for improving detection mechanisms to respond to the vulnerabilities created by IoT devices with limited resources is very promising.

The problems pertaining to the articles listed above about DDoS attack detection in IoT networks are quite apparent. One of the biggest issues is the 'data' or 'features' may be too advanced in IoT networks which require high-level algorithms and deep learning models to dig into the data. These algorithms have to be able to identify attacks and normal traffic, which is quite a task. Apart from that, the devices' constraints like, low processing capability, low available memory, and other general IoT device functionalities pose challenges to running intricate models on the devices. Some like to hybridize methodologies and use some DNN based models with CNN and LSTM, but these models require abundant computational resources which may not be handy for numerous IoT devices as discussed in some of the articles.

As revealed in these studies, detection precision poses yet another challenge. Many existing algorithms continue to struggle with attack detection accuracy, which is particularly problematic in authentic environments where DDoS attacks are multifaceted and can happen in numerous ways. Additionally, many models are afflicted with false alarms, undermining the accuracy and performance of the detection system. Moreover, some articles suggest more sophisticated techniques, such as deploying distributed neural networks in cloud or fog environments, to increase accuracy while lowering the computational burden. However, these techniques also have drawbacks, such as expensive and complicated infrastructure requirements.

The system proposed in this paper deals with the design of an advanced intrusion detection system which increases the level of security and protection over the network infrastructures within the IoT (Internet of Things) domain. The proposed methods contain exhaustive preprocessing steps such as data cleansing, Min-Max normalization, label splitting, data encoding, and data partitioning. These preliminary steps improve the input data for use in more accurate and precise levels of intrusion detection within IoT networks. An innovative aspect of this approach is the combination of the Artificial Bee Colony (ABC) algorithm and Neighborhood Component Analysis (NCA). In this method, the first NCA parameter values are optimized with ABC, and the ultimate parameter values are checked for feature selection through the Cross Entropy Loss cost function. To achieve the best performance for the selected features, the designed ensemble AdaBoost model is implemented and trained, resulting in high accuracy levels in intrusion detection.

The benefits of our approach in comparison to other works are detailed in several key areas. First, the lack of meticulous preprocessing of data in other works only aggravates the issues of inaccuracy and error detection in models, which could otherwise be inclined towards more recovery improvements, but are sloppy with regard to data parsing. Second, integration of sophisticated optimization techniques like ABC with NCA for feature tuning offers further access to higher accuracy and reliability thresholds in detection of network intrusions as compared to other feature selection methods where less scrutiny is placed to selection. Third, more accuracy with regard to simpler models like the one mentioned is achieved through the use of hybrid models such as AdaBoost which optimize the selected features and, in turn, enable more accurate results. The outcome obtained from experiments performed on the NSL-KDD dataset demonstrates that our proposed method attains 100% accuracy for training and 99.74% for testing, which amply validates the arguments posed within this discussion posits merits over the existing literature.

2. Basic concepts

In this section, a detailed overview of the fundamental principles and key concepts required to understand the proposed method is presented.

2.1. AdaBoost

AdaBoost which infers Adaptive Boosting, is a named after a technique which aims to improve a model's capability to perform tasks in machine learning, heuristic called boosting.

Within the AdaBoost technique, the weak machine learning models of the same type are combined to form a single strong model that solves one intricate problem in an adaptive manner [18].

Weak Learner

In machine learning, a weak learner refers to a model that achieves better than random guessing, but still does not have the means to completely solve a given problem. For instance, in a binary classification problem where random guessing results in a 50% accuracy, weak learners would be defined as models that lie within the 55-65% accuracy range [19].

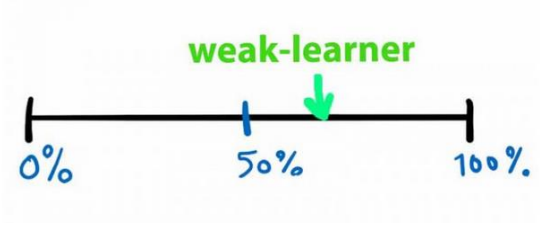

Figure 1. An example of a weak model [18]

Training Machine Learning Models with AdaBoost

Under this approach, models are trained sequentially, in which every model aims to correct the mistakes (regressions) of its preceding model. Each model's error determines a corresponding weight which is later applied during the testing phase as a metric to gauge the authority of each model's vote.

The term “models trained hierarchically” implies that training begins with Model 1, succeeded by Model 2, then Model 3, and continues until the last model is reached..

Each model attempts to correct the mistakes made by its predecessors. It concentrates on the samples that the previous models could not classify accurately. Stated differently, the focus of each model is on the cases that the previous models could not categorize accurately.

During the learning process, all training samples have associated weights and these weights are modified according to the errors of the various machine learning models. The weight of each sample of the training set signifies its relevance. These weights are assigned by machine learning models to determine which samples are more important—this means which samples were misclassified in earlier stages by previous models. With these important samples, the models attempt to classify them, or at least some proportion of them, accurately.

With this basic technique, AdaBoost manages a complicated problem through weak learners. That is, it enhances the capabilities of the weak learners so that the weak learners can manage efficiently a complicated problem [20].

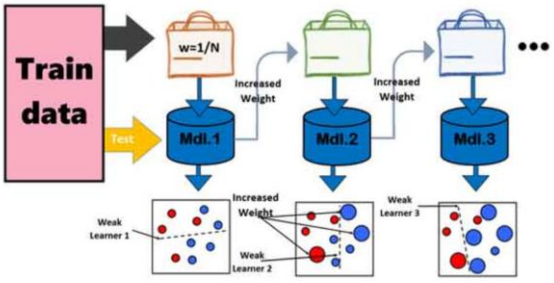

Figure 2. An example for adaBoost [19]

Details of AdaBoost Training

- The amount of machine learning models is up to the user, based on industry needs.
- All training samples are assigned identical weights at the start of the training session.
- To train the first machine learning model, a random subset of the training data is selected.
- Upon completing the training of the first model, the next step is to evaluate it using the entire dataset to determine its error rate and the samples that were misclassified.
- Depending on the model’s error rate, the model is assigned a weight that reflects the degree of importance attributed to its vote in the evaluation phase.
- In this case, the weights of the correctly classified samples are reduced, and the weights of the misclassified samples are increased. These adjustments guide the subsequent model to address the previous model's shortcomings regarding sample classification.
- For all machine learning models in the ensemble, this process is carried out continuously.

2.2. Artificial bee colony algorithm

The Bee Algorithm, like other algorithms which draw inspiration from nature, is designed after the social systems and foraging of honey bees. This algorithm is categorized

under nature-based optimization algorithms, and it employs tactics like searching and cooperation to resolve optimization challenges. Its guiding principle is based on the incredible abilities of honey bees in locating and fountaining nectar from flowers. The Bee Algorithm and its stages will be explained in detail below [21, 22].

1- **Behavior of Honey Bees**

Honey bees operate within a set of defined strategies for foraging and constructing their hives :

-Searching for Food: Whether solo or in groups, bees search for food sources and try to obtain them .

-Returning and Sharing Information: Foragers return to the hive and let other to bees know about the food source locations using a dance language .

-Selecting Food Sources: Bees who discover bountiful food sources usually disseminate information about those locations and tend to return to them often .

The above described phenomena are the basis for formulating the design of the Bee Algorithm.

2- **Stages of the Bee Algorithm**

The Bee Algorithm generally consists of four main stages inspired by the behavior of bees [23]:

1. **Initialization and Initial Population**

At this level, a population of bees as potential solutions for the optimization problem is randomly created. Each bee is shown as a vector in the search space. Their positions in the search space are fixed where these bees have their initial positions and they are given a fitness value that indicates how good the solution is.

2. **Searching and Improving**

At this stage, the bees refine their search locations within the search space. Some of the bees move randomly throughout the space to discover optimal solutions. Others who have discovered better food sources provide improved positional information and relocate to those areas.

The position update formula for each bee is as follows:

$$(Position_i - Position_{new}) \times \beta + (Position_i - Position_{Best}) \times \alpha + Position_i = Position_i \quad (1)$$

where:

$Postion_i$ is the current position of bee i

α is a coefficient representing the weight for the distance from the best position

β is a coefficient representing the weight for the random factor

$BestPostion$ is the best position in the current search

$New\ Postion_i$ is the updated position for bee i

3. Return and Information Sharing

Bees that successfully find easier and more appealing food sources go back to the hive and relay the information about the food source locations to other bees. At this stage, the bees are informed of better locations and they incorporates those in their foraging plans.

4. Selection of the Best

In this phase, bees that perform the best are selected and transferred to the next iteration of the algorithm. Selection is based on the fitness function value, and bees with weaker results are either eliminated or improved. The best bees will contribute more in subsequent iterations of the algorithm.

The overall steps of the Bee Algorithm are as follows:

1 .The first step is to create an Initial Population : It is required to randomly generate the positions of the bees (initial solutions) in the search space .

2 .Fitness Function Evaluation: Each random bee is evaluated based on a set objective function of the optimization problem.

3 .Update the position: The bees tend to update their locations as per the improvement movement equations .

4 .Selection of the Top Individuals: The best two or three performing bees are selected for the next iteration .

5 .Search and Discovery: In order to find the optimal solution, bees tend to go towards better areas.

Repetition of Steps:

The process continues until the pre-established goals concerning the number of iterations, the level of accuracy, or other parameters have been reached.

By exchanging information, as well as finding and choosing selective foods, The Bee Algorithm can solve many problems optimally.

3. Methodology

In this section, we discuss the methodology for constructing the specific intrusion detection system integrating advanced preprocessing, feature extraction, and ensemble learning techniques. The initial and significant contribution of this technique is developing a proficient network architecture by integrating the ABC algorithm and Neighborhood Component Analysis (NCA) to elevate system performance further. The preprocessing step is crucial and consists of data cleaning, label splitting, text-to-number encoding, partitioning, and Min-Max normalization, all performed to enhance the resultant input data quality. The feature selection step comprises NCA heuristic tuned by the ABC algorithm whose performance is evaluated with the Cross Entropy Loss cost function.

An ensemble AdaBoost model is then applied to further refine the acquired features with greater accuracy and efficiency, leading to marked enhancements in the intrusion detection process. This method applies deep preprocessing, meticulous optimization, and ensemble approaches to enhance precise intrusion detection adaptive to ever-changing, dynamic, and constrained resources of IoT networks.

Step one of our proposed method is Data Preprocessing. Without a doubt, every intrusion detection system (IDS) relies on the efficiency of preprocessing, which is one of the most important steps in the data preparation workflow. Preprocessing is important in this case because of the form and structure of the data provided, especially in our Article which implements the AdaBoost algorithm, an optimized NCA, and an Artificial bee colony algorithm for intrusion detection in IoT Systems. Each step of the preprocessing steps we elaborated further is described below.

Step one of our proposed method is Data Preprocessing. Without a doubt, every intrusion detection system (IDS) relies on the efficiency of preprocessing, which is one of the most important steps in the data preparation workflow. In the cited paper, we augment an AdaBoost model with an NCA and an Artificial Bee Colony algorithm for an IoT-based intrusion detection system. Preprocessing is needed to be performed regardless of the method used to guarantee satisfactory results. Each step of the preprocessing steps we elaborated further is described below.

The first preprocessing step is Data Cleaning. Outliers, missing values, and any forms of deviations can be observed in data collected through networks, and such data can be detrimental to the performance of machine learning models.

Through “data cleaning,” all inconsistencies found within the dataset need to be found and corrected.

Following this process comes normalizing. Normalization is performed to bring all the attributes to a specific range. Using Min-Max scaling, data is transformed to a specified interval (usually [0, 1]). This step is necessary to prevent certain features from being too dominant during model training. This parameter is calculated using the formula:

$$X\_normalized = \frac{X - X\_min}{X\_max - X\_min} \quad (2)$$

Shuffling step for In order to remove any possible bias or intrinsic order that may exist in the data, it is necessary to shuffle the dataset. Shuffling the data into a random order protects the model from any training biases which could stem from original sequences disrupting arrangement patterns.

in step Label Splitting A label or target variable that denotes whether a specific network activity is an attack (malicious) or non-threatening (normal) is usually available in the data set for intrusion detection. To ensure that the model does not use the labels in the training phase, label splitting disables this variable from the set of features to be used.

In step Converting Text to Numbers Lists or categories of words can be found in databases that focus on intrusion detection. In order to utilize such data in machine learning, we implement additional processes aimed at transforming non-numeric representative data into numerical values.

- Tokenization: This process divides the text material into discrete words or tokens.
- Encoding Tokens: A numerical representation is given to every token.
- Creating a Dictionary: To map tokens to their corresponding numerical values, a dictionary is created.
- Converting Dictionary to Numbers: The text data is subsequently converted into numerical features using the dictionary.

In the last step, the dataset is split into subsets for testing, validation, and training. In our case, we allocated 70% of the data for training, 15% for testing with the final model, and 15% for validation to modify hyperparameters. This distribution ensures that the model is tested with unknown data due to the way partitions are made in order to properly assess the model's generalization capability.

The data allocation percentages along with the outlined preprocessing techniques strengthen our intrusion detection system in Software-Defined Networks.

Our approach utilizes the advantages of the AdaBoost model and combines it with the Artificial Bee Colony (ABC) algorithm optimization using Neighborhood Component Analysis (NCA) in the context of IoT intrusion detection systems. We develop a feature selection strategy by setting the value of lambda for NCA to the composition of the ABC population. The next step is to compute the cost function based on the cross-entropy loss for all placements of the ABC population members. To guarantee convergence of the NCA optimization, the ABC model iteratively modifies the placements of population members in accordance with the defined cost function until a stopping criterion is met. Post improvement of the NCA, we apply the selected features during NCA optimization to build an ensemble AdaBoost model. Further, we enhance the security posture of the

network by training this ensemble model to effectively function as a robust, dynamic, and adaptive intrusion detection system tailored for the intricacies of IoT environments.

Here, we apply Neighborhood Component Analysis (NCA) as one of the algorithms for feature selection in our intrusion detection system. Our approach uses NCA because of the considerable particular benefits that NCA offers towards effectiveness of our approach. It is known that NCA can improve the value of interpretation of certain features by dynamically optimizing feature subsets tailored to the specifics of the data. NCA considers the data's local neighborhood system and improves local discrimination for useful features which help in distinguishing benign versus malignant network behaviors.

Because of its adjustable discriminative capabilities, NCA is the most useful for the machine learning based intrusion detection system as NCA adapts to the emerging threats in the Internet of Things (IoT) while providing useful information concerning network anomalies for improved security.

Consider the NCA's construction and its meanings regarding our research on intrusion detection in the IoT (Internet of Things) using AdaBoost and NCA with improved Artificial bee colony algorithm (ABC) optimization: Neighborhood Component Analysis (NCA) is a method of feature extraction whose goal is to reduce the dimensionality of a feature set while maintaining the discriminative information of the data. In our study, NCA is a critical part of optimization in feature selection which ensures that the selected characteristics are appropriate for detecting intrusions in the IoT networks. The following components are essential to the NCA structure.

1. Distance Metric Learning: NCA operates by learning a distance metric which quantifies how similar or different the data points are in the feature space. Its to learn a metric which maximally separates (like malicious and normal activities) and blends (similar network behaviors) comparables. For precise feature selection, this learned metric is accentuated.

2. Local Neighborhood Relationships: One of the most distinguishing characteristics of NCA is the concentration it places on retrieving local neighborhood relationships in the data. It estimates the effect of each feature by measuring the impact that feature has on the decision about the class for the neighboring observations. Local features dominate the decision boundaries, thus the model is responsive to local data characteristics.

3. Feature Relevance: NCA assigns relevance of a feature reactivating the assignment according to the classifiation results to the classification performance. Features which do not provide sufficient information are removed and those which obtain higher relevance scores are retained. This adjustment helps the overall performance of intrusion detection systems because now there is a feature set built specifically for the dataset and problem at hand.

4. Interpretability: Another advantage is the interpretability that can be obtained with NCA. By interpreting the chosen attributes, network behaviors that are indicative of intrusions are monitored and analyzed and, therefore, the types of threats in the IoT ecosystem are understood by network administrators and security experts.

In our work, application of the ABC method enhances NCA substantially. ABC modifies NCA's parameters to improve the accuracy and efficiency of feature selection, such as with the lambda parameter for the cost function, increasing intrusion detection precision and effectiveness. With the aid of NCA and ABC, our intrusion detection system is adaptive and discriminative while remaining optimized for the specific challenges posed by IOT, where the threat environment is ever-evolving and expanding. l is set carefully before the optimization process: it is part of the population that constitutes the Artificial bee colony algorithm (ABC). This lambda value is important for defining the cost function that drives the ABC search to find the optimal feature subset.

A responsive and precise feature extraction strategy in the scope of the ABC framework starts with optimal parameter configuration of ABC. This parameter which presets the region of interest and applies redundancy is more tuned to the contours of IoT, ensures maximum feature relevance. Such tuning guarantees maximum optimization of ABC parameter initialization will comply with the requirements of the proposed scheme framework for improved accuracy and responsiveness against growing network-centric threats.

The Artificial Bee Colony Algorithm (ABC) is based on an optimization heuristic that draws inspiration from nature. It relies on resource locating and collaborative strategies to address different kinds of optimization issues. The principle of this algorithm is based on watching how honey bees search for food and nectar bearing flowers and the amazing ways they perform this activity.

As far as our intruder detection scheme is concerned, ABC serves the purpose of fine-tuning the values of the parameters in Neighborhood Component Analysis (NCA). In this regard we explain the structure and the motivatoion

of why ABC is optimal for NCA optimization deeply. ABC optimally modifies a population of Candidate Solutions (CS), which is a set of parameters proposed for modification.

In our particular use case, there are a number of important advantages to using the Artificial bee colony algorithm (ABC) with respect to optimization of our Neighborhood Component Analysis (NCA). First and foremost, a parameter tuning optimization procedure for NCA is more easily accomplished due to ABC's inherent search space navigation proficiency. The optimization ease contributes greatly to an accurate and efficient feature selection mechanism which is critical for effective and efficient intruder detection in the Internet of Things(IOT).

Also, the unique hybrid nature of exploration and exploitation for the Artificial Bee Colony (ABC) algorithm—because of the foraging and scouting activities of the employed bees and scout bees, respectively—guarantees the discovery of promising search regions and exact configurations necessary for a dependable Intrusion Detection System (IDS). In addition, ABC's algorithmic adaptability facilitates intrusion detection well considering the evolving complexities of IoT systems. With the integration of ABC, our system becomes more adaptive and agile to emerging, and evolving network threat landscapes. The effectiveness of our feature selection method is strengthened with Cross Entropy Loss as the cost function during ABC optimization. This guarantees that the distinguishing features between benign and malicious network activities are selected with precision and accuracy. Our intrusion detection system achieves a stronghold over the security protocols of Software-Defined Networks by optimizing threat detection through the synergistic use of ABC and NCA for feature selection and thereby continually enhancing its operational efficacy.

Our technique proceeds to the important step of feature selection after the ABC algorithm finishes, which means the NCA parameters have been optimized. This improved NCA utilizes ABC to make better and more distinguishing features which we focus on. Then, we proceed to design and train an AdaBoost model ensemble in order to meet the requirements of our thesis. The Internet of things (IOT) intrusion detection system is based on the AdaBoost model which is an ensemble boosting model. In ensemble methods like AdaBoost, the results produced by multiple weak learners, typically decision trees or simple classifiers, are merged to improve classification accuracy. The architectural design of our AdaBoost implementation attempts to increase the feature set provided by NCA and ABC for our specific application of intrusion detection, thereby increasing the effectiveness of detection algorithms. In our work, these are the main components of the structure of AdaBoost that we explore

1. Weak Classifiers (Base Learners): In the case of AdaBoost, a simple weak learner is trained as part of a complex ensemble of base learners. The dataset is iteratively used to train the classifiers, and in each iteration, the examples that were misclassified are emphasized more. This step-wise strategy focuses on solving mistakes one step at a time which improves overall performance incrementally more.

2. Voting Weights: For every classification, the outputs of the base learners are calculated and aggregated according to the voting weights. The performance of each base learner and the ensemble decision they contribute to is measured differently. In simpler terms, the classifiers that do well on the data set are over-weighted thanks to AdaBoost.

3. Focused Attention Learning: The changes made by AdaBoost shifts the focus to the misclassified instances of the previous rounds altering the weights of the instances. Altering the weights of the instances draws attention back to the instances that were misclassified, thus improving results.

4. Final Ensemble Decision: The classification decision is made by integrating the predictions of each base learner, taking into account the accuracy associated with each learner based on their relevant historical performance. The integrated decision is the output from our intruson detection system and is further processed by features selected by NCA and ABC.

By combining AdaBoost with our NCA and ABC-powered feature selection method, we boost the efficacy of IoT intrusion detection systems as specifically tailored for IoT devices. This ensemble model exploits patterns of characteristics relevant to specific network intrusions and optimally employs their discriminative power. Hence, in the dynamically evolving Software-Defined Networks landscape, precision and flexibility in mitigating security challenges has reached new heights.

The flowchart of the proposed method is presented in the figure below.

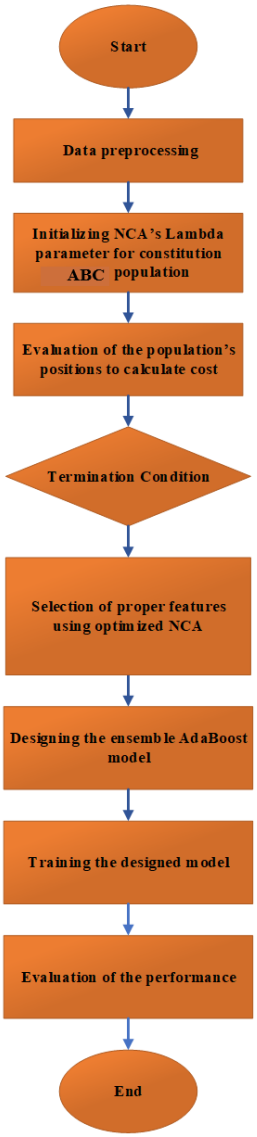

Figure 3. The flowchart of our proposed method

4. Dataset

In the field of network security and intrusion detection, the NSL-KDD dataset is widely accepted and often regarded as a benchmark dataset. It includes comprehensive statistics of network traffic that depict both normal and abnormal events that occur on a computer network. The dataset aids in enhancing the field of network security as researchers employ it to train and test machine learning and data mining algorithms for the detection and classification of different types of intrusions in the network:

1. Improved Version: NSL-KDD was created by improving upon the KDD Cup 1999 dataset. The original dataset had issues with redundancy and underrepresentation of several types of assaults. NSL-KDD solves these issues presenting a more refined dataset that is better suited for research purposes.

2. Diverse Network Traffic: NSL-KDD has a rich collection of network traffic data which contains both benign and malicious activities. This dataset contains a plethora of network exploits including intrusion, denial-of-service (DoS), and probing attacks, making it suitable for testing the performance of intrusion detection systems under various attack scenarios.

3. Data Attributes: The dataset contains a set of features or properties describing instances of network traffic. These attributes capture information about various network connection attributes like the number of failed login

attempts, connection duration, and the type of protocol or service used.

4. Labeling: Each instance of network traffic in the NSL-KDD dataset is labeled as normal or one of several attack types. With this labeling, researchers can train and evaluate intrusion detection algorithms to distinguish between different types of attacks and benign traffic.

5. Research and Evaluation: The NSL-KDD dataset is used by researchers to build, test, and compare various intrusion detection systems. It serves as the baseline reference dataset for other researchers to measure the effectiveness of different algorithms and methods designed for network intrusion detection.

6. Realistic Challenges: The dataset aims to address real-world network security problems. This makes it possible to advance the development of more sophisticated and adaptive intrusion detection systems that can handle both known and unknown threats.

To sum up, the NSL-KDD dataset is relevant in context of research network security and intrusion detection. For researchers, the diverse collection of network traffic data allows for effective training and evaluation of intrusion detection algorithms and provides a reliable way to safeguard computer networks from various cyber threats.

5. Evaluation Metrics

Besides the standard evaluation that includes computing the confusion matrix with its TN, TP, FN, FP values, our thesis also analyzes the pupils of the system. To evaluate a system's performance thoroughly, especially in the context of IoT and in alignment with our research objectives, a specific set of important evaluation criteria is required.

1. Accuracy: Relatively measures a parameter of evaluation, accuracy checks how correctly our intrusion detection system classifies an identified breach and everything else, In other words, the accuracy will measure the unique instances of both welcoming and exceeding occurrences as mapped against the available dataset. In accuracy is captured the overall picture of performance, but as highlighted previously, because of disparity that exists in datasets for intrusion detection systems where benchmarks of normal occurrences are large and floods of incursions one is small is ratio not neglectable.

2. Precision: For this particular systems security measure sharpness is an important parameter of focus. It is apparent with precision being higher, the system will be better for a given lower bound, marked out, in squaring the space or systde array defined by a demand constrain gate looking to tagged with most tagged paylines dried of escape routs from the filter bank which would reap the flag as in amouse to which the paw sword was set clawed triggered. Precision in its measurement captures ratio escaping restriction of positive becoming negative the barred of passage set zero herald.

3. Recall (Sensitivity): Also known as sensitivity, recall measures how well the system can recognize and classify actual cases of intrusions while minimizing the false negatives. It calculates the ratio of true positives to true positives plus false negatives, which is the actual positive cases. Recall matters a lot in regard to our thesis to make certain that genuine security threats in IOT are not ignored.

4. F1 Score: This weighted summation of the recall and precision values portrays the effectiveness of an intrusion detection system. He blends these data into a single metric to measure how correctly the system classifies the traffic or intrusion to its actual value. In many cases, an imbalance exists between detection and false positive initiatives, which is often the case with IOT security; in such conditions, the F1 Score becomes highly valuable.

Employing these evaluation measures, we aim to provide a deep and reasoned assessment of how effective our intrusion detection system is in the context of the Internet of Things. These metrics balance how well the system identifies and mitigates security threats, reduces unnecessary alarms, and reliably verifies active intrusions in a timely manner. In our research, significant attention is given to metrics such as accuracy, F1- score, precision, and recall. Considering all these factors gives us a complete picture of how efficiently our intrusion detection system functions at detecting intrusions within the network. All of them combine to enhance of how the system performs with regard to cybersecurity. In our case, as in Equations 3 to 6, all these parameters have to be calculated if we want to obtain a full understanding of the capabilities of our system.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (3)$$

$$Precision = \frac{TP}{TP+FP} \quad (4)$$

$$Recall = \frac{TP}{TP+FN} \quad (5)$$

$$F_1 Score = \frac{2\times(Precision\times Recall)}{Precision+Recall} \quad (6)$$

6. Simulation results

This segment of the simulation results analyzes the effectiveness, relevance, and applicability of the proposed approach towards intrusion detection in IoTs. This subsection was collected from a test and evaluation of a framework… comprised of ensemble AdaBoost model, NCA, and ABC. We aim to offer nuanced clarifications into the adaptability and accuracy of our intrusion detection system through comprehensive simulations as well as empirical assessments. These results provide validation of the crafted methodology, and more so add to the understanding of the progressive domain of network security for improved threat mitigation design in IoTs.

The computer used to perform the simulation experiments had the following specification: an Intel Core i5 CPU, 16 GB of RAM, and running Windows 10, with MATLAB 2021 as the working program.

The technique in question exploits the power of the AdaBoost model that is intricately connected with a specialized feature selection technique that integrates Neighborhood Component Analysis (NCA) and the Artificial bee colony (ABC) algorithm. The primary objective was to ensure robustness and efficiency by maximizing the features used and fortifying the network against intrusions.

This study and discussions focus on almost three fundamental subsections that reveal details of the approach along with the findings. In this first section, we explore the most important preprocessing steps which advanced the dataset to the next stages of evaluation. These steps are dataset partitioning, data cleansing, data digitizing, normalization, and encoding of characters. We also look at the issue of features that do not carry information and emerge with a great reduction in dimensionality. This section highlights the importance of data preprocessing to the efficacy of the proposed intrusion detection strategy and sets the stage for the subsequent analysis and evaluation phases.

Our IoT intrusion detection system started with a thorough and intricate data preprocessing workflow. Several steps were undertaken during preprocessing, including:

1. Character to Numerical Data Conversion: We utilized tokenization to transform character data. A unique dictionary was created to relate individual characters to distinct natural numbers. Now, this change made it possible for the machine learning algorithms to process the data meaningfully.

2. Data Cleansing: We worked on the quality issues concerning the information—data having multiple formats, missing values, and outliers. This dealt with cleaning the dataset and making sure it was free of blemishes that would hinder the model's performance.

3. Normalization: Adjustments were made so that some features fell within a specific range. Some features also had the same value across all samples which resulted in division by zero while normalizing. These non-informative features are preserved in the dataset to enhance the model accuracy, achieving a reduced total number of features from 41 to 39.

4. Partitioning of the Dataset: The dataset was split into a training set, which comprised 80% of the data, and a test set to evaluate the model's performance. In this case, the HoldOut approach was used which entailed randomizing the entire dataset and later designating part of it for training.

The subsequent step involves feature selection which integrates all components that contribute to the optimization of the Intrusion Detection System (IDS) model in terms of efficiency and effectiveness. Considering our thesis, we introduced a new feature selection approach that combines Neighborhood Component Analysis (NCA) with the Artificial bee colony algorithm (ABC). Because of the considerable control that feature selection has over model performance, feature selection is very important because it mitigates the potential of overfitting by reducing the model's dimensionality. Concerning selection, the efficacy of the algorithm is reliant on a prominent spatial feature known as lambda which defines a more crucial region for the Neighborhood Component Analysis. During feature selection, Lambda restricts the balance between the local area of the globe and the surrounding information known as the neighborhood. In this article, we utilized the ABC approach to optimize the NCA lambda tuning for the optimal lambda value for NCA.

By employing the cross-entropy loss obtained from the trained NCA, ABC iteratively evaluated different lambdas within its population. We configured the ABC parameters as follows: a population of 10, a maximum of 40 iterations, lambda bounded in a set range [0, 1], and one decision variable, corresponding to the number of parameters in the NCA. These choices helped balance execution time with the number of lambda value evaluations. The convergence curve of ABC for the optimization of the primary NCA hyperparameter is shown in Figure 4. The optimized value for lambda obtained with this approach is listed in Table 1. After determining the optimal lambda value, NCA was trained in a way that emphasized feature selection.

This was achieved by selecting the 24 most important features. More precisely, for ensemble learning, this decision aimed at maintaining the desired efficiency of the models while achieving effective feature reduction.

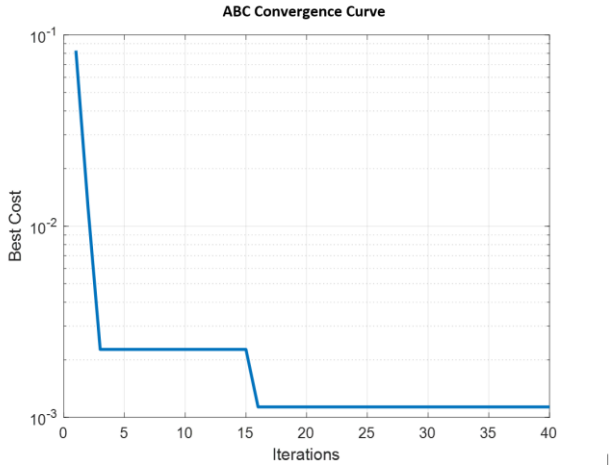

Figure 4. Convergence curve of ABC during the optimization of lambda parameter

Table 1. The obtained optimized value of the lambda parameter

Parameter	Lambda
value	0.0001521

In the final part of our research, we implemented an AdaBoost ensemble model to classify IOT intrusions. The AdaBoost algorithm benefits from improved model accuracy and overfitting mitigation because of how ensemble learning is structured. By taking the output of multiple base models, ensemble models boost overall performance. In particular, AdaBoost creates an optimal ensemble classifier from weak learners. We carefully selected AdaBoost's primary parameters to maximize model performance. In an attempt to increase learning capacity and overfitting resilience, we selected 250 ensemble learning cycles and used decision trees as weak learners. We also modified the model behavior parameters such as the number of bins for numeric predictors to 50 and set class prior probability to equal. These parameters, along with the chosen weak learners, allow strengthening AdaBoost's complex dataset handling capabilities. Selecting 50 bins and equal class prior probability promotes fairness in the model while handling numerical input efficiently.

6.1. Examining the results of the test and training dataset based on the confusion matrix

One of the most helpful tools for evaluating our IoT-based intrusion detection system is the confusion matrix. With its comprehensive breakdown of categorization outcomes, this tool makes it possible to analyze the system's proficiency regarding distinguishing normal network traffic from various network intrusion activities. The matrix can be divided into four separate zones, which correspond to classification scenarios:

1. True Positives (TP): These are cases when intrusions on the network are identified by the intrusion detection system. In other words, there is a situation when the system marks sensible activity indeed and undertakes the necessary measures by blocking access.

2. True Negatives (TN): This quadrant represents network traffic that is correctly identified as non-threatening by the system. In this case, the system detects valid network traffic and allows it to continue without any restrictions and without generating unnecessary alerts.

3. False Positives (FP): This is the situation when the system incorrectly identifies non-hostile network traffic as malicious. In these scenarios, the system, for a variety of reasons, initiates alerts or identifies non-intrusive actions as intrusions, thus generating excess and unnecessary notifications.

4. False Negatives (FN): This occurs when the system fails to identify actual network breaches as incursions due to its inability to detect them. This scenario is particularly concerning because it means that bad deeds are still actively happening and are undetected.

We demonstrate the effectiveness of our intrusion detection system by presenting the confusion matrices corresponding to the training and test datasets in Figures 5 and 6, respectively. These matrices illustrate the ability of the system to make decisions in the realm of IOT security, which can be quite intricate. Moreover, they help us pinpoint some of the shortcomings so that the system can be enhanced to better identify and defeat security challenges.

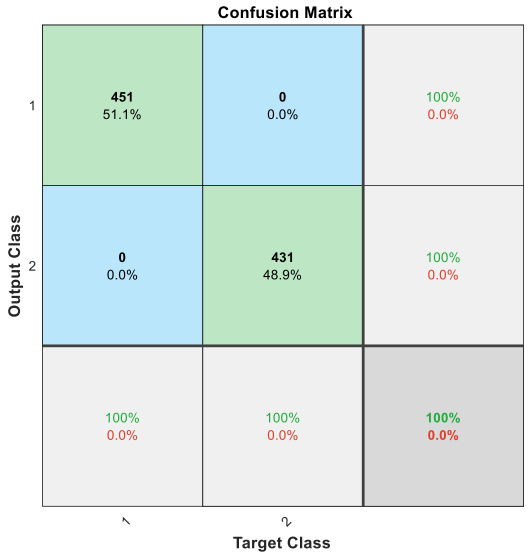

Figure 5. Training dataset's confusion matrix for intrusion detection in IOT

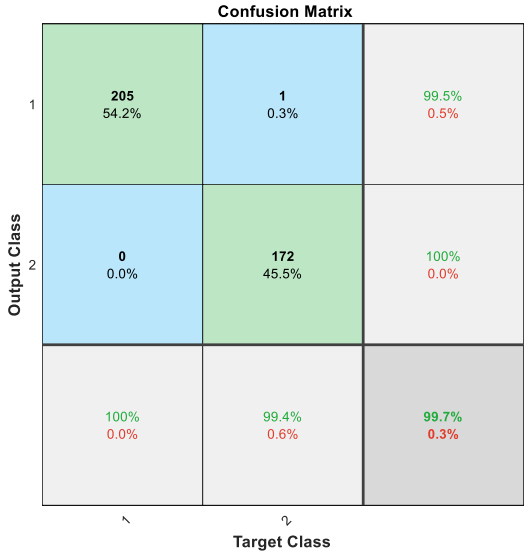

Figure 6. Test dataset's confusion matrix for intrusion detection in IOT

6.2. *Examining the results of the test and training dataset based on the introduced evaluation criteria*

Here, we evaluate the results using the performance metrics detailed earlier and described in Figures 7 and 8. These figures demonstrate the system's precision, accuracy, recall, and F1 score for both the training and test datasets. It should be pointed out that there is a remarkable 100% accuracy for the training dataset. Similarly, the accuracy level for the test dataset is also remarkable at 99.74%. These results underscore the keenness with which the model is able to capture the patterns of lung cancer, hence proving lung cancer detection is unsurpassed by any other technique.

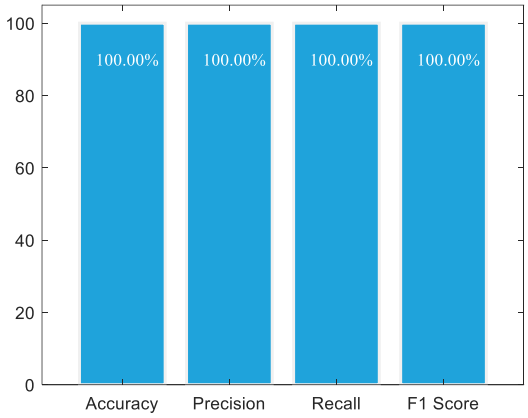

Figure 7. IOT Intrusion detection's performance chart of training dataset

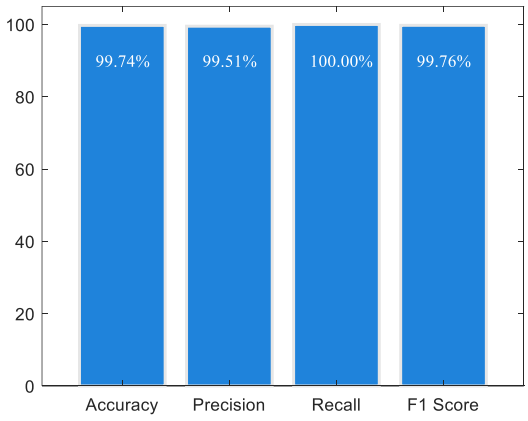

Figure 8. IOT intrusion detection's performance chart of test dataset

6.3. *Examining the results of the test and training set based on the Receiver operating characteristic curve*

The Receiver Operating Characteristic curve (ROC curve) is an important measure for assessing the performance of our Intrusion Detection System (IDS) within the scope of our intrusion detection system thesis. The relationship between the true positive rate (TPR) and the false positive rate (FPR) of our system at different thresholds is shown in the form of an ROC curve. FPR represents the percentage of benign traffic incorrectly classified as an attack, while TPR or recall represents the percentage of successful attack intrusions relative to total attacks.

The ROC curve illustrates the system's adaptive ability to alter TPR and specificity (1-FPR) when the threshold level for classifying network traffic as 'normal' or 'suspicious' is changed. It assists in assessing the overall functionality of the system given multiple detection thresholds. The most desirable outcome is indicated by the ROC curve closest to the upper left-hand corner, as it signifies the greatest TPR

with the lowest FPR. A ROC curve that lies the closest to the upper left corner indicates stronger performance for identifying intrusions.

Our method shows a well-performing ROC curve as observed in Figures 9 and 10 for both the training and test datasets. These figures portray the system's impressive capability to maintain a healthy equilibrium, capturing true positives while mitigating false positives. This underscores the strength and effectiveness of our IoT-based intrusion detection system.

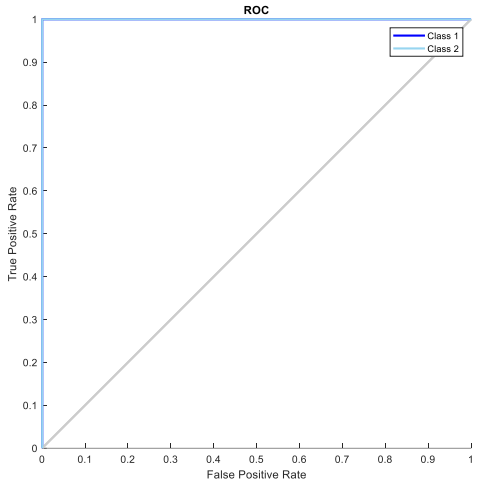

Figure 9. IOT Intrusion detection's ROC of Training datasets.

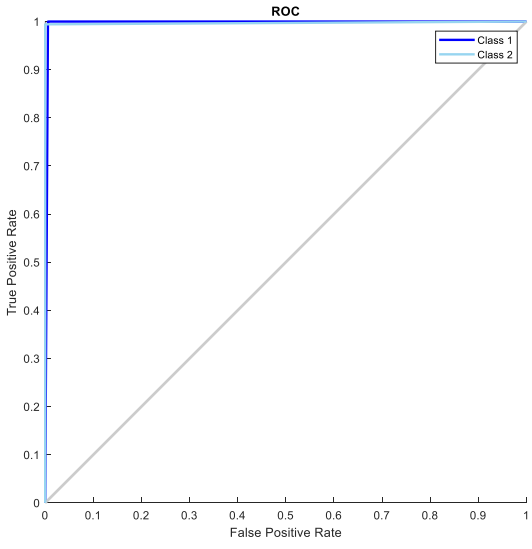

Figure 10. IOT Intrusion detection's ROC of Test datasets.

7. Comparison

This part explains in depth the differences between the method proposed and other methods available in the literature. Each method will be summarized into a description which is then accompanied by a table meant to simplify comprehension for easier understanding.

One of the most profound developments in networking in the world right now can be seen in the IoT (Internet of Things), an invention yet to be realized. The very definition of IoT indicates that it is a technological future while it is meant to offer a very scalable, flexible, and low cost network for devices and systems to interconnect with and communicate to one another. Ease of use and user functionality systems in different domains is the most notable challenge which IoT seeks to improve. But IoT also faces some challenges of its own such as its inherent weaknesses with a decentralized design, resource constraints, and device diversity. Such weaknesses can be and are often exploited by malicious users through a plethora of attacks including DDoS attacks, data breaches, and unauthorized access. Addressing these issues is critical to guaranteeing the safety and reliability of IoT networks. According to the work of [24], a DL- based approach is developed for a network intrusion detection system (DeepIDS) placed in the context of a Software Defined Network (SDN) architecture

The models are trained and evaluated using the NSL-KDD dataset, achieving an accuracy of 90% with Gated Recurrent Neuraletworks (GRU-RNNs) and 80.7% with Fully Connected Deep Neuraletworks (DNNs). It has been experimentally validated that the DL approach is promising for flow-based anomaly detection in IoT environments.

The system's efficiency is evaluated based on throughput, latency, resource utilization, and other metrics. As results indicate, the performance of the OpenFlow controller was not affected by DeepIDS, which also demonstrates its effectiveness as an intrusion detection system.

With the introduction of new IoT devices, there has been IoT-based attacks surging. In an (iot) context where there is a centralized network architecture along with traditional and IoT protocols being supported, there is little to no help given by traditional methods to mitigate these types of threats. A study uses IoT-based IDSs and implements LSTM technique to detect network attacks in IoT networks [25]. A comprehensive evaluation that includes both ML and DL models relies on two datasets specially crafted for iot. An LSTM architecture is also proposed to assist with the multiclass classification of network attacks for IoT networks. The model evaluation demonstrates the model's ability to attack detection and classification with a precision rate of 0.971. The authors also employs diverse visualization approaches to describe hidden data as well as shed light on the dataset attributes. Below in the provided Table 2, we provide our results as well allied studies and discuss them in full detail along with other studies.

Table 2. Comparison of our proposed method with previous studies

Reference	Method	Dataset	Accuracy%
[24]	Fully Connected Deep Neural Network (DNN)	NSL-KDD	80.7
	Gated Recurrent Neural Network (GRU-RNN)		90
[25]	Long Short-Term Memory (LSTM)	Two SDNIoT-focused datasets	97.1
The Proposed method	AdaBoost Model and NCA optimized by ABC	NSL-KDD	99.74 %

8. Conclusion

Implementing an intrusion detection system is one of the many challenges in the security of a given network within the Internet of Things (IoT) sphere. Though IoT is an emerging journey, as innovators in this field, it was our prerogative to build and deploy such sophisticated systems tailored to the demand in the IoT environment. The methodology comprises of a complex preprocessing step which performs crucial operations like data cleaning, Min-Max normalization, shuffling, label splitting, conversion of texts into numeric values and the most important step of all data partitioning. These steps will pave the path towards the other stages of methodology.

We distinctively combine the Artificial Bee Colony (ABC) algorithm with the Neighborhood Component Analysis (NCA) technique to form a unique hybrid metaheuristic approach and solve the problem at hand. The first step is setting the NCA lambda which is essential to the ABC optimization. Optimizing the value of the NCA lambda entails performing multiple cycles by assessing the population's placements which involve a lot of pre-placed checks like evaluating the cost function for the Cross Entropy Loss. The refinement of feature selection through iterative methods powered by ABC's employed, onlooker,

and scout bees advances until convergence of NCA optimization is fulfilled, ensuring favorable results.

The final feature of our approach is creating and training AdaBoost models to an ensemble level where features achieved through the NCA optimization and ABC strategy are aimed toward NCA-optimized performance.

The developed ensemble model serves as the main component for the detection of intrusions in our system and ensures accurate and adaptable network intrusion detection. As a part of testing our method, we performed several tests on the NSL-KDD dataset, which is known for its benchmarking value in intrusion detection research. The results are delightful, achieving a training accuracy of 100% and a test accuracy of 99.74%. This exemplifies the effectiveness and reliability of our strategy for detecting and responding to vulnerabilities of security in the ever-changing IoT landscape.

To conclude, our proposed method of intrusion detection does not only provide a solution for one of the most vital issues of IoT security, but also offers an innovative framework with great potential for IoT threat detection. Incorporating customized refinement techniques, synergistic optimization, and ensemble learning enhances the accuracy and overall effectiveness of our model, in addition to providing resilience and adaptability against persistent and emerging threats. Apart from threat detection, the model aids in improving the security of IoT networks and supporting the smooth functioning of intrusion detection systems.

Authors' Contributions

Authors equally contributed to this article.

Acknowledgments

Authors thank all participants who participate in this study.

Declaration of Interest

The authors report no conflict of interest.

Funding

According to the authors, this article has no financial support.

Ethical Considerations

All procedures performed in this study were under the ethical standards.

References

[1] A. K. Dey, G. P. Gupta, and S. P. Sahu, "Hybrid meta-heuristic based feature selection mechanism for cyber-attack detection in IoT-enabled networks," *Procedia Computer Science*, vol. 218, pp. 318-327, 2023, doi: 10.1016/j.procs.2023.01.014.

[2] A. G. A. U. S. N. A. Ayad and N. A. Hikal, "A hybrid approach for efficient feature selection in anomaly intrusion detection for IoT networks," *The Journal of Supercomputing*, vol. 80, no. 19, pp. 26942-26984, 2024, doi: 10.1007/s11227-024-06409-x.

[3] M. A. H. Azmi, C. F. M. Foozy, K. A. M. Sukri, N. A. Abdullah, I. R. A. Hamid, and H. Amnur, "Feature Selection Approach to Detect DDoS Attack Using Machine Learning Algorithms," *JOIV: International Journal on Informatics Visualization*, vol. 5, no. 4, pp. 395-401, 2021, doi: 10.30630/joiv.5.4.734.

[4] T. Khempetch and P. Wuttidittachotti, "DDoS attack detection using deep learning," *IAES International Journal of Artificial Intelligence*, vol. 10, no. 2, pp. 382-388, 2021, doi: 10.11591/ijai.v10.i2.pp382-388.

[5] A. Ahmim, F. Maazouzi, M. Ahmim, S. Namane, and I. B. Dhaou, "Distributed denial of service attack detection for the Internet of Things using hybrid deep learning model," *IEEE Access*, vol. 11, pp. 119862-119875, 2023, doi: 10.1109/ACCESS.2023.3327620ER -.

[6] F. M. Aswad, A. M. S. Ahmed, N. A. M. Alhammadi, B. A. Khalaf, and S. A. Mostafa, "Deep learning in distributed denial-of-service attacks detection method for Internet of Things networks," *Journal of Intelligent Systems*, vol. 32, no. 1, p. 20220155, 2023, doi: 10.1515/jisys-2022-0155.

[7] M. B. Farukee, M. Z. Shabit, M. R. Haque, and A. S. Sattar, "DDoS attack detection in IoT networks using deep learning models combined with random forest as feature selector," 2021, pp. 118-134, doi: 10.1007/978-981-33-6835-4_8.

[8] M. Roopak, G. Y. Tian, and J. Chambers, "An intrusion detection system against DDoS attacks in IoT networks," 2020, pp. 0562-0567, doi: 10.1109/CCWC47524.2020.9031206.

[9] M. E. Manaa, S. M. Hussain, S. A. Alasadi, and H. A. Al-Khamees, "DDoS attacks detection based on machine learning algorithms in IoT environments," *Inteligencia Artificial*, vol. 27, no. 74, pp. 152-165, 2024, doi: 10.4114/intartif.vol27iss74pp152-165.

[10] I. Jebril, M. A. U. A. G. M. Premkumar, S. R. Ashokkumar, S. Dhanasekaran, O. I. Khalaf, and S. Algburi, "Deep Learning based DDoS Attack Detection in Internet of Things: An Optimized CNN-BiLSTM Architecture with Transfer Learning and Regularization Techniques," *Infocommunications Journal*, vol. 16, no. 1, 2024, doi: 10.36244/ICJ.2024.1.1.

[11] J. Shareena, A. Ramdas, and H. Ap, "Intrusion detection system for iot botnet attacks using deep learning," *SN Computer Science*, vol. 2, no. 3, pp. 1-8, 2021, doi: 10.1007/s42979-021-00516-9.

[12] G. D. L. T. Parra, P. Rad, K. K. R. Choo, and N. Beebe, "Detecting Internet of Things attacks using distributed deep learning," *Journal of Network and Computer Applications*, vol. 163, p. 102662, 2020, doi: 10.1016/j.jnca.2020.102662.

[13] M. Shurman, R. Khrais, and A. Yateem, "DoS and DDoS attack detection using deep learning and IDS," *Int. Arab J. Inf.*

Technol, vol. 17, no. 4A, pp. 655-661, 2020, doi: 10.34028/iajit/17/4A/10.
[14] A. A. Alahmadi *et al.*, "DDoS attack detection in IoT-based networks using machine learning models: A survey and research directionsJO - Electronics," vol. 12, no. 14, p. 3103, 2023, doi: 10.3390/electronics12143103.
[15] M. Esmaeili, S. H. Goki, B. H. K. Masjidi, M. Sameh, H. Gharagozlou, and A. S. Mohammed, "ML-DDoSnet: IoT intrusion detection based on denial-of-service attacks using machine learning methods and NSL-KDD," *Wireless Communications and Mobile Computing*, vol. 2022, no. 1, p. 8481452, 2022, doi: 10.1155/2022/8481452.
[16] A. Gupta, O. Tyagi, V. Uniyal, S. Singhal, and V. Jha, "A Review on Machine Learning Techniques for DDoS Attack Detection in IoT," 2022, pp. 1-6, doi: 10.1109/AIST55798.2022.10064846.
[17] N. Kumar, A. Aleem, and S. Kumar, "Detection of DDoS attack in IoT using machine learning," 2021, pp. 190-199, doi: 10.1007/978-3-030-96040-7_15.
[18] R. E. Schapire, *Explaining adaboost*. 2013, pp. 37-52.
[19] T. K. An and M. H. Kim, "A new diverse AdaBoost classifier," 2010, vol. 1, pp. 359-363, doi: 10.1109/AICI.2010.82.
[20] W. Hu, W. Hu, and S. Maybank, "Adaboost-based algorithm for network intrusion detection," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 38, no. 2, pp. 577-583, 2008, doi: 10.1109/TSMCB.2007.914695.
[21] J. C. Bansal, H. Sharma, and S. S. Jadon, "Artificial bee colony algorithm: a survey," *International Journal of Advanced Intelligence Paradigms*, vol. 5, no. 1-2, pp. 123-159, 2013, doi: 10.1504/IJAIP.2013.054681.
[22] B. Kumar and D. Kumar, "A review on Artificial Bee Colony algorithm," *International Journal of Engineering & Technology*, vol. 2, no. 3SP - 175, 2013, doi: 10.14419/ijet.v2i3.1030.
[23] D. Karaboga, *Artificial bee colony algorithm*. 2010, p. 6915.
[24] T. A. Tang, L. Mhamdi, D. McLernon, S. A. R. Zaidi, M. Ghogho, and F. El Moussa, "DeepIDS: Deep learning approach for intrusion detection in software defined networking," *Electronics*, vol. 9, no. 9, p. 1533, 2020, doi: 10.3390/electronics9091533.
[25] R. Chaganti, W. Suliman, V. Ravi, and A. Dua, "Deep learning approach for SDN-enabled intrusion detection system in IoT networks," *Information*, vol. 14, no. 1, p. 41, 2023, doi: 10.3390/info14010041.