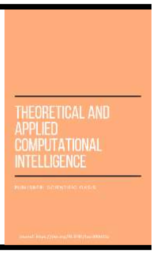
SCIENTIFIC OASIS

Theoretical and Applied Computational Intelligence

Journal homepage: www.taci-journal.org
ISSN: 3009-3325



THEORETICAL AND APPLIED COMPUTATIONAL INTELLIGENCE

An Implementation of Effective Machine Learning Approaches to Perform Sybil Attack Detection in IoT Networks

Hafiz Burhan ul Haq[1], Muhammad Saqlain[2,*]

[1] Department of Information Technology, Faculty of Computer Sciences, Lahore Garrison University, Lahore 54000, Pakistan
[2] Departments of Mathematics, Faculty of Science, King Mongkut's University of Technology Thonburi (KMUTT), Bangkok 10140, Thailand

ARTICLE INFO

Article history:
Received 23 July 2023
Received in revised form 23 August 2023
Accepted 23 August 2023
Available online 23 August 2023

Keywords:
Contiki-Cooja; Sybil Attack; IDS; Security

ABSTRACT

The rapid expansion of the technology industry has resulted in the emergence of many new areas of research, one of which is known as "intrusion detection". The objective of an intrusion detection system is to categorize user behaviors as either benign or malicious, and then to notify the relevant parties under this classification. However, varieties of strategies for attack detection have been developed, such as the Sybil attack. However, present techniques are restricted to concentrating on both aspects simultaneously because of constraints in detection accuracy and energy consumption. One such strategy is the Sybil attack. To circumvent these restrictions, a framework for the detection of Sybil attacks that is more effective in terms of detection accuracy (security) and energy usage (power) is presented. Nevertheless, the suggested structure is simple, uncomplicated to understand, and does not need any computing requirements. In this particular architecture, in addition to the Cooja-Contiki simulator, three distinct machine-learning methods are used. The NSL-KDD dataset is used to test the performance of the proposed framework. This dataset attained the maximum accuracy possible, which was 99.1 %. In addition, a comparison examination of the suggested work with the state-of-the-art is carried out to.

1. Introduction

Wi-Fi is an absolute need in today's society due to the increased reliance that smart gadgets have on wireless sensor networks. Wi-Fi may be found in a broad range of public and private locations, such as offices, bars, military sites, schools, and cafés. Research in wireless communication and electronics has made significant strides in recent years, which has paved the way for the creation of low-power, low-cost, and compact multifunctional sensor nodes. A wireless sensor network is formed by a large number of individual sensor nodes that are very inexpensive and compact [1]. This network may then be used to collect data and monitor the environment around it.

[*] *Corresponding author.*
E-mail address: msgondal0@gmail.com

https://doi.org/10.31181/taci1120232

The paradigm change that has been brought about by the Internet of Things (IoT) is quickly becoming a topic of discourse that is mainstream. In an IoT application, the objects themselves may be equipped with sensors, actuators, and an Internet Protocol (IP) connection for them to be able to operate independently and satisfy a sensory demand. These devices have a restricted capacity for both memory and computational power. While new IoT applications need mobile items, traditional networks are built up almost entirely of fixed ones (nodes). The behaviors of mobile nodes and their neighbours will cause a network's structure as well as the patterns of data flow on that network to change over time. As a direct consequence of this, the likelihood of discovering new vulnerabilities in the network's security has increased [2]. Some of the common attacks related to IoT are discussed below.

1.1 Categories of Network Attacks in IoT

1.1.1 Denial of Service

A malicious packet or an attacking node that obstructs a device’s communication channel or bandwidth may start a denial of service attack (DOS). In this form of attack, the victim could be attacked simultaneously from several distinct angles. This is a potentially dangerous attack on the IoT [3].

1.1.2 Node-Imitation Assault

The attacker in this form of assault can create their node, replete with a fictitious ID. The node thus has a possibility of receiving communications that were not intended for it. With this technique, the attacker can attack the whole IoT ecosystem.

1.1.3 Assault-related Application-level

In this type of attack, the attacker concentrates first on the target node, alters the transmission, and then sends back it to the original target node. The attacker can trick the nodes by utilizing this technique to give them false signals.

1.1.4 Sybil Attacks

In this type of attack, the attacker copies the source node, changes its identity, and then forces all other nodes to get away from the network. A Sybil attack utilizes a single node to operate several active fake identities (sometimes referred to as Sybil identities) within the framework of a peer-to-peer network. By capturing control of the great majority of key nodes in the network, this type of attack aims to reduce the authority or power of a reputable system [3].

The Sybil attack, intended to cause an attack on security, is the one that harms wireless sensor networks the most. One type of assault that can happen in peer-to-peer networks is a Sybil assault. By having one network node effectively manage many identities at once, this type of assault jeopardizes the authority or power in credibility schemes. To carry out illegal acts on the infrastructure, the attack's main goal is to gain control of the majority of the network. A single entity (a computer), which is also capable of doing so, may generate and maintain many identities (user accounts, accounts on an IP address). Outsiders nearly always perceive these many fictional personas as real and consistent people.

The proposed study describes Sybil attack detection (SAD) using Contiki and Cooja. We can test out the scenario we want to develop using Contiki using Cooja, a simulator that is incorporated into Contiki. Now that technology has advanced, many gadgets and pieces of equipment can communicate with one another without a person's help. The IoT refers to this kind of communication. IoT communication encompasses a wide range of applications, including smart cities, corporate communications, defense hardware, traffic systems, highway patrols, smart workplaces, smart toll collection, and satellite television. Linked to one another, these devices communicate with one another mechanically. The network is effectively protected by Contiki and Cooja, which are used in IPv6 protocols. The suggested models will have 15 distinct Sybil assaults, including ftp_write and password guessing. Security and energy were two other fundamental factors that the suggested framework focused on. The first component relates to how successfully the suggested algorithm detects the Sybil assault, while energy demonstrates how our proposed models are reliable and use less computing power in comparison to existing state-of-the-art techniques. The suggested model is evaluated using the NSL-KDD Dataset [4]. We utilized information from comparable simulations of real-world scenarios since there were not enough publicly accessible IoT attack statistics. Cooja simulation creates raw packet capture files for text-based analysis, which are then transformed into CSV files. The CSV files are then sent to our system's pre-processing module. The best accurate model for Sybil attack detection when the factor energy is taken into consideration is shown by a comparative examination of the suggested framework. However, the planned work's contribution is outlined as follows:

i. Data gathering is carried out, and parameters are set;
ii. Data mapping depending on either the Normal or Sybil class;
iii. Sybil attacks are classified to ftp_write, guess_password, and others;
iv. The gathered dataset is subjected to 15 distinct Sybil assaults using the suggested machine learning models;
v. The experiment analysis is completed to evaluate the efficiency with which our proposed model uses energy (computational power) and security (detection) in terms of safety.

The remainder of this article is structured as follows. Various kinds of problems caused by Sybil attacks along with prevention mechanisms are discussed in Section 2. Related work is discussed in Section 3. The proposed framework is explained in Section 4. Experiments and results are given in Section 5, while the conclusions and future work are discussed in Section 6.

2. Sybil Attacks Problems and Prevention

2.1 User Exclusion from the Network

When a Sybil assault occurs, attackers with enough identities formed might outvote trustworthy nodes and refuse to send or accept blocks.

2.2 Assault at a Fifty-one Percent Intensity

A Sybil attack is when one adversary controls at least fifty-one percent of the computing power or hash rate of a network. This sort of attack puts a blockchain network's stability in danger and might result in data loss. A 51 percent attack can change the sequence of transactions, permit unauthorized duplication of expenditure, or even stop transactions from being performed entirely.

2.3 Sybil Attack Prevention

2.3.1 Verification of Identity

By exposing the real identities of malicious actors, identity verification may prevent Sybil attacks. The basis for validation is a centralized authority that confirms the validity of network entities and can do reverse lookups. It is possible to carry out identity verification directly or indirectly:

i. Direct validation occurs when a local entity requests that a central authority confirm the identity of other, distant entities;
ii. Based on the vouching of other network nodes, indirect validation means that the local entity "vouchs" for the identification of a remote entity.

2.3.2 Graphs Related to Social Trust

Analyzing social graph link information may help to stop attacks carried out by Sybil. This will enable anonymity and reduce the potential impact of a single Sybil attacker.

2.3.3 Expenses in Monetary Terms

Financial investments may operate as false gatekeepers, which will significantly increase the cost of a Sybil attack. For example, contemporary cryptocurrencies use the Proof of Work (PoW) method, which necessitates financial inputs on resources like stake or storage.

2.3.4 Proof Identification

A "one entity per person" restriction and identity verification may be mandated by P2P networks. It is conceivable to employ a tactic that does not require the validating authority to be aware of the participants' real identities. Users might, for instance, show up at a predetermined time and location to prove their identification.

2.3.5 Protection of Application

There have been many different distributed protocols established, each with its built-in protection against Sybil attacks. These include:

i. SumUp and DSybil are Sybil-proof online content voting and recommendation systems;
ii. Whanau is an algorithm for distributed hash tables that includes Sybil protection built right in;
iii. Kademlia – The I2P implementation of this protocol may reduce the number of Sybil attacks.

3. Related Work

The detection of Sybil attacks is an intriguing subject for researchers, and many studies that conduct Sybil attack detection have been published in scholarly literature. To produce an accurate and high-performance solution, Revathi and Malathi [5] used a methodology that was based on five distinct machine-learning algorithms. These algorithms were Random Forest (RF), Naive Bayes, Support Vector Machine (SVM), and CART. After applying this approach to the NSL-KDD dataset, which contains 41 features, it achieved excellent detection accuracy in recognizing attacks and anomaly detection, with the RF algorithm obtaining a high accuracy of 97 %. It was suggested by Tang *et al.* [6] that a hybrid deep learning approach should be used to improve accuracy and arrive at a strategy that is both superior and more helpful. These methodologies make use of two distinct types of deep learning classifiers in their decision-making. The authors were able to construct a hybrid method that was based on the NSL-KDD dataset and used six characteristics to train the classifier. This hybrid method was created by merging a Gated Recurrent Unit (GRU) with a Recurrent Neural Network (RNN), which is referred to as a GRU-RNN. The hybrid approach method established its superiority in the SDN working environment by achieving an accuracy increase of 89 % when compared to the earlier strategy.

In [7], Kurochkin and Volkov outlined a plan for the development of IDS for SDN. The researchers used both machine learning and deep learning methods to accomplish the task of comparing the results. A deep learning algorithm known as GRU was used to distinguish the six unique types of attacks that were carried out using an appropriate strategy. The method beat machine learning classifiers in terms of accuracy and performance, and many types of datasets were used for training and comparing the results of the method. The method was shown to be highly successful, revealing the possibility for a very effective application of deep learning.

In the framework of software-defined networking (SDN), Hadi *et al.* [8] introduced an approach known as the Network Intrusion Detection System-Deep Learning module (NIDS-DL). Network Intrusion Detection Systems (NIDSs) were integrated into their methodology in a manner that made use of several different deep learning strategies. To extract 12 features from the NSL-KDD dataset, which contains a total of 41 features, their method made use of a technique for selecting features. The following classifiers were utilized: CNN, DNN, RNN, LSTM, and GRU. When they examined the scores of the various classifiers, the technique produced accuracy results of 98.63 %, 98.5 %, 98.1 %, 98 %, and 97.78 %, respectively.

A variety of techniques for identifying different kinds of network dangers were detailed in [9]. However, VANET is still susceptible to a variety of attacks, the Sybil attack being the most notable of them. The Sybil attack is one of the most challenging types of attacks used against VANETs because it creates fake identities inside the network to disrupt communication between network nodes. This attack has a substantial impact on the safety services provided by transport providers and has the potential to create traffic congestion. An original collaborative architecture that is run on majority voting has been built to detect the Sybil attack that has been launched against the network. The functionality of the system is achieved by the parallelization of many classifiers, including K-nearest Neighbour, Nave Bayes, Decision Tree, Support Vector Machine, and Logistic Regression. The approach of majority voting, which includes both hard and soft options, is utilized to provide a final prediction. A comparison of majority voting hard and majority voting soft is carried out so that the best approach can be identified. The approach that was advised is accurate 95 % of the time.

Murali [10] suggested a novel artificial bee colony (ABC)-inspired mobile Sybil attack models and lightweight intrusion detection approach for mobile RPL Sybil assaults. Murali and Jamalipour [11], proposed a one-of-a-kind parent selection approach for mobility in RPL on the mobility-aware parent selection algorithm for low-power and lossy networks, together with the Dynamic Trickle (D-Trickle) to cut down on the number of control overhead operations. This was done to lower the amount of control overhead operations.

Mishra *et al.* [12] published the general analytical model for the Sybil attack in the IoT. However, in developing the Sybil attack, it is very challenging to modify it such that it may be used against low-power RPL nodes. They chose a trust-based strategy for recognizing and isolating security assaults like the rank as well as the Sybil attack, which was recommended by Airehrour et al. [13] for IoT, and the SecTrust-RPL was proposed for use with it. In this case, they considered static RPL rather than mobile RPL to raise the level of security within a 6LoWPAN network and to increase the level of security inside of 6LoWPAN networks. SVELTE is an intrusion detection system for the Internet of Things that makes use of the expected transmission count (ETX) metric. This system was presented by Shreenivas *et al.* [14].

Deshmukh-Bhosale and Sonevane [15] used an intrusion detection system to identify wormhole attackers and attacks on wormholes. They relied only on the received signal strength as a parameter for determining which nodes were malicious to accomplish their task. The kind of intrusion detection system (IDS) that is now being used is a hybrid system that has both centralized and distributed modules. Centralized and distributed modules were responsible for performing attack detection. This technique, which is used in the Cooja simulator that is part of the Contiki operating system, had a success rate of 90 %. Singhal *et al.* [16] were able to successfully carry out Sybil attack detection by making use of the software Cooja simulator and the operating system Contiki. The researchers proposed defensive methods including the Compare and Match (CAM) approach for detecting purposes to validate the Sybil attack posture and prevent it from occurring.

Table 1 shows that there were several methods in the literature for detecting Sybil attacks that took both security and energy into account. As it can be easily understood, the existing method is limited in terms of detection rate and energy (power) consumption. If some method considers the energy factor, then it will be limited in terms of accuracy. It means that the existing methods focused on a single factor at a time, whereas our proposed method focuses on both factors and performs tremendously as compared to other state-of-the-art methods.

Table 1
Overview of existing methods

Authors	Approach	Remarks
Revathi and Malathi [5]	**RF, Naive Bayes, SVM, CART, J84**	**Limitation in terms of time consumption, Achieved accuracy 97 %**
Tang *et al.* [6]	**GRU-RNN**	**Limited in terms of accuracy which was 89 %**
Hadi *et al.* [8]	**CNN, DNN, RNN, LSTM, GRU**	**Achieved accuracy 97.7-98.6 %**
Azam *et al.* [9]	**K-Nearest Neighbor, Nave Bayes, Decision Tree, SVM, Logistic Regression**	**Achieved accuracy 95 %**
Murali [10]	**Artificial Bee Colony**	**Achieved accuracy 95 %**
Murali and Jamalipour [11]	**Mobility-aware Parent Selection Algorithm, Dynamic Trickle**	**90 % of nodes are kept alive with a mobility of 50 %, but only 50 % are kept alive at a mobility of 100 %**
Mishra *et al.* [12]	**SecTrust-RPL**	**SecTrust-RPL needed to be improved in terms of trusted node integration.**
Airehrour *et al.* [13]	**K-mean clustering technique**	**Achieved accuracy 48.7 %**
Deshmukh-Bhosale and Sonavane [15]	**Hybrid IDS using Cooja and Contiki**	**Achieved accuracy 95 %**
Singhal *et al.* [16]	**Compare and Match approach**	**There was a need to enhance the detection rate.**

4. Proposed Methodology

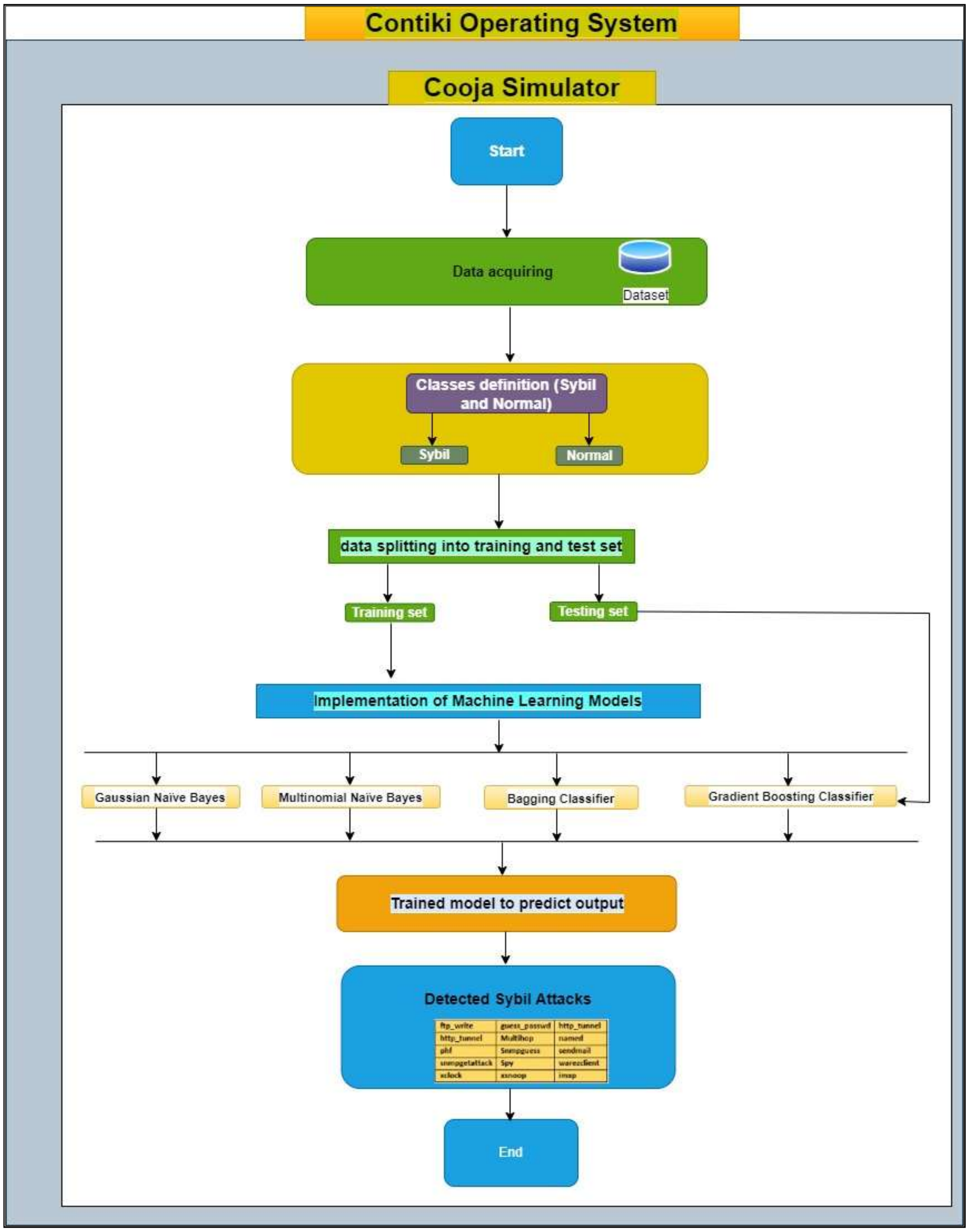
Figure 1 shows the proposed architecture. The proposed work consists of the following modules:

i. Data acquisition is performed and parameters defined;
ii. Mapping the data based on both classes;

iii. Classification of Sybil attacks is performed;

iv. The proposed machine learning models are used to perform different Sybil attacks on the acquired dataset;

v. The experiment analysis is done so that we can judge how well our proposed model works in terms of security (detection) and energy (computational power).



ftp_write	guess_passwd	http_tunnel
http_tunnel	Multihop	named
phf	Snmpguess	sendmail
snmpgetattack	Spy	warezclient
xclock	xsnoop	imap

Fig. 1. Proposed architecture diagram

As a direct consequence of this, it is easier to accurately evaluate various learning strategies. It is feasible to conduct the experiments on the whole set if there is a sufficient number of records in both the train set and the test set. This eliminates the need to randomly choose a portion of the data. As a consequence, the outcomes of the evaluations taken from the many research projects will be comparable and consistent. Because there were so few IoT attack statistics that were readily accessible to the public, we utilized data from simulations that were designed to replicate real-world occurrences. To use the raw packet capture files that are generated by the Cooja simulation, text-based techniques will first need to have the CSV files translated before they can be utilized. After that, the CSV files are sent to the pre-processing module of our system so that they may be processed. On the other side, the dataset is received from a source on the internet and is then translated using Contiki-Cooja.

4.1 Mapping of Data Based on Classes

Following the acquisition of the dataset, mapping will be carried out. In this stage of the process, we define two nodes, Sybil (1) and Normal (0), to determine which nodes are malicious. The Sybil node represents a Sybil attack, while the Normal node represents a node that is not vulnerable to such an assault.

4.2 Sybil Attacks Classifications

Similarly, the number of Sybil assaults is determined during the categorization step. The categorization is based on 15 distinct types of Sybil attacks, including ftp_write, guess_passwd, http_tunnel, imap, multihop, named, phf, sendmail, snmpgetattack, snmpguess, spy, warezclient, warezmaster, xclock, and xsnoop, among others.

4.3 Implementation of Machine Learning Models

After the data gathering step is completed, the next step is to execute the preprocessing of the data. Now that the data are available, the models may be trained on them as well as tested on them. During this phase, we added one more step, which was the categorization of assaults based on protocol, such as Internet Control Message Protocol (ICMP), Transmission Control Protocol (TCP), and User Datagram Protocol (UDP).

Figure 2 depicts attacks that are protocol-based. Following the completion of the classification step, the machine learning process is then applied to the dataset to identify Sybil assaults.

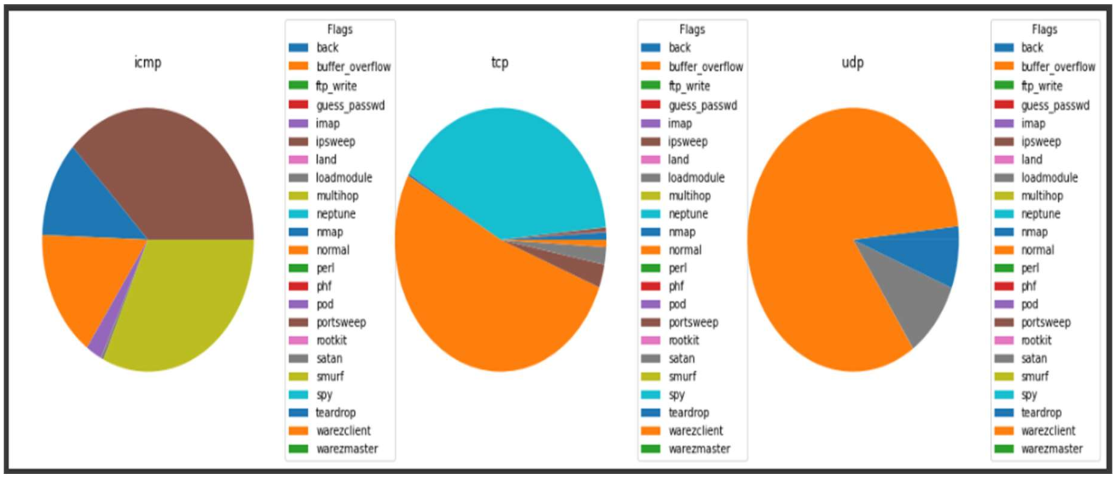


Fig. 2. Statistics of attacks based on protocols

4.4 Gaussian Naïve Bayes

The Bayes theorem is the foundation of the Naive Gaussian Bayes classification method, which is a probabilistic approach to grouping data that follows stringent independence guidelines. When discussing classification, the concept of "independence" relates to the idea that the presence of one value of a characteristic does not influence the presence of another value of the characteristic. The notion that the qualities of an object have no relation to one another is what's meant to be understood by the term "naive," which is an adjective. It is common knowledge in the field of machine learning that Naive Bayes classifiers are exceptionally expressive, scalable, and accurate to a satisfactory degree. Despite this, their performance improves along with the amount of data used for training. The efficiency of Naive Bayes classifiers may be traced back to a wide range of distinguishing characteristics and factors. They can readily deal with continuous features, scale well with the size of the training data set, and do not need any parameter adjustment for the classification model [17–18]. The formula developed by Bayes, which is used in several machine learning methods, may be seen below:

$$\boldsymbol{GP(x|y) = \frac{GP(x \cap y)}{GP(y)} = \frac{GP(x).GP(x|y)}{GP(y)}}, \tag{1}$$

whereas *GP(x)* denotes the likelihood of *x*, and *GP(y)* denotes the likelihood of *y*, *GP(x|y)* represents the probability of *x* provided by *y*, *GP(y|x)* represents the probability of *y* given by *x*. The probability of both *x* and *y* is denoted by *GP(x∩y)*.

The Gaussian Naive Bayes formula is similar:

$$\boldsymbol{GP(m_i|v) \ = \frac{1}{\sqrt{2\pi\sigma^2}} \, exp\left(\frac{(m_i - \mu_v)^2}{2\pi\sigma_v^2} \right)}, \tag{2}$$

where the variance may be independent of both *V* and *m_i*, just one of them, or both. The Gaussian Naive Bayes technique, which also models basic continuous-valued features as existing independently and following a Gaussian (normal) distribution, provides support for this approach.

4.5 Multinomial Naive Bayes

As a Bayesian learning methodology, the Multinomial Naive Bayes (MNB) method is often used in natural language processing. It determines the probabilities of each tag for a certain sample and produces the tag with the highest likelihood. Each feature is classified separately from all other characteristics using the Naive Bayes classifier. This classifier is made up of several algorithms. The inclusion or exclusion of one feature does not affect the inclusion or exclusion of another feature. Since probability calculations are all that are required, implementation is simple. This approach applies to both continuous and discrete data. Real-time application forecasting is possible with this easy method. Large datasets are no problem because of its great scalability [19]. It may be assessed using the following formula, which is:

$$\boldsymbol{GP(x|y) = \frac{GP(x).GP(x|y)}{GP(y)}}, \tag{3}$$

$$\boldsymbol{GP(x|y) = GP(y_1|x) \times GP(y_2|x) \times \cdots \times GP(y_n|x) \times GP(x)}, \tag{4}$$

4.6 Gradient Boosting Classifier

We must complete several stages to create the Gradient Boosting Classifier (GBC):

i. Satisfy the model;
ii. Adapt the model's hyperparameters and parameters;
iii. Construct predictions;
iv. Describe the findings.

With Scikit-Learn, fitting models is rather simple since after building up the model, we often simply need to execute the fit() function. However, we need to actively choose how to adjust the model's hyper-parameters. We may adjust several arguments and hyper-parameters to attempt to improve the model's accuracy. Changing the model's learning rate is one method we may do this. When making predictions, we should pick the optimal learning rate after evaluating the model's performance on the training set at various learning rates. In Scikit-Learn, predictions may be produced. Utilize the predict() method once the classifier has been fitted to very easily learn. You should make predictions based on the testing dataset's attributes and then contrast your results with the actual labels. Checking a classifier's accuracy is often the first step in assessing it, after which the model's parameters and hyper-parameters are adjusted until the user is pleased with the classifier's accuracy [20].

4.7 Bagging Classifier

A kind of ensemble learning known as bagging (or Bootstrap aggregating) involves the independent concurrent training of many base models on various subsets of the training data. Using bootstrap sampling, each subset is created by randomly selecting and replacing data points. For the Bagging classifier, the final prediction is created by combining the all-base model predictions and utilizing majority voting. Regression involves averaging all-base model predictions to arrive at the final prediction; this process is known as bagging regression. The key benefit of bagging is that it may lower the variance of predictions provided by a supervised learning system without affecting the accuracy of such predictions. Since it enables us to trade off some accuracy for improved resilience, it is an appealing solution for issues where the penalty of making a mistake is significant (such as in medical diagnosis or credit card fraud detection) [21–22].

4.8 Proposed Algorithm

Algorithm – GNB, MNB, GBC, BC.
Input – NSL-KDD dataset.
Step 1 – Dataset loading.
Step 2 – Mapping and preprocessing the data.
Step 3 – Categorization of attacks based on protocol ICMP, UDP, TCP.
Step 4 – Applying the machine learning model to the dataset.
Step 5 – Calculate accuracy.

5. Experiments and Results

5.1 Experiments

All of the testing was conducted on a PC with an Intel Core i5-6200U CPU running at 2.4 GHz and 8 GB of RAM. However, the Contiki operating system was implemented and Cooja-Contiki was utilized as a simulator. Python is used as a programming language as well. Figure 3 describes how the complete architecture functions.

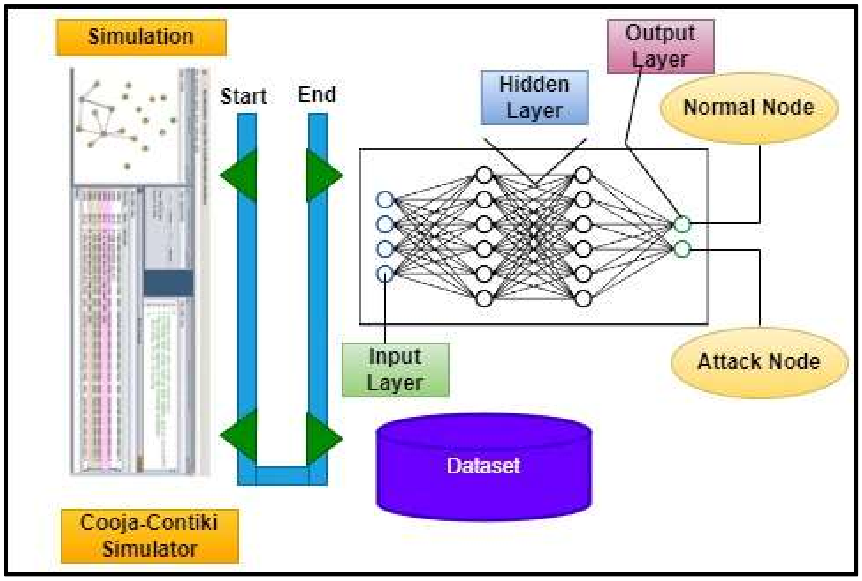


Fig. 3. Working of the entire framework using Cooja-Contiki

Additionally, an experimental study is performed using an NSL-KDD dataset. Table 2 below discusses the statistics of the training and testing sets.

Table 2
NSL-KDD training and testing set statistics

Training set			
	Original data	**Distinct data**	**Rate of reduction (%)**
Attacks	**3,925,650**	**262,178**	**93.32**
Normal	**972,781**	**812,814**	**16.44**
Total	**4,898,431**	**1,074,992**	**78.05**
Testing set			
	Original data	**Distinct data**	**Rate of reduction (%)**
Attacks	**250,436**	**29,378**	**88.26**
Normal	**60,591**	**47,911**	**20.92**
Total	**311,027**	**77,289**	**75.15**

The recall, accuracy, and precision of the suggested framework are used to evaluate its performance. For certain evaluation parameters, the following equations offer mathematical formulae:

$$\mathbf{Precision}(\boldsymbol{P}) = \frac{\boldsymbol{TP}}{\boldsymbol{TP}} + \boldsymbol{FP}, \tag{5}$$

$$\textbf{Recall}(\boldsymbol{R}) = \frac{\boldsymbol{TP}}{\boldsymbol{TP}} + \boldsymbol{FN}, \tag{6}$$

$$\textbf{Accuracy} = \boldsymbol{TP} + \frac{\boldsymbol{TN}}{\boldsymbol{TP}} + \boldsymbol{FP} + \boldsymbol{TN} + \boldsymbol{FN}, \tag{7}$$

5.2 Results

After training or testing the model on the NSL-KDD dataset, the proposed model achieves a tremendous result as compared to other state-of-the-art methods. All the models performed better in terms of detection and consumed less power (energy). However, there are neither hardware dependencies nor any specialized computational requirements. In the proposed framework, the BC performed much better and achieved higher accuracy (99.3 %) as compared to the GNB (53.8 %), MNB (80.5 %), and GBC (99 %). Figure 4 shows the confusion matrix of the proposed framework. Similarly, the accuracy is described in Figure 5.

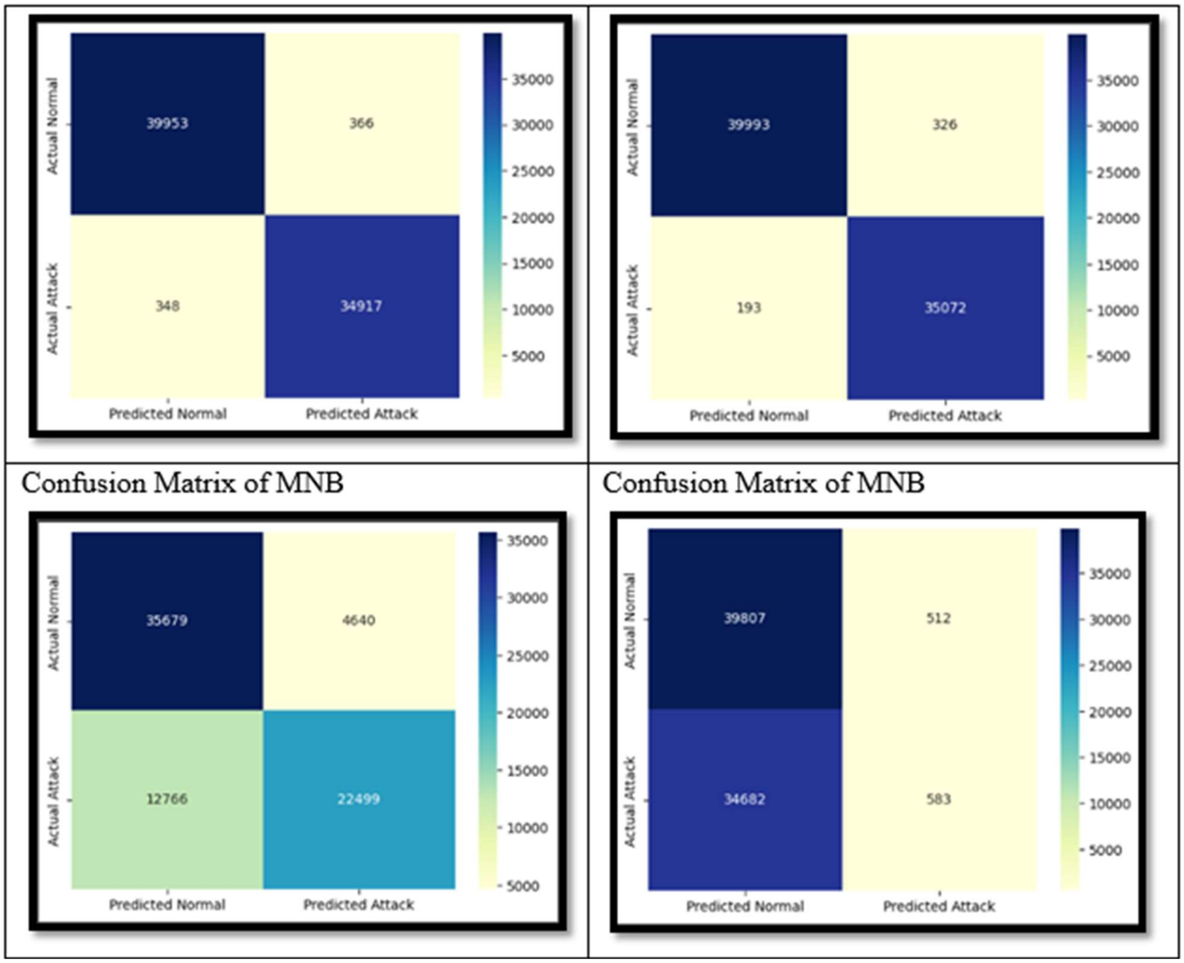


Fig. 4. Confusion matrix of GBC, BC, MNB, and GNB

5.3 Comparative Analysis with the Existing Methods

The literature describes several techniques for identifying Sybil attacks or intrusions. This section does a comparative study by taking into account two factors:

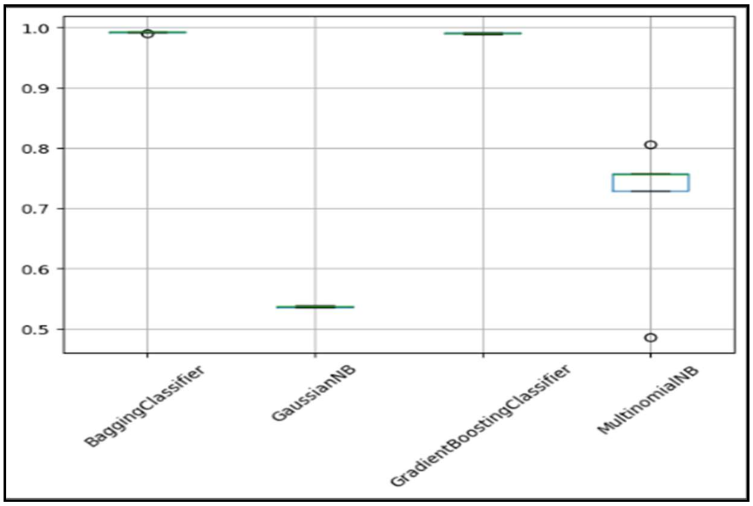


Fig. 5. Accuracy of GBC, BC, MNB, and GNB

i. (F_1) *Energy consumption/mobility*;
ii. (F_2) *Accuracy/detection rate/security*.

The number of researchers who have worked on the suggested domain is shown in Table 3. However, the current methods have not concentrated on both elements, or if they had, they did it in an unreliable manner. On the other hand, our framework, which emphasized both elements, led to superior outcomes. The proposed research is also straightforward, simple to comprehend, and very power-efficient (i.e., no hardware requirement).

Table 3
Comparative analysis with the existing state-of-the-art methods

References	***F₁***	***F₂***
Revathi and Malathi [5]	**X**	**97 % accuracy**
Tang *et al.* [6]	**X**	**89 % accuracy**
Hadi *et al.* [8]	**X**	**97.7–98.6 % accuracy**
Azam *et al.* [9]	**X**	**95 % accuracy**
Murali [10]	**X**	**95 % accuracy**
Murali and Jamalipour [11]	**✓**	**90 % accuracy**
Airehrour *et al.* [13]	**X**	**48.7 % accuracy**
Deshmukh-Bhosale and Sonavane[15]	**X**	**95% accuracy**
Proposed model	**✓**	**99.3% accuracy**

5. Conclusion and Future Work

Sybil attack detection is discussed in this research. The proposed design is straightforward, simple to comprehend, and very power-efficient. Three separate models are used for detection: GBC, BC, MNB, and GNB. Specifically, the NSL-KDD dataset is employed for experimental analysis. Following the experimental study, it was discovered that BC outperformed GNB, GBC, and MNB in terms of detection, with a 99.3 % accuracy. However, although 15 various forms of Sybil assaults are addressed, the whole system concentrates on two key factors, namely security and energy.

Furthermore, to make sure that our proposed research was very effective while taking into account both of the aforementioned variables, a comparison study was also done with current state-of-the-art methodologies. This study will be expanded in the future to focus on additional Sybil assaults and to increase the rate of detection.

Acknowledgment

This research was not funded by any grant.

References

[1] Saraswathi, R. V., Sree, L. P., & Anuradha, K. (2016). Dynamic and probabilistic key management for distributed wireless sensor networks. In *2016 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC),* 1-6. IEEE. https://doi.org/10.1109/ICCIC.2016.7919666.

[2] Medjek, F., Tandjaoui, D., Romdhani, I., & Djedjig, N. (2017). Performance evaluation of RPL protocol under mobile sybil attacks. In *2017 IEEE Trustcom/BigDataSE/ICESS*, 049-1055. IEEE. https://doi.org/10.1109/Trustcom/BigDataSE/ICESS.2017.351.

[3] Rahbari, M., & Jamali, M. A. J. (2011). Efficient detection of Sybil attack based on cryptography in VANET. *International Journal of Network Security & its Applications*, *3*(6). https://doi.org/10.48550/arXiv.1112.2257.

[4] NSL-KDD Dataset [Online]. https://www.unb.ca/cic/datasets/nsl.html.

[5] Revathi, S., & Malathi, A. (2013). A detailed analysis on NSL-KDD dataset using various machine learning techniques for intrusion detection. *International Journal of Engineering Research & Technology*, *2*(12), 1848-1853.

[6] Tang, T. A., Mhamdi, L., McLernon, D., Zaidi, S. A. R., & Ghogho, M. (2018). Deep recurrent neural network for intrusion detection in sdn-based networks. In *2018 4th IEEE Conference on Network Softwarization and Workshops (NetSoft)* (pp. 202-206). IEEE. https://doi.org/10.1109/NETSOFT.2018.8460090.

[7] Kurochkin, I. I., & Volkov, S. S. (2020). Using GRU based deep neural network for intrusion detection in software-defined networks. In *IOP Conference Series: Materials Science and Engineering*, 927(1), 012035. IOP Publishing. https://doi.org/10.1088/1757-899X/927/1/012035.

[8] Hadi, M. R., & Mohammed, A. S. (2022). A novel approach to network intrusion detection system using deep learning for Sdn: Futuristic approach. *arXiv preprint arXiv:2208.02094.* https://doi.org/10.5121/csit.2022.121106.

[9] Azam, S., Bibi, M., Riaz, R., Rizvi, S. S., & Kwon, S. J. (2022). Collaborative learning based Sybil attack detection in Vehicular AD-HOC Networks (VANETS). *Sensors*, *22*(18), 6934. https://doi.org/10.3390/s22186934.

[10] Murali, S., & Jamalipour, A. (2019). A lightweight intrusion detection for sybil attack under mobile RPL in the internet of things. *IEEE Internet of Things Journal*, *7*(1), 379-388. https://doi.org/10.1109/JIOT.2019.2948149.

[11] Murali, S., & Jamalipour, A. (2018). Mobility-aware energy-efficient parent selection algorithm for low power and lossy networks. *IEEE Internet of Things Journal*, *6*(2), 2593-2601. https://doi.org/10.1109/JIOT.2018.2872443.

[12] Mishra, A. K., Tripathy, A. K., Puthal, D., & Yang, L. T. (2018). Analytical model for Sybil attack phases in internet of things. *IEEE Internet of Things Journal, 6*(1), 379-387. https://doi.org/10.1109/JIOT.2018.2843769.

[13] Airehrour, D., Gutierrez, J. A., & Ray, S. K. (2019). SecTrust-RPL: A secure trust-aware RPL routing protocol for Internet of Things. *Future Generation Computer Systems, 93*, 860-876. https://doi.org/10.1016/j.future.2018.03.021.

[14] Shreenivas, D., Raza, S., & Voigt, T. (2017). Intrusion detection in the RPLconnected 6LoWPAN networks. *In Proc. 3rd ACM Int. Workshop IoT Privacy Trust Security (IoTPTS)*, Abu Dhabi, UAE, 2017, pp. 31-38.

[15] Deshmukh-Bhosale, S., & Sonavane, S. S. (2019). A real-time intrusion detection system for wormhole attack in the RPL based Internet of Things. *Procedia Manufacturing, 32*, 840-847. https://doi.org/10.1016/j.promfg.2019.02.292.

[16] Singhal, P., Sharma, P., & Arora, D. (2018). An approach towards preventing IoT based Sybil attack based on contiki framework through cooja simulator. *International Journal of Engineering & Technology, 7*(2.8), 261-267.

[17] Rohan, V. (2022). Gaussian Naive Bayes: What You Need to Know?

[18] Gaussian Naive Bayes. https://iq.opengenus.org/gaussian-naive-bayes/.

[19] Shriram. (2023). Multinomial Naive Bayes Explained: Function, Advantages & Disadvantages, Applications in 2023.

[20] Dan, N. (2023). Gradient Boosting Classifiers in Python with Scikit-Learn.

[21] Debomit, D. (2023). Bagging classifier.

[22] Moamen, E. (2022). What is Bagging classifier.