


Article

A Hierarchical Deep Learning-Based Intrusion Detection Architecture for Clustered Internet of Things

Rania Elsayed¹, Reem Hamada¹, Mohammad Hammoudeh^{2,*} , Mahmoud Abdalla¹
and Shaimaa Ahmed Elsaid¹

¹ Electronics and Communications Engineering, Zagazig University, Zagazig 44519, Egypt

² Information & Computer Science, King Fahd University of Petroleum and Minerals, Dhahran 31261, Saudi Arabia

* Correspondence: m.hammoudeh@kfupm.edu.sa

Abstract: The Internet of Things (IoT) system's ever-expanding attack surface calls for a new intrusion detection system (IDS). These systems may include thousands of wireless devices that need to be protected from cyberattacks. Recent research efforts used machine learning to analyze and identify various attacks and abnormal behavior on IoT systems. Most of these techniques are characterized by low accuracy and they do not scale to today's IoT-enabled smart cities applications. This article proposes a secure automatic two-levels intrusion detection system (SATIDS) which utilizes the minimum redundancy maximum relevance (MRMR) feature selection technique and an enhanced version of long short-term memory (LSTM) based on an artificial recurrent neural network (RNN) to enhance the IDS performance. SATIDS aims at detecting traffic anomalies with greater accuracy while also reducing the time it takes to perform this task. The proposed algorithm was trained and evaluated using two of the most recent datasets based on realistic data: ToN-IoT and InSDN datasets. The performance analysis of the proposed system proves that it can differentiate between attacks and normal traffic, identify the attack category, and finally define the type of sub-attack with high accuracy. Comparing the performance of the proposed system with the existing IDSs reveals that it outperforms its best rivals from the literature in detecting many types of attacks. It improves accuracy, detection rates, F1-score, and precision. Using 500 hidden and two LSTM layers achieves accuracy of 97.5%, precision of 98.4%, detection rate of 97.9%, and F1-score of 98.05% on ToN-IoT dataset, and precision of 99%, detection rate of 99.6%, and F1-score of 99.3% on InSDN dataset. Finally, SATIDS was applied to an IoT network which utilizes the energy harvesting real-time routing protocol (EHRT). EHRT optimizes the low-energy adaptive clustering hierarchy (LEACH) routing technique using a modified artificial fish swarm algorithm. The integration between the optimized LEACH and the proposed IDS enhances the network lifetime, energy consumption, and security.

Keywords: intrusion detection system (IDS); internet of things (IoT); deep learning (DL); long short-term memory (LSTM); energy harvesting real-time routing protocol (EHRT); low-energy adaptive clustering hierarchy (LEACH); artificial fish swarm algorithm (AFSA); ToN-IoT dataset; InSDN dataset



Citation: Elsayed, R.; Hamada, R.; Hammoudeh, M.; Abdalla, M.; Elsaid, S.A. A Hierarchical Deep Learning-Based Intrusion Detection Architecture for Clustered Internet of Things. *J. Sens. Actuator Netw.* **2023**, *12*, 3. <https://doi.org/10.3390/jsan12010003>

Academic Editor: Mingjun Xiao

Received: 7 December 2022

Revised: 20 December 2022

Accepted: 23 December 2022

Published: 28 December 2022



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

IoT is a powerful platform for connecting everyday objects and users worldwide without human interference. It links devices with the Internet enabling smart identification, tracing, positioning, administration, and monitoring of objects and people. It has several uses including healthcare, transport, and smart grids. Despite its many advantages, IoT networks are susceptible to cyber-attacks which affect their operation, stability, and energy consumption. To address some of the IoT security issues, an intrusion detection system (IDS) is used to detect traffic abnormalities, protect the network from attacks, and reduce functional and financial losses by alerting the network administrator about those behaviors. An IDS system may use signature rules, anomaly-based rules, or a combination of these to increase its intrusion detection performance [1–3].

In the last decade, IDSs started to utilize machine learning (ML) or deep neural network (DNN) techniques to increase their attack-detection efficiency. The more extensive DNN networks are called deep learning (DL), which consider an ML border subfield. DL contains many hidden layers to solve many of the complicated problems in various applications. The input of each layer is the output of the previous layer. Although DL results in high performance, its main limitation is a longer training time for more essential training data. Convolution neural network (CNN) and recurrent neural networks (RNN) are two popular types of DL models. CNN is used in computer image processing and language processing while RNN is used in the processing of natural language and text processing. Long short-term memory (LSTM) network is considered an extension of the RNN network which is capable of performing the line sequences learning pattern. LSTM can be used to identify knowledge as normal traffic or attack. The primary benefit of this form of DL is that the raw data is used directly [3–5].

1.1. Problem Statement

Based on previous research, we conclude that IoT applications face numerous security-related concerns and obstacles. With the proliferation of cyberattacks in the IoT environment, the entire system may become susceptible to assault. The primary obstacles are outlined below:

- Due to the dynamic nature of IoT networks such as IoT devices, fog, and cloud, designing a distributed security framework for distributed IoT applications is a difficult issue. Additionally, the growth of attacker behavior can intercept the IoT transmission network.
- Distributed analysis of the massive amounts of data (big data) produced by IoT devices is a complex problem that requires a security mechanism IoT that uses the right analytical tool in a distributed working architecture.
- As datasets grow in size and complexity, maintaining network security becomes increasingly difficult due to the increasing computational complexity involved in measuring the dataset parameters. There are fewer intrusions than non-intrusions, making it more challenging to train for the latter [6,7].
- Most IoT devices are powered by batteries. Energy is consumed during data collection, processing, and packet transmission or reception, hence an intrusion deduction system that limits overall network energy consumption is necessary [8–10].

1.2. Our Contribution

The main contribution of this research is summarized as follows:

- An efficient IDS which utilizes minimum redundancy maximum relevance (MRMR) feature selection technique and an enhanced version of long short-term memory (LSTM) based on an artificial recurrent neural network (RNN) is proposed. The proposed IDS system performs well in terms of detecting traffic anomalies with greater accuracy while also reducing the time it takes to perform this task. Minimum redundancy maximum relevance (MRMR) feature selection method is exploited to reduce the dataset dimension and size by selecting only the most essential features. It helps training IDSs more quickly with low computation and high accuracy because it removes irrelevant features to maximize system accuracy.
- The performance of the proposed IDS system was analyzed using two of the most recent datasets based on realistic data: ToN-IoT and InSDN datasets [11,12].
- The proposed MRMR-SATIDS was applied to a cluster based IoT network which uses an optimized hierarchical routing technique to address the energy consumption issue and conserve devices energy.
- Finally, the performance of the proposed system was compared with and without using an MRMR feature selection technique, then was compared to other ML IDSs.

The remainder of the paper is structured as follows: Contributions of other researchers to the IDS field are presented in Section 2. Section 3 describes the methods and methodology used in this work. The proposed system is presented in Section 4. Performance analysis

of the proposed system and a comparison to existing systems are presented in Section 5. Section 6 concludes the article and presents future work avenues.

2. Related Work

Deep learning (DL) helps researchers and cyber analysts gain insights into threats and stay ahead of opponents. Attack detection continues to be an active area of research that receives considerable attention due to the wide attack vector and the possible impact of attacks on IoT systems. Many research efforts have been carried out to detect network attacks using DLs. Using network traffic analysis, an SDN-based intrusion detection system that is executed as an application module on the controller was developed in [3]. In [4], The Bot-IoT dataset was used to train DRNN and SMOTE-DRNN models, and the simulation results show that high-class imbalance in the training data has a negative impact on the DRNN model's precision, recall, F1 score, area under the receiver operating characteristic curve (AUC), geometric mean (GM), and Matthew's correlation coefficient (MCC). In [5], the authors had recommended adopting Blockchain to protect FL algorithms in Internet of Things (IoT) devices from attacks. Integrating Blockchain and FL ensures the integrity of trained models, preventing model poisoning attacks. In [6], the authors had suggested a clustering method based on unsupervised feature selection and cluster center initialization for intrusion detection. This method computes initial cluster centers using sets of semi-identical examples that signal dense data space and avoids outliers. A model was created in [7] that extracts important features from the given features and then classifies incursions using a deep learning approach. Notably, underlying data points cannot be considered samples from a single distribution, but rather from two distinct distributions—one generic to all network intrusions and the other domain-specific. A deep learning-based deep belief network (DBN) algorithm was advocated in [8] for intrusion detection systems. The survey presented in [9] offers a thorough analysis of LEACH descendant clustering procedures. This is the first study to categorize LEACH-based routing protocols into CH selection, data transmission, and combined CH selection and data transmission methods.

The authors in [10] suggested a cyberattack detection framework based on ensemble learning and fog-cloud architecture for IoMT networks. The ensemble design incorporates the individual learners: decision tree, naive Bayes, and random forest. Training the models with just examples of normal classes, a new method is presented in [13] based on long short-term memory (LSTM) autoencoder and one-class support vector machine (OC-SVM) to detect anomaly-based assaults in an unbalanced dataset. Using a variety of machine learning algorithms and ensemble learning, [14] compares several of these intrusion detection systems. Using a random forest regressor technique, a subset of attributes (significant) is selected from the primitive set of attributes, and this subset is then utilized to train different classifiers. Table 1 shows the most significant existing IDS systems.

Table 1. Existing IDS approaches.

Ref.	Technique	Dataset	Accuracy
[3]	An intrusion detection system based on the software defined networks model is presented which is executed as an application module in the controller	NSL-KDD CICIDS2017	-
[4]	Develop DRNN and SMOTE-DRNN models	Bot-IoT dataset	99.87%
[6]	A clustering method based on unsupervised component selection and initialization of the cluster center	KDD CICIDS2017 Wormhole	93.07% 88% 94.06%
[7]	CNN technique with L2 regularization and the dropout methods to address the overfitting problem	InSDN	93.01%

Table 1. Cont.

Ref.	Technique	Dataset	Accuracy
[8]	Trustworthy privacy-preserving secure framework for intelligent cities is presented using three modules: a trustworthiness module, a two-level privacy module, and an intrusion detection module	ToN-IoT BoT-IoT	98.84% 99.99%
[10]	Ensemble learning and fog-cloud architecture-driven cyberattack detection framework for IoMT networks	ToN-IoT	96.35%
[13]	Hyper approach based on LSTM autoencoder and one-class support vector machine to detect anomaly based attacks in an unbalanced dataset	InSDN	90.5%
[14]	Secured privacy-preserving framework (SP2F) for smart agricultural UAV	ToN-IoT IoT Botnet	99.77% 99.98%
[15]	CNN technique with L2 regularization and the dropout methods to address the overfitting problem	InSDN	90.5%
[16]	Deep learning model for bot detection that uses a novel cascade forward back propagation neural network model with a subset of features using the correlation-based feature selection (CFS) technique	NF-UNSW-NB15 NFToN-IoT NF-BoT-IoT NF-CSE-CIC IDS2018 ToN-IoT-Windows	-
[17]	A machine learning based block chain method is proposed for protecting IoT network	ToN-IoT BoT-IoT	-
[18]	A newly developed intrusion detection system that relies on machine learning and deep learning techniques to identify new attacks in an IoT environment	ToN-IoT-Windows	98.39% medium neural network 98.22% weighted KNN 97.97% fine Gaussian SVM
[19]	The suggested model extracts the features of recurrent model hidden layers and then uses a kernel-based principal component analysis (KPCA) feature selection strategy to find the best features. Finally, the best features of the recurrent models are combined, and classification is performed with an ensemble meta-classifier	SDN-IoT	99% in network attacks detection 97% network attacks classification

3. Datasets and Methods Used in SATIDS

3.1. Database Description

The performance of an AI-powered IDS is dependent on the quality of the dataset used for its training. The lack of an up-to-date, real-world generated dataset is one of the key obstacles in training effective AI-based IDS. The main reason that public data sets are not available for the intrusion detection area is for privacy and legal considerations. To verify the proposed system's performance in two different environments, it was trained and evaluated using two of the most recent realistic datasets, namely ToN-IoT and InSDN datasets which are introduced in the following subsections. These datasets were selected to test the system robustness and generalization and to say with a certain confidence that the proposed IDS behaves well in both SDN and IoT environments. The ToN-IoT dataset simulates an IoT environment with 43 features and nine types of attacks, while the InSDN dataset enables some cyberattacks using OpenFlow protocols in the SDN architecture, with 79 features and eight types of attacks.

3.1.1. ToN-IoT Dataset

The ToN-IoT dataset is obtained from a practical and large-scale network developed by the UNSW Canberra Cyber IoT Lab [11]. The dataset contains a range of routine and cyberattack events from IoT networks. A newer testbed was built at the same IoT lab to connect a range of virtual computers, physical tools, hacking systems, cloud and fog systems, and IoT sensors to simulate the functionality and scalability of the automotive

IoT and Enterprise 4.0 networks. This generated dataset contains different recent smart city-based attacks such as DoS, DDoS, and ransomware. These attacks have been deployed against web applications, IoT gateways, and computer systems across the IoT network.

This dataset contains 43 features with 461,043 total observations divided into 300,000 normal and 161,043 attack observations. The dataset was divided into 70% and 30% for train and test sets, respectively. Table 2 shows the statistic of normal traffic and different attack vectors as presented in the dataset [10].

Table 2. ToN-Iot database statistics.

Attacks	No. of Instances	%
Normal	300,000	65.07
Backdoor	20,000	4.34
Scanning	20,000	4.34
Injection	20,000	4.34
Password	20,000	4.34
XSS	20,000	4.34
Ransomware	20,000	4.34
DDOS	20,000	4.34
DOS	20,000	4.34
MITM	1043	0.23

3.1.2. InSDN Dataset

The InSDN dataset covers many scenarios and attack classes, including Probe, DoS application, web attacks, brute force attack, password guessing, U2R, and DDos attacks [12]. In addition, InSDN regular traffic also encompasses several common features. The dataset source of attacks originates from both an internal and an external network to imitate the actual attack scenarios. It covers about 80 statistical aspects in CSV format such as protocol, duration, byte number, and packet number. The overall number of dataset instances for normal and attack traffic is 343,939, a total of 68,424 for normal traffic, and 275,515 for attack traffic as shown in Table 3.

Table 3. InSDN database.

Attacks	No. of Instances	%
Normal	68,424	52.32
DoS-Application	31,628	24.19
Probe	15,225	11.64
DDOS	9943	7.6
DoS-Network	3772	2.88
Brute force attack	1405	1.07
Web Application	192	0.147
Botnet	164	0.125
U2R	17	0.013

The attack traffic in InSDN emulates the same normal behavior since normal and malicious traffics are transmitted to the SDN controller for decision-making. Furthermore, the centralized perspective of the SDN network and the separation of the data plane from the control plane gives the attacker a new option compared to the traditional network of numerous attacks. These intrusions are not easy to be detected since the intruder is

permitted to access to the victims’ server. As a result, such a data collection might be a good indicator for the model assessment that reflects the real-world scenario. Furthermore, the InSDN dataset contains no redundant records to prevent the learner model from distracting itself from the most common records [12,13,15].

3.2. LSTM Network

DNN is a type of neural network which has a hierarchical structure with many hidden layers. The network becomes denser when hidden layers increase. DNN has become trendy among cybersecurity researchers in the last few years. LSTM is a RNN network that uses the input information to predict the next sequence according to the detected pattern. The recurrence of the network delay, which represents the complex system output, is the RNN’s most essential characteristic. Furthermore, RNNs maintain an activation vector that makes the RNN an intense neural network. However, due to explosion and gradient difficulties, it is also difficult to teach RNNs to rely on data from time series for a long time [20].

The LSTM architecture makes a good choice for solving the RNN extinction gradient problem. LSTM uses a memory cell, which in sequential data may represent long-term dependencies. Figure 1 shows the internal description of the LSTM memory cell. The four internal memory gates are responsible for regulating the interlinkage between memory units. The input gate decides whether the input signal will alter the memory cell’s status or not. On the other hand, the output gate determines which memory status can be changed depending on the previous input. The forgotten gate can decide to forget its status (or remember it), under the following equations, at time t :

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i) \tag{1}$$

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f) \tag{2}$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_{t-1} + b_o) \tag{3}$$

$$c_t = f_t c_{t-1} + i_t \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \tag{4}$$

$$h_t = o_t * \tanh(c_t) \tag{5}$$

where $i_t, f_t, o_t,$ and c_t are the input, forget, output and cell gate activations, respectively, at any time t .

σ : Logistic sigmoid function.

W_x : Weight matrices.

b_x : Variable biases.

h_{t-1} : Hidden state at time step $t-1$.

c_{t-1} : Cell state step at time $t-1$.

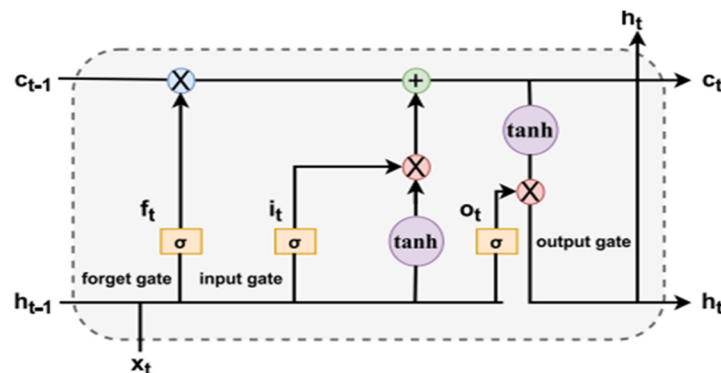


Figure 1. LSTM internal architecture structure surroundings “Reprinted/adapted with permission from Ref. [20]”.

3.3. MRMR-Based Feature Selection

To deal with the high dimensionality of huge datasets, feature selection (FS) approaches are used. FS aids an IDS in both increasing its output and decreasing the amount of time required to train its model. To create better IDS models, a subset of the original features is chosen. It is possible to categorize the methods by the level of supervision that is available: supervised, unsupervised, and semi-supervised. Filtering, wrapping, and embedding are all examples of different FS approaches [21,22].

MRMR-FS is one of the most popular FS methods available. Bioinformatics and multimedia processing are only two examples of the many fields where it is widely employed. Using mutual information as an indicator of reliance, it is a forward-selection filter approach. In MRMR, features are picked based on their relevance to the target variable while also minimizing the correlation between them [22,23].

MRMR-FS detects the most effective features as follows:

- I The highest relevance feature $\max_{x \in \Omega} V_x$ is selected and added to an empty feature set, S , which is calculated using Equations (6) and (7).

$$M_I = \sum_{x \in X} \sum_{y \in Y} p(x, y) \log \frac{p(x, y)}{p(x)p(y)} \tag{6}$$

$$V_x(S) = \frac{1}{|S|} \sum_{x \in S} M(C, x_i) \tag{7}$$

where x and y are two random variables.

C : the output

M_I : Mutual information of feature x and y .

$M(C, x_i)$: Mutual information of feature x_i with class C .

- II Nonzero relevance and redundancy features are chosen in both S and S_c . In case S_c does not contain a nonzero relevance and zero redundancy feature, go to Step IV.
- III Select the largest relevance feature, $\max_{x \in S^c, w_x=0} V_x$ and add the selected feature to set S .
- IV For all features in set S_c , repeat Step II until nonzero redundancy occurs.
- V Choose nonzero relevance and nonzero redundancy feature with the largest mutual information quotient (MIQ) value within S_c and add that feature to S set using Equation (8).

$$\max_{x \in S^c} MIQ = \max_{x \in S^c} \frac{I(x, y)}{\frac{1}{|S|} \sum_{z \in S} I(x, z)} \tag{8}$$

where x and y are two random variables and $I(x, y)$ is the mutual information between x and y .

- VI Step IV should be repeated until there is no longer any relevancy for any feature S_c .
- VII Zero relevance features are added randomly to S .

3.4. Modified Artificial Fish Swarm Algorithm (AFSA)

AFSA is one of the best swarm intelligence algorithms for optimization. Fish schooling and swarm intelligence inspired the development of population-based evolutionary computing techniques. Random AFSA solves optimization issues based on the movement and intelligence of swarms in the process of finding food. In order to solve the optimization challenge, random and parallel searching algorithms are used to mimic fish behavior such as following and swarming in order to find the best possible solution for the entire population as in Figure 2. In AFSA, fish have three behaviors in eating: preying, swarming, and following [24].

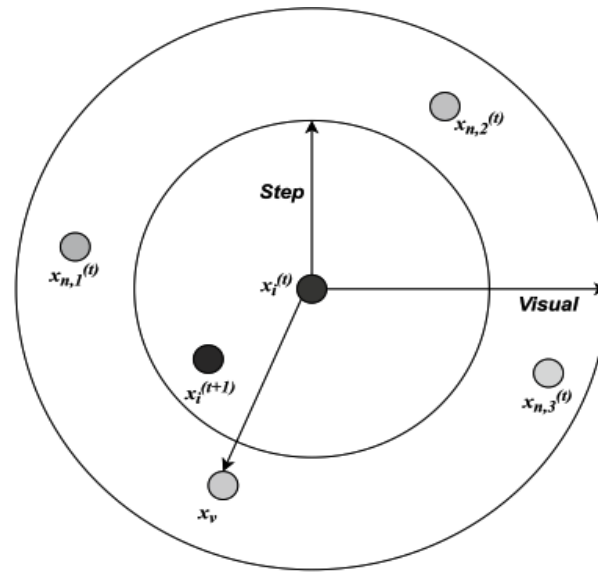


Figure 2. The relationship between an artificial fish and its surroundings “Reprinted/adapted with permission from Ref. [24]”.

The three possible behaviors of an AF are described in the following:

- Preying behavior: A fish’s basic activity is to prey on other fish in order to relocate to an area where there is more food available. This behavior is modeled in AFSA using an AF’s current position and the area immediately surrounding it.

$$\vec{x}_v = \vec{x}_i^{(t)} + (v \cdot u(0, 1) \cdot \frac{\vec{u}}{\|\vec{u}\|}) \tag{9}$$

where $\vec{x}_i^{(t)}$: the current i-th position of AF

\vec{x}_v : the random i-th position of AF

v : visual length

$u(0, 1)$: random number

\vec{u} : vector of uniformly distributed in $(-1, 1)$

$\frac{\vec{u}}{\|\vec{u}\|}$: Unit vector

- Swarming behavior: To avoid danger and crowds, fish gather together in groups in nature. It is a common feature in AFSA that fish swarms behave this way. According to the next equation

$$\vec{x}_{s,i} = \vec{x}_i^{(t)} + (s \cdot u(0, 1) \cdot \frac{\vec{x}_c - \vec{x}_i^{(t)}}{\|\vec{x}_c - \vec{x}_i^{(t)}\|}). \tag{10}$$

- Following behavior: When an AF locates a spot where there is a greater concentration of food, other fish will follow it, as shown by the following equation.

$$\vec{x}_{f,i} = \vec{x}_i^{(t)} + (s \cdot u(0, 1) \cdot \frac{\vec{x}_j - \vec{x}_i^{(t)}}{\|\vec{x}_j - \vec{x}_i^{(t)}\|}). \tag{11}$$

- As the number of artificial fish (AFs) increases, AFSA's convergence speed and accuracy are improved. However, as the number of AFs rise, so does the algorithm's complexity. As the number of iterations increases, so does the effect of an AF's fitness function value on the ultimate optimal outcome in AFSA, regardless of whether that value is too large or too small. As a result, the algorithm complexity is increased by the use of these weak fish. For maximum or minimum optimization, if a weak AF's fitness function value exceeds or falls below a defined threshold, it will be swallowed in the AFSA computations [25,26].

3.5. Energy Harvesting Real-Time Routing Protocol (EHRT)

EHRT depends on the principle of clustering using LEACH routing protocol which is optimized by a modified AFSA that modifies the fitness function of AFSA [9,25–27]. EHRT protocol consists of the following two phases:

- Phase I: Cluster Formation and CH Election

In this step, clusters are established and cluster heads (CHs) are selected depending on the clustering algorithm's guiding principle. Equal segmentation of the area space is utilized to form clusters. Each cluster's CH is selected based on the node that has the highest energy and lowest distance from the base station (BS).

As all sensor nodes initially have the same energy, in our architecture, the node in the cluster's center is designated as its center of neighboring nodes. The processes of cluster creation and CH election are shown in Figure 3. After this election, the BS runs the modified AFSA (phase II) to select the optimal CHs set at the beginning of each round.

- Phase II: Optimal CH Selection by Using modified AFSA

As CHs spend significant amount of energy during the data collection and transfer phase, EHRT rotates the CH role among sensor nodes for each cluster using modified AFSA at the start of each transmission round. As a result, energy waste is being avoided and the energy usage of each node is being optimized to extend the network's lifetime. Figure 4 depicts the CH selection procedures.

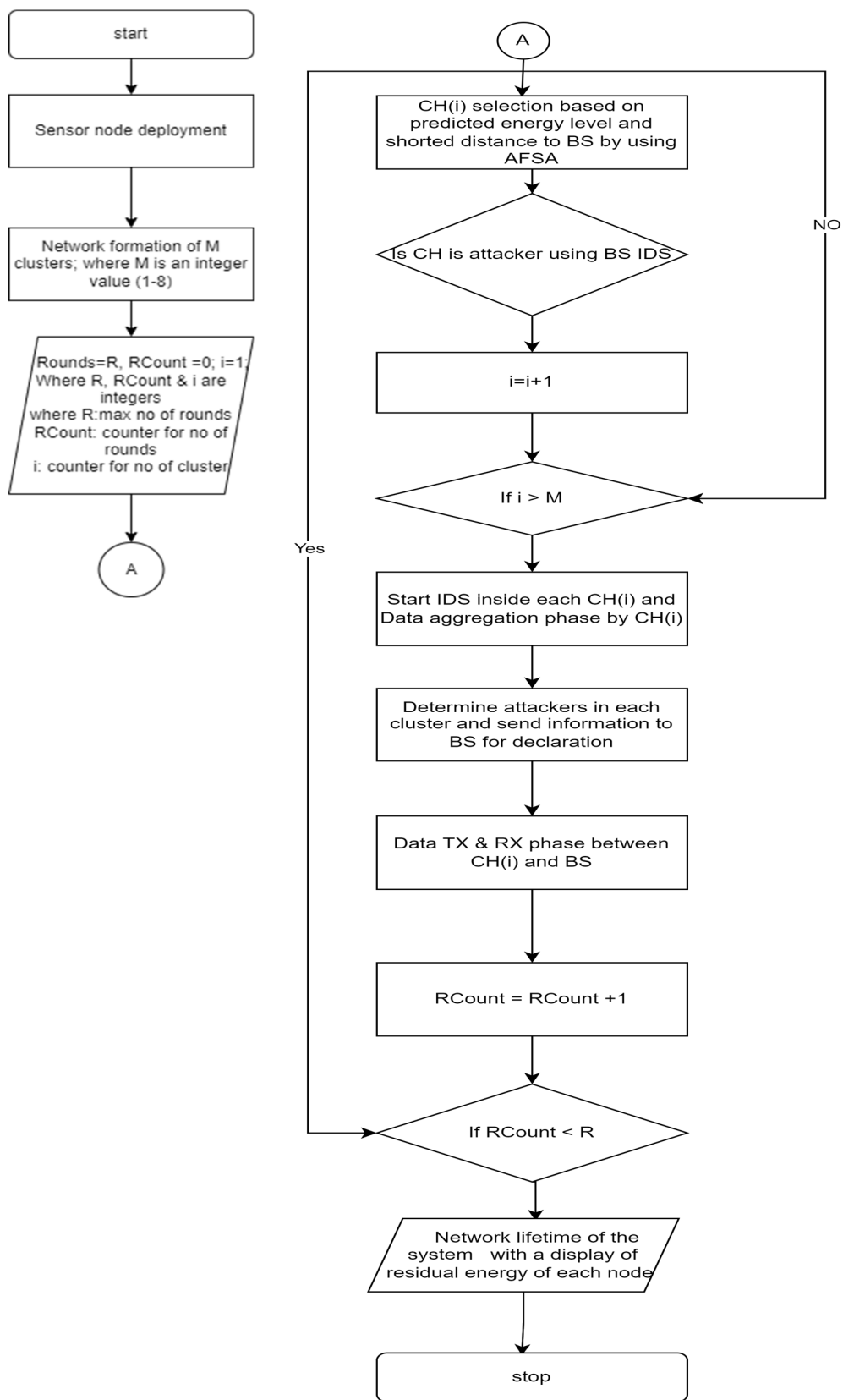


Figure 3. Flowchart of EHRT.

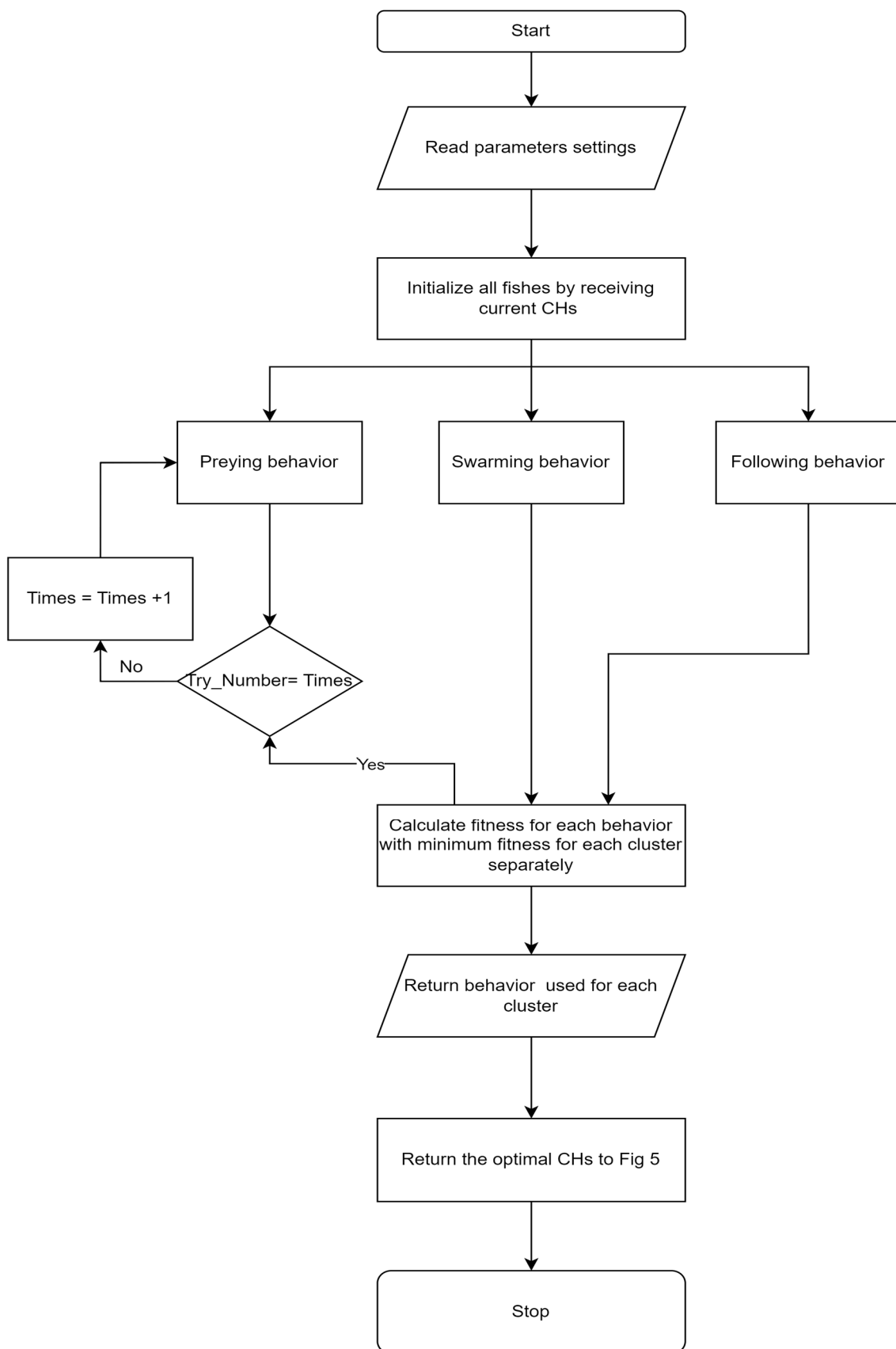


Figure 4. Flowchart of CHs selection using AFSA “Reprinted with permission from Ref. [26]”.

4. The Proposed MRMR-SATIDS System

The main objective of any IDS system is to improve the accuracy of attack detection and discovering the attack type. This section provides a detailed description of the proposed SATIDS system which depends on both the MRMR FS technique and a LSTM network for the classification purpose. Then, SATIDS is applied to a WSN after its training and testing to study its performance when dealing with large number of mobile sensor nodes and its effect on the energy consumption.

Figure 5 presents the inner structure of the proposed SATIDS system. Rectified linear unit (Relu layer) activation function performs a threshold operation to an input element concerning zero as some input data contain negative values. Relu layer passes the positive input and gives zero value to the negative values. Then, a number of LSTM layers are used; each layer is followed by a 0.2 probability dropout layer. The dropout layer probability value is chosen to be 0.2 using a trial and error method to give the highest overall performance metrics as presented in Figure 6.

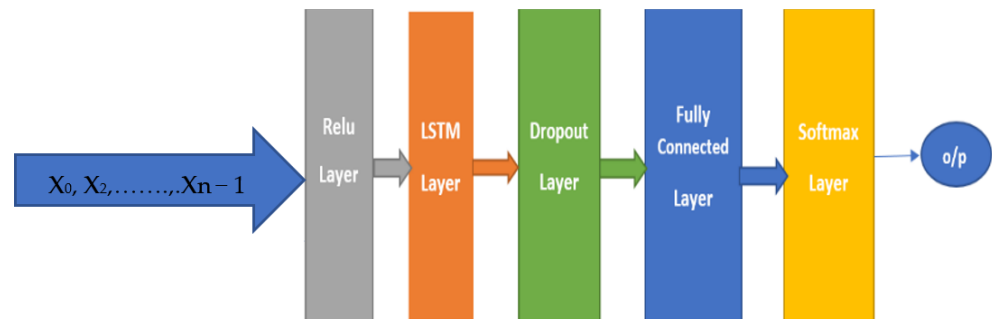


Figure 5. Modified DL-based LSTM network.

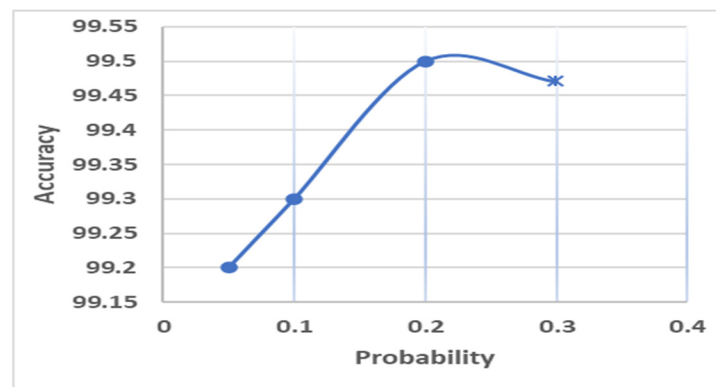


Figure 6. Proposed algorithm parameters (accuracy vs. dropout layer probability).

The dropout layer sets the input to avoid overfitting, which decreases the network performance parameters. The initial learning rate was set to 0.0001, and LSTM batch size of 2000 to provide the highest performance parameters. A fully connected layer flattens the output of the previous layers and converts it to a single vector. Finally, the Softmax layer eventually enters the classification layer classifying the data as benign or any form of attacks.

4.1. Training and Testing of SATIDS

Figure 7 and Algorithm 1 summarize different phases of the proposed MRMR-SATIDS. To accurately detect attack type, the following steps are followed:

- 1 Preparation of the dataset: For continuous features, remove NaN and infinity values from the data using the mean of the appropriate column. In addition, the IP address and time stake label are removed.

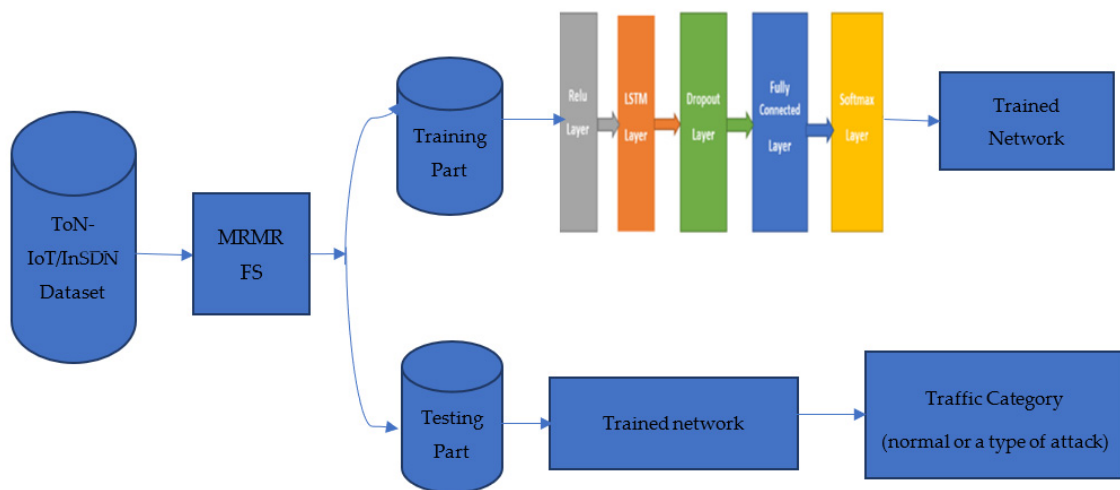


Figure 7. Block diagram of the proposed system.

The two datasets contain raw data. If this raw data is input into our proposed system, slow convergence will occur. Normalization strongly influences the performance of a neural network. With unnormalized data, numerical ranges of features may vary strongly. The data (x) can be normalized by subtracting the mean (μ) of each feature and dividing by the standard deviation (σ). This way, each feature has a mean of 0 and a standard deviation of 1. This results in faster convergence.

$$x := \frac{x - \mu}{\sigma}$$

- 2 Feature selection process: Extract the important features of the dataset using the MRMR-FS and then modify the dataset.
- 3 The dataset is divided into 70% for training and 30% for testing.
- 4 Training process: SATIDS network based on LSTM network is trained using the modified training part of the dataset. The data is separated from the type of data. Each line of data enters the network separately.
- 5 After training, the network is ready to identify the category of data (normal or malicious).
- 6 When the tested part enters the trained model, it is categorized as normal traffic or a type of attack.
- 7 The network performance is analyzed by comparing the system output to the testing result.

Algorithm 1: Training and Testing the Proposed MRMR-SATIDS

OUTPUT traffic type (normal/anomaly) and determine the attack type

INPUT ToN-IoT/InSDN datasets

Normalize dataset

Remove IP address and time stack from the dataset

Select the more effective features and eliminate the less effective ones using MRMR-FS

Divide the dataset into 70% for training and 30% for testing

Train the proposed IDS using the training dataset which contains traffic data and label

For each data line

train network with each traffic record and consider the traffic label

END FOR

Insert the testing dataset to the trained IDS

Classify traffic (normal or a type of attack)

Calculate IDS performance

4.2. Applying SATIDS to an IoT system

After testing, the SATIDS is applied to a large-scale IoT with mobile nodes. To simulate this environment, 250 sensor nodes are distributed randomly at a 300 m × 300 m region divided into eight clusters (seventh level hierarchy) as shown in Figure 8. BS is located at (0,0). The initial energy of each node is set to 0.2 J and its threshold energy is 50 mJ. Each node uses 50 nJ/bit for the transmission and reception process. Energy dissipation is 100 pJ/bit and the data packet size is set to 4000 bits. Attacked nodes are randomly distributed between the clusters. Algorithm 2 presents SATIDS’s procedures to secure an IoT system.

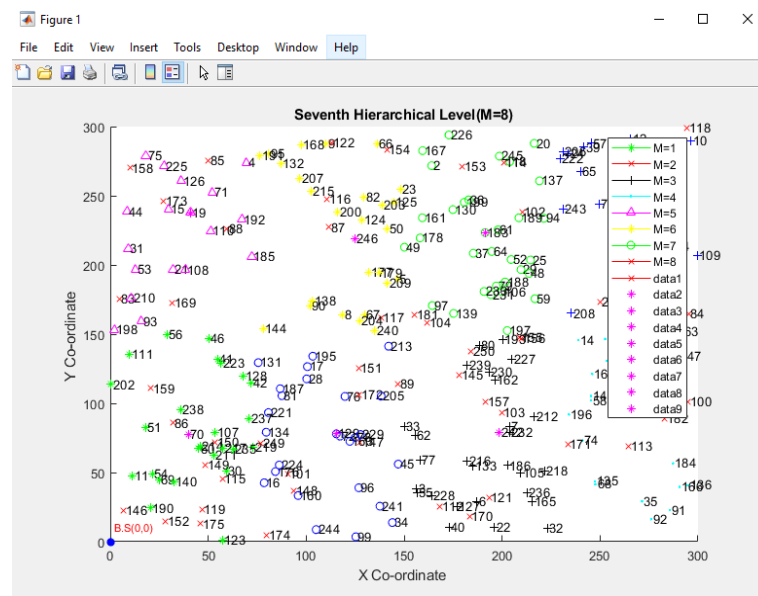


Figure 8. Eight cluster IoT network.

AFSA with a modified fitness function is used to optimize a clustering approach based on EHRT. As compared to selecting a single cluster behavior, the fitness function has been adjusted so that optimal CHs for each cluster are selected. Cluster formation and the optimal CHs election are the two phases of the EHRT technique (described in Section 3.5). Algorithm 2 summarizes the application of the proposed MRMR-SATIDS to an IoT network.

Algorithm 2: Applying MRMR-SATIDS to a WSN network

OUTPUT Detecting the source of the anomaly traffic and determine the attack type

INPUT Randomly distributed IoT nodes

CHs are selected according to EHRT algorithm

Each CH aggregates traffic from its neighboring nodes

CHs uses the proposed SATIDS to detect any anomaly traffic from neighboring nodes

Determine the ID of this node and send its ID to the BS

BS saves the ID of attacked nodes in a blacklist and shares it with other nodes

FOR the received traffic from CHs

BS uses the proposed SATIDS to scan received traffic from all CHs

IF any attacked CH is detected

BS declares its ID and the election process is repeated in that cluster to elect another CH

END IF

END FOR

5. Experimental Evaluation

5.1. Performance Metrics

Using MATLAB on an Intel Core-i7 processor with 8 GB RAM on the Windows 10 platform, the proposed SATIDS system was simulated, trained, and tested. The performance of the proposed system was analyzed using the confusion matrix, which is a table structure commonly used for multi-class assessment. It consists of four main values determined for each class [17]:

- True Positive (TP): The number of normal observations in dataset traffic classified correctly as normal by the technique.
- True Negative (TN): The number of intrusion observations in the IoT network properly classified by the algorithm as intrusions.
- False Positive (FP): The number of normal IoT traffic observations wrongly identified by the algorithm as abnormal.
- False Negative (FN): The number of abnormal observations wrongly identified by the algorithm in IoT network traffic as normal observations.

To prove the efficiency of the proposed system, the number of LSTM layers and hidden layers in each LSTM layer were varied and the following performance metrics of detecting different types of attack were recorded using the whole features and compared with using MRMR-FS technique.

- Accuracy: The ratio of correctly calculated observations by the network to the total number of samples present in the determined dataset.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \quad (12)$$

- Precision: The number of correctly detected observations, divided by the total number of observations that the model identifies as an attack.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \quad (13)$$

- Detection rate (DR): The ratio of the correct observations observed by the model to the total number of tests.

$$\text{DR} = \frac{\text{TP}}{\text{TP} + \text{FN}} \quad (14)$$

- F1-score: Evaluating the test accuracy by calculating the average weight of the detection rate.

$$\text{F1 - score} = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \quad (15)$$

5.2. Results and Discussion

5.2.1. ToN-IoT Dataset

Figure 9 presents the performance of the proposed SATIDS system with and without using MRMR-FS when detecting normal/anomaly traffic. As seen in Figure 9, SATIDS has its highest performance when using MRMR-FS, two LSTM layers and 500 hidden layers: 97.5% accuracy and 98.05% F1-score. In the case of detecting a backdoor attack, as shown in Figure 10, the system has almost the highest performance when using MRMR-FS, two LSTM layers and 500 hidden layers: 97% precision and 97.28% F1-score. The detection rate values are almost equal in all cases.

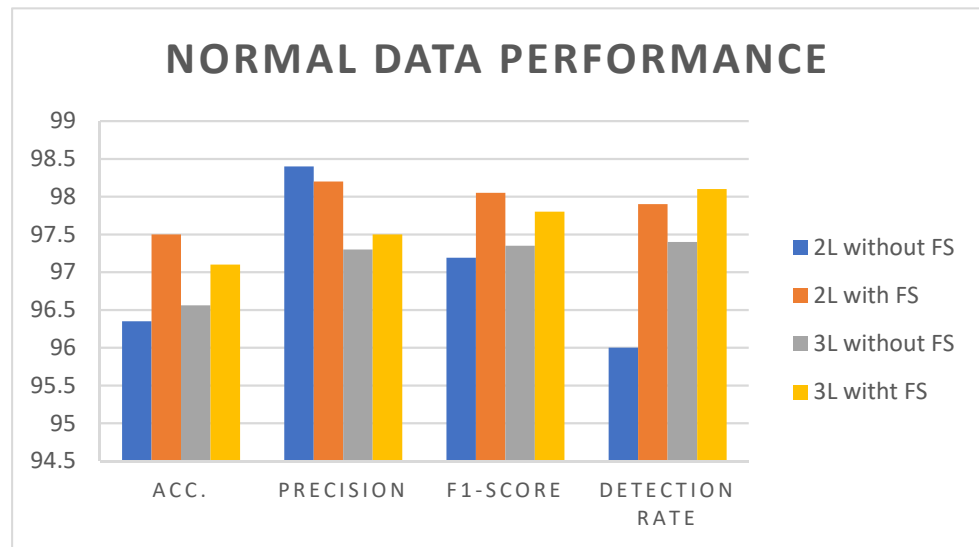


Figure 9. Performance comparison of the proposed system using ToN-IoT dataset.

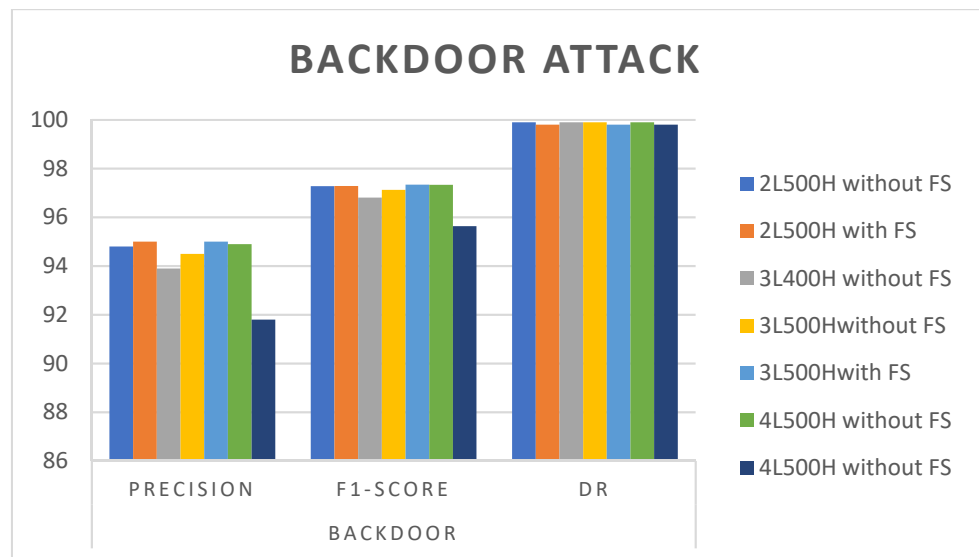


Figure 10. Detecting a backdoor attack.

Figures 11 and 12 represent the performance of the proposed SATIDS system in detecting DOS and ransomware attacks, respectively. As shown, the network has its highest performance when using three LSTM layers and 500 hidden layers when using the MRMR-FS technique. The performance of the proposed SATIDS system in detecting DDOS, injection, and scanning attacks, is shown in Figures 13–15, respectively. As seen in Figures 13–15, the network also has its highest performance when using three LSTM layers and 500 hidden layers without using the FS technique.

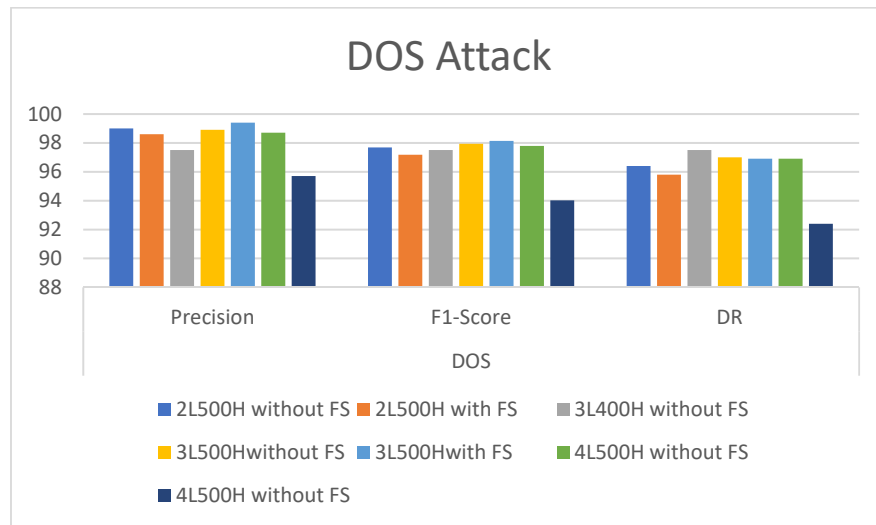


Figure 11. Detecting a DOS attack.

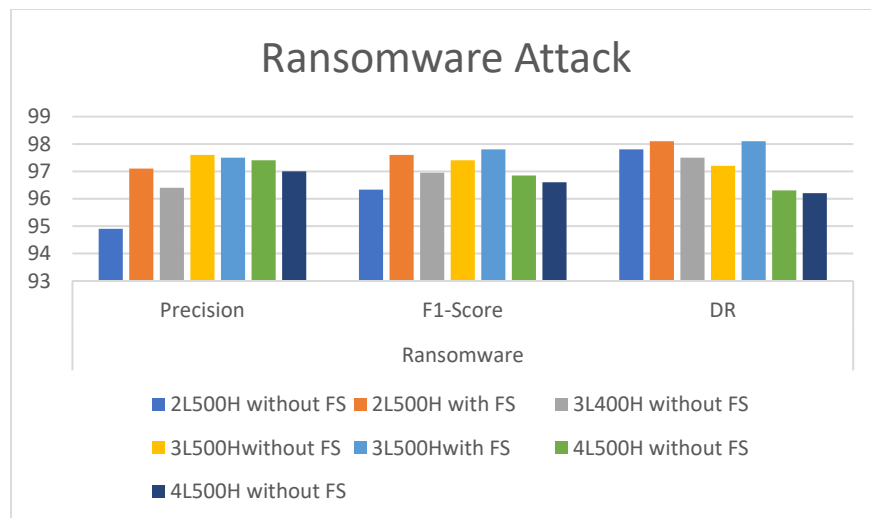


Figure 12. Detecting a ransomware attack.

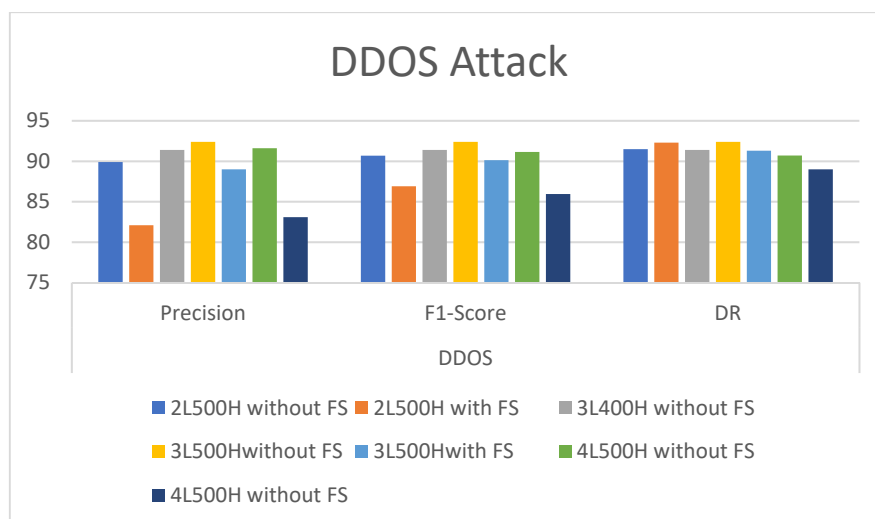


Figure 13. Detecting a DDOS attack.

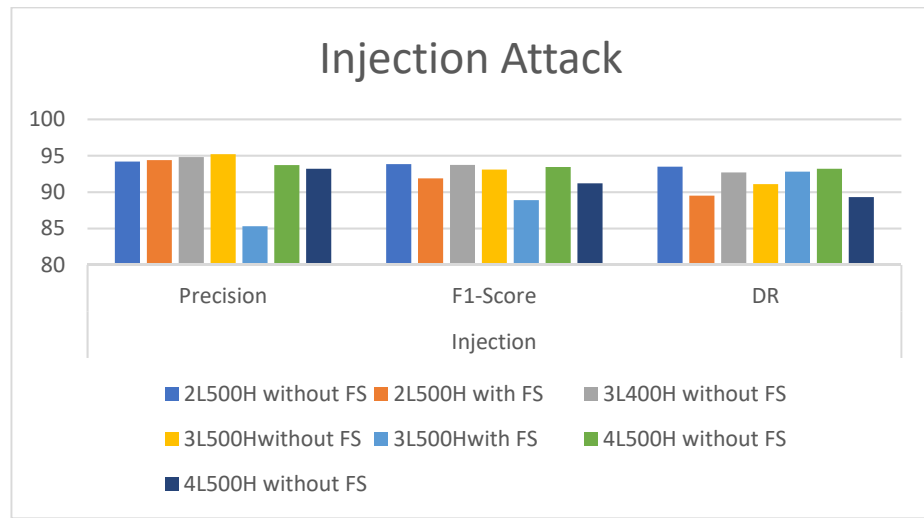


Figure 14. Detecting an injection attack.

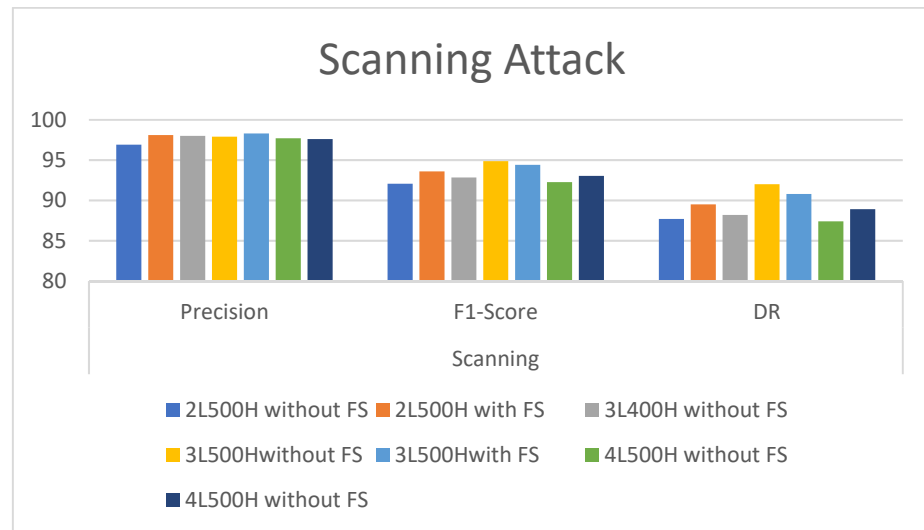


Figure 15. Detecting a scanning attack.

The proposed SATIDS network achieves its highest performance when detecting password attack when using two LSTM layers and 500 hidden layers without using the FS technique as shown in Figure 16. SATIDS has its highest performance in detecting XSS attack when using three LSTM layers 400 hidden layers as shown in Figure 17.

5.2.2. InSDN Dataset

Figures 18–23 show the performance metrics of SATIDS system when using the InSDN dataset for classifying the type of received traffic. The system has its highest performance when using two LSTM layers and 500 hidden layers with MRMR-FS for detecting normal/anomaly traffic, DDOS attack, and probe attack as in Figures 18–20, respectively.

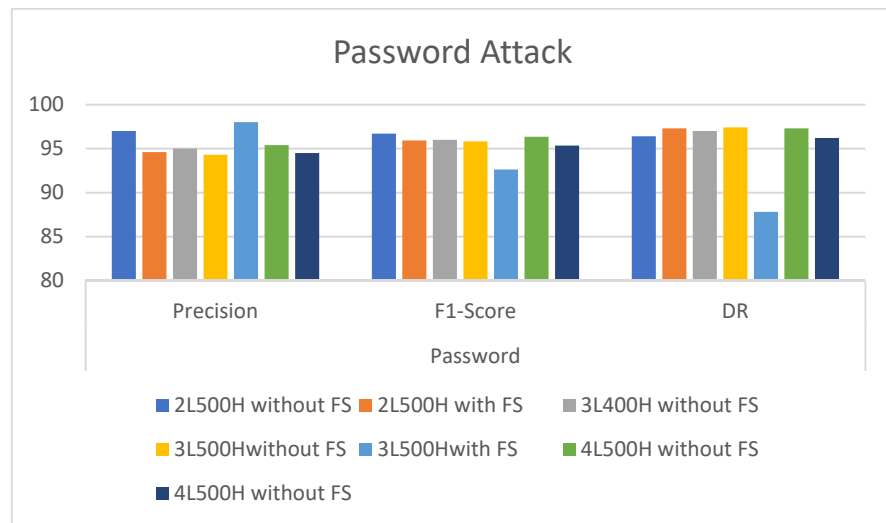


Figure 16. Detecting a password attack.

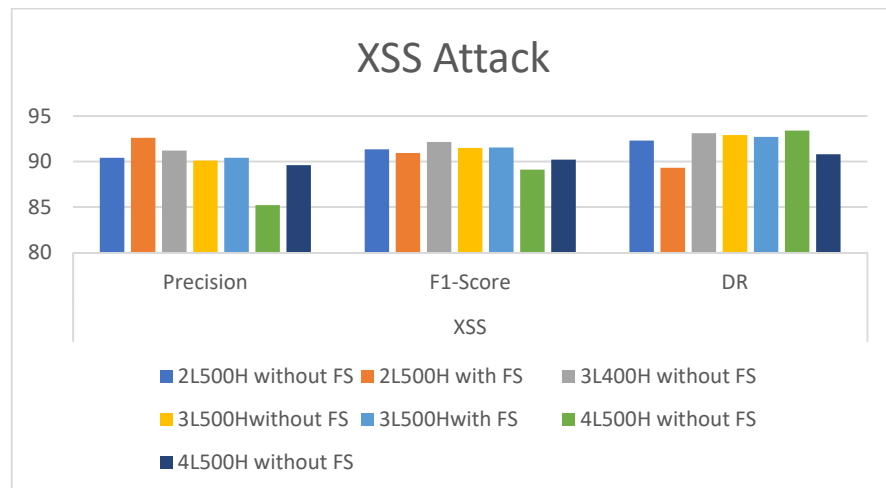


Figure 17. Detecting an XSS attack.

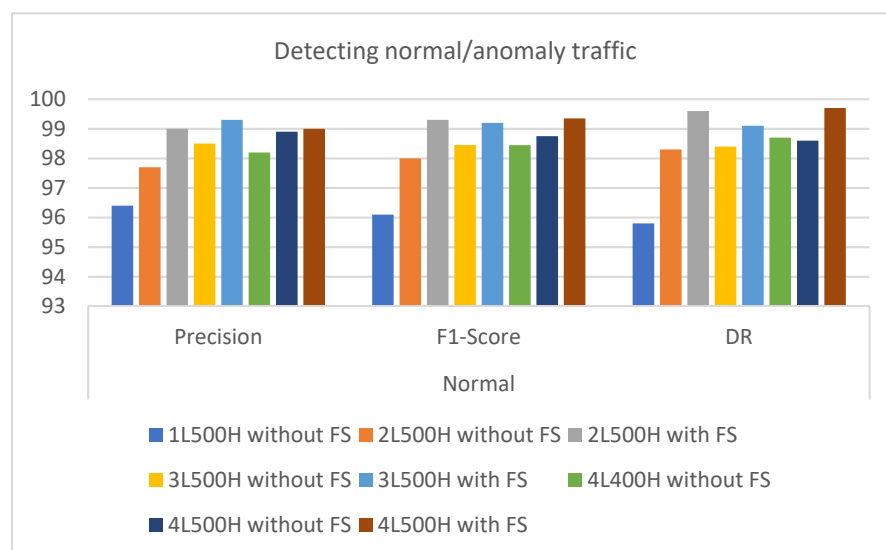


Figure 18. Performance comparison of the proposed system using the InSDN dataset.

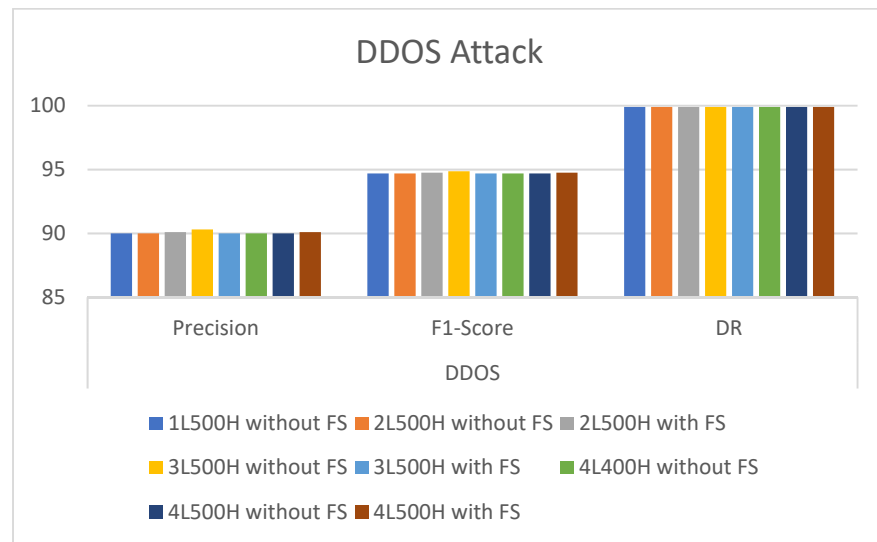


Figure 19. Detecting a DDOS attack.

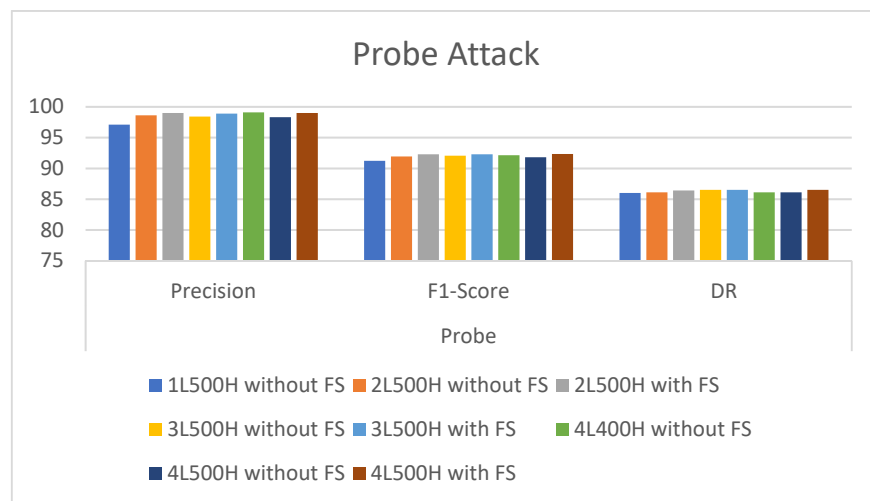


Figure 20. Detecting a probe attack.

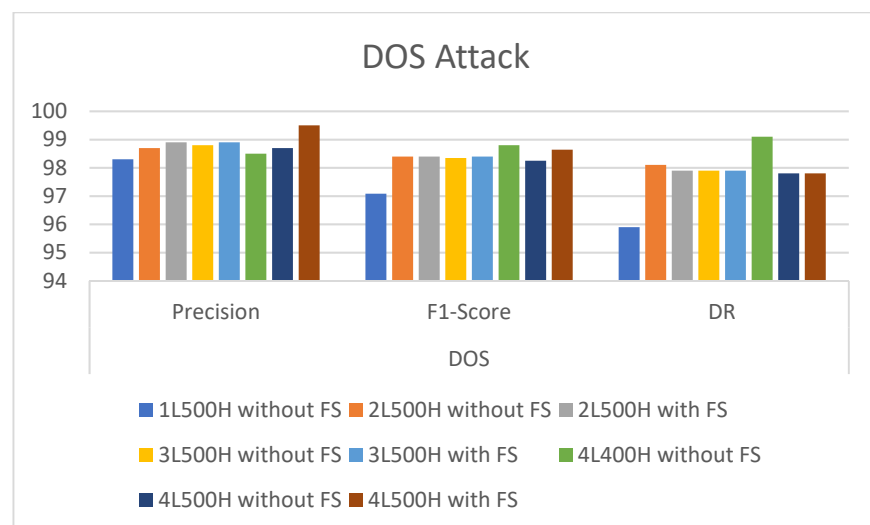


Figure 21. Detecting a DOS attack.

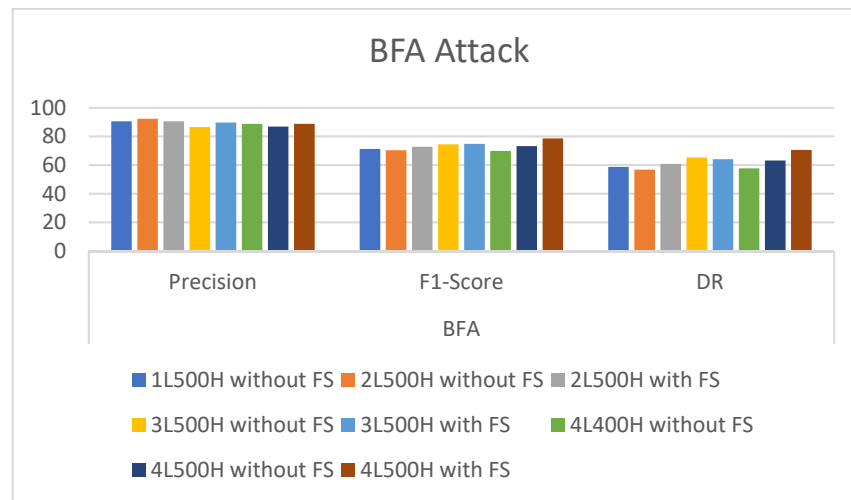


Figure 22. Detecting a BFA attack.

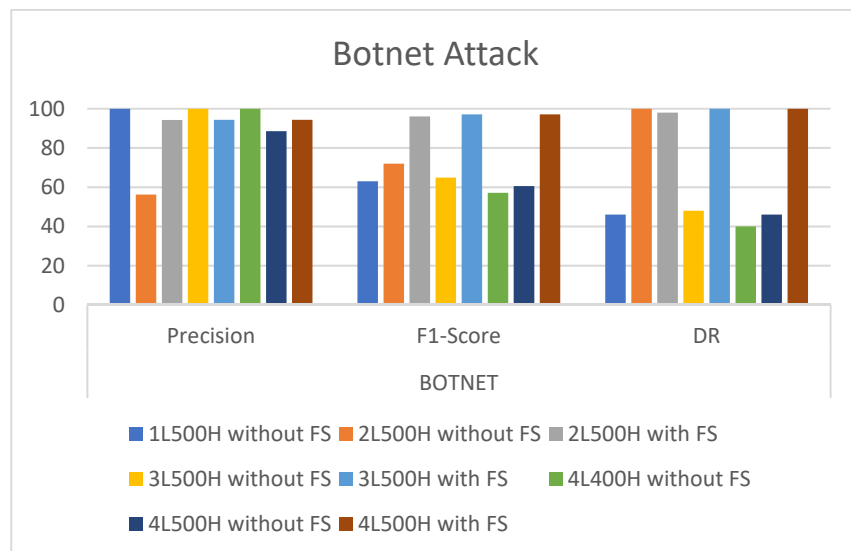


Figure 23. Detecting a botnet attack.

For detecting DOS attacks, the system has its highest performance when using four LSTM layers and 500 hidden without using MRMR-FS technique as shown in Figure 21. However, as shown in Figures 22 and 23, the highest performance of the system occurs when using three LSTM layers and 500 hidden with using MRMR-FS technique for detecting BFA and BOTNET attacks, respectively.

5.3. Performance Comparison

Table 4 shows a performance comparison between the proposed MRMR-SATIDS system and other approaches using the ToN-IoT dataset. GRAD [28] outperforms MRMR-SARIDS, RF and NB in detecting normal, backdoor, DOS, and XSS attacks with a range of 0.2% to 2.8% difference in DR. However, SATIDS outperforms the technique in [28], RF and NB with the same percentage difference in DR in case of DOS, Injection, and Ransomware. MRMR-SARIDS also outperforms RF and NB in detecting DOS and XSS attacks, with a DR difference ranging from 5% to 70%.

Table 4. Performance comparison of DR % using ToN-IoT database.

Technique \ Parameter	Normal	Backdoor	DDOS	DOS	Injection	Ransomware	XSS
MRMR-SATIDS	98.1	99.8	91.3	96.9	92.8	98.1	92.7
Random Forest (RF) [28]	100	99.98	90.40	91.97	93.53	99.4	85.47
Naïve Bayes (NB) [28]	100	99.22	26.8	91.7	92.96	79.98	19.02
GRAD-Transform [28]	100	100	91	99	90	98	93

5.4. Implementing the Proposed SATIDS System in an IoT System

The MRMR-SATIDS algorithm is implemented in an IoT that contains 250 (225 normal nodes + 25 attacked nodes) randomly distributed nodes. The region is divided into eight clusters (seventh level hierarchy). To conserve power, the routing protocol is used. By applying both the MRMR-SATIDS trained algorithm and the EHRT routing protocol, the IoT network saves the device’s power and data during the transmission process. Figure 24 depicts the alert messages that the CHs generate when an attacker node is detected. These messages contain the node ID and the cluster number. BS saves the node IDs in a blacklist and declares it to other nodes.

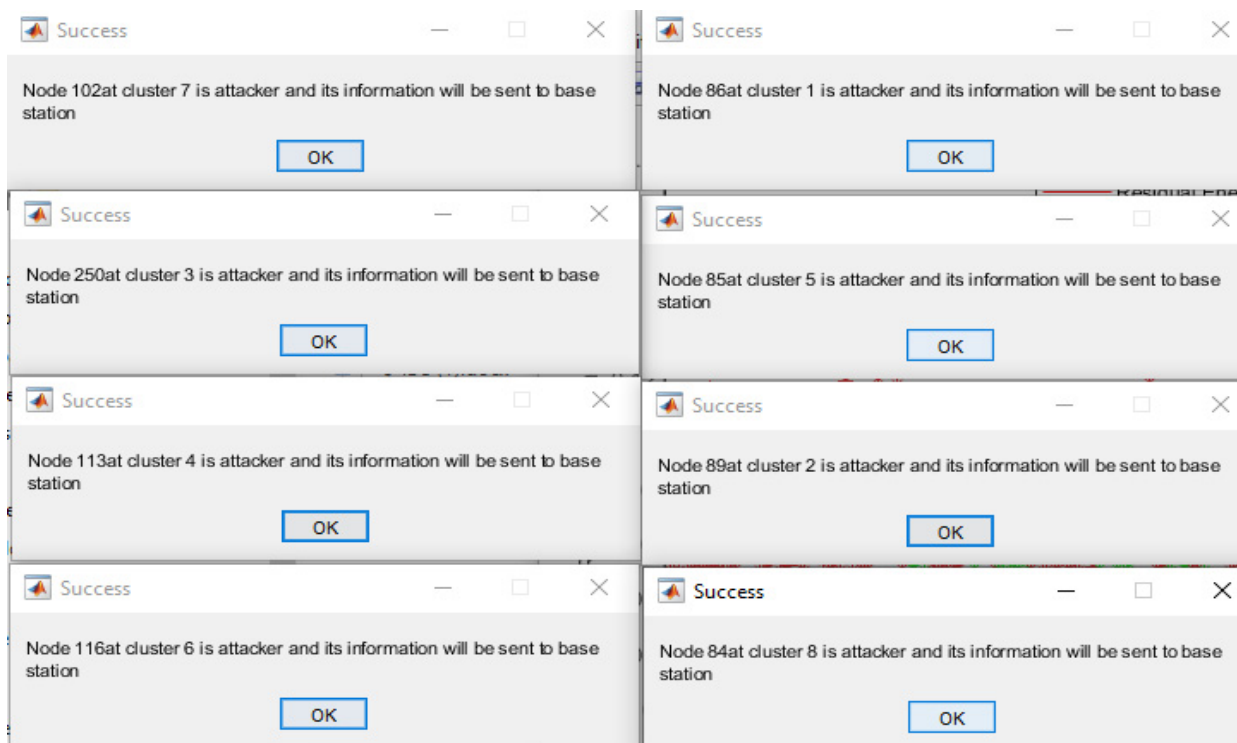


Figure 24. Alerts generated by CH nodes.

Figure 25 represents the network lifetime after 400 rounds. The red curve represents the network lifetime without applying the MRMR-SATIDS security technique, while the green curve represents the network lifetime when applying the MRMR-SATIDS security technique. As shown in Figure 25, the difference between the two curves as the security algorithm prevents attack nodes from being CHs. Despite of the decrease of network lifetime by 5.33% when using our proposed security system, the system security increased.

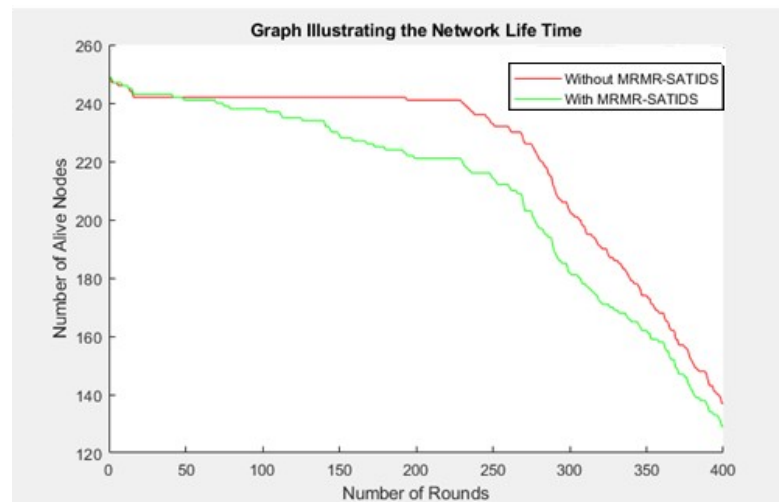


Figure 25. Network lifetime.

The residual energy after 400 rounds is represented in Figure 26. The red points present the normal nodes while the green points present the attacker nodes. As noticed from the figure, after 400 rounds, only 30% of the normal nodes reach the threshold energy. Those nodes are the ones that were elected many times to be the CH. Only 8% of the attacked nodes reached the threshold since the BS did not select any of the attacked nodes as a CH after putting them in a blacklist. The proposed system was able to detect 24 out of 25 attacked nodes.

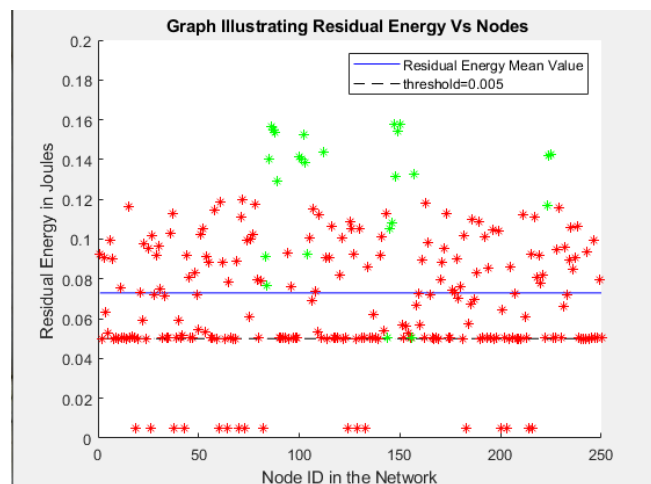


Figure 26. Nodes residual energy.

5.5. Robustness and Generalization Test

In machine learning and deep learning, robustness and generalization are critical. To demonstrate the robustness and generalizability of the proposed method, its performance is evaluated on two different network intrusion detection datasets such as ToN-IoT and InSDN datasets for network attack detection and network attack classification. Figures 9–23 exhibit detailed performance results for network attack detection and classification. On both datasets, the proposed model detected the network attacks with 99% accuracy. As a result, the proposed model should be regarded as robust and generalizable. The proposed approach, however, may not be resilient or generalizable to an adversarial updated dataset or adversarial environment, therefore the case study is not shown in this paper.

6. Conclusions and Future Work

IoT is an extremely powerful platform for connecting objects to the Internet. However, IoT applications are susceptible to many security vulnerabilities that need to be mitigated. Users' safety and privacy concerns are on the rise due to IoT system's massive attack vector. Improved security measures can be achieved with the use of ML. It is a never-ending challenge to find the best ML-based IDS techniques.

This research offers a two-level deep learning methodology called SATIDS based on the LSTM network to improve the performance of IDS systems. MRMR feature selection technique is utilized to minimize the huge number of features in the dataset. Using the MRMR technique improved the proposed IDS performance and its ability to detect various types of attacks.

Two of the latest realistic IoT datasets, namely ToN-IoT and InSDN, are used to evaluate the efficiency of the proposed system. These datasets contain various types of attacks that could cause IoT failure. Using MRMR offers a much better overall system performance. Using 500 hidden and two LSTM layers achieves accuracy of 97.5%, precision of 98.4%, detection rate of 97.9%, and F1-score of 98.05% on ToN-IoT dataset, and precision of 99%, DR of 99.6%, and F1-score of 99.3% on the InSDN dataset. In every experiment across multiple network datasets, the suggested model outperformed the state-of-the-art approaches. The proposed MRMR-SATIDS system succeeded in detecting a wide range of attacks. This demonstrates the suggested model's strength and flexibility for use with other network traffic datasets.

Finally, the trained algorithm is implemented in a power-saving IoT system that contains 250 mobile nodes to get close to the real network. This network used the EHRT routing protocol to save sensor energy and extend the network's lifetime.

The following changes will be made in future work and are anticipated to increase the proposed system's decision accuracy:

- Analyze the performance of the proposed system against FGSM and JSMA attacks [29] to determine how robust the system is in an adversarial environment. We will continue to concentrate on the defense strategies that resist such attacks in the future.
- Focus on developing methods for enhancing the SATIDS techniques to achieve better results for low-record attacks and detecting zero-day attacks on IoT systems.
- Study the performance of the proposed SATIDS on a large, mobile, and heterogeneous real IoT network.

Author Contributions: Conceptualization, R.H. and S.A.E.; methodology, S.A.E.; software, R.H.; validation, R.H., R.E. and S.A.E.; formal analysis, R.H.; investigation, S.A.E. and M.A.; resources, R.H.; data curation, R.H.; writing—original draft preparation, R.E.; writing—review and editing, S.A.E. and M.H.; visualization, S.A.E. and M.A.; supervision, M.H. and M.A.; project administration, R.H. and S.A.E. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Muthanna, M.S.; Muthanna, A.; Rafiq, A.; Hammoudeh, M.; Alkanhel, R.; Lynch, S.; Abd El-Latif, A.A. Deep reinforcement learning based transmission policy enforcement and multi-hop routing in QoS aware LoRa IoT networks. *Comput. Commun.* **2022**, *183*, 33–50. [[CrossRef](#)]
2. Popoola, S.I.; Adebisi, B.; Ande, R.; Hammoudeh, M.; Atayero, A.A. Memory-efficient deep learning for botnet attack detection in IoT networks. *Electronics* **2021**, *10*, 1104. [[CrossRef](#)]
3. Choobdar, P.; Naderan, M.; Naderan, M. Detection and Multi-Class Classification of Intrusion in Software Defined Networks Using Stacked Auto-Encoders and CICIDS2017 Dataset. *Wirel. Pers. Commun.* **2022**, *123*, 437–471. [[CrossRef](#)]
4. Popoola, S.I.; Adebisi, B.; Ande, R.; Hammoudeh, M.; Anoh, K.; Atayero, A.A. Smote-drrn: A deep learning algorithm for botnet detection in the internet-of-things networks. *Sensors* **2021**, *21*, 2985. [[CrossRef](#)] [[PubMed](#)]

5. Unal, D.; Hammoudeh, M.; Khan, M.A.; Abuarqoub, A.; Epiphaniou, G.; Hamila, R. Integration of federated machine learning and blockchain for the provision of secure big data analytics for Internet of Things. *Comput. Secur.* **2021**, *109*, 102393. [CrossRef]
6. Prasad, M.; Tripathi, S.; Dahal, K. Unsupervised feature selection and cluster center initialization based arbitrary shaped clusters for intrusion detection. *Comput. Secur.* **2020**, *99*, 102062. [CrossRef]
7. Thakur, S.; Chakraborty, A.; De, R.; Kumar, N.; Sarkar, R. Intrusion detection in cyber-physical systems using a generic and domain specific deep autoencoder model. *Comput. Electr. Eng.* **2021**, *91*, 107044. [CrossRef]
8. Manimurugan, S.; Al-Mutairi, S.; Aborokbah, M.M.; Chilamkurti, N.; Ganesan, S.; Patan, R. Effective attack detection in internet of medical things smart environment using a deep belief neural network. *IEEE Access* **2020**, *8*, 77396–77404. [CrossRef]
9. Daanoune, I.; Abdennaceur, B.; Ballouk, A. A comprehensive survey on LEACH-based clustering routing protocols in Wireless Sensor Networks. *Ad Hoc Netw.* **2021**, *114*, 102409. [CrossRef]
10. Kumar, P.; Prabhat, G.P.G.; Tripathi, R. An ensemble learning and fog-cloud architecture-driven cyber-attack detection framework for IoMT networks. *Comput. Commun.* **2021**, *166*, 110–124. [CrossRef]
11. Moustafa, N. ToN_IoT Datasets, IEEE Dataport, 2019, Online. Available online: <http://dx.doi.org/10.21227/fesz-dm97> (accessed on 10 February 2020).
12. Elsayed, M.S.; Le-Khac, N.A.; Jurcut, A. InSDN: A Novel SDN Intrusion Dataset. *IEEE Access* **2020**, *8*, 165263–165284. [CrossRef]
13. Elsayed, M.S.; Le-Khac, N.A.; Dev, S.; Jurcut, A.D. Network Anomaly Detection Using LSTM Based Autoencoder. In Proceedings of the 16th ACM Symposium on QoS and Security for Wireless and Mobile Networks, Alicante, Spain, 16–20 November 2020; pp. 37–45.
14. Kumar, V.; Choudhary, V.; Sahrawat, V.; Kumar, V. Detecting intrusions and attacks in the network traffic using anomaly based techniques. In Proceedings of the 2020 5th International Conference on Communication and Electronics Systems (ICCES), Coimbatore, India, 10–12 June 2020; IEEE: New York, NY, USA, 2020; pp. 554–560.
15. Elsayed, M.S.; Jahromi, H.Z.; Nazir, M.M.; Jurcut, A.D. The role of CNN for intrusion detection systems: An improved CNN learning approach for SDNs. In *International Conference on Future Access Enablers of Ubiquitous and Intelligent Infrastructures*; Springer: Cham, Switzerland, 2021; pp. 91–104. Available online: https://link.springer.com/chapter/10.1007/978-3-030-78459-1_7 (accessed on 6 December 2022).
16. Jayalaxmi, P.L.S.; Kumar, G.R.S.; Conti, M.; Kim, T.-H.; Thomas, R. DeBot: A deep learning-based model for bot detection in industrial internet-of-things. *Comput. Electr. Eng.* **2022**, *102*, 108214. [CrossRef]
17. Sugitha, G.; Solairaj, A.; Suresh, J. Block chain fostered cycle-consistent generative adversarial network framework espoused intrusion detection for protecting IoT network. *Trans. Emerg. Telecommun. Technol.* **2022**, e4578. [CrossRef]
18. Mohamed, R.H.; Mosa, F.A.; Sadek, R.A. Efficient Intrusion Detection System for IoT Environment. *Int. J. Adv. Comput. Sci. Appl.* **2022**, *13*. [CrossRef]
19. Ravi, V.; Chaganti, R.; Alazab, M. Recurrent deep learning-based feature fusion ensemble meta-classifier approach for intelligent network intrusion detection system. *Comput. Electr. Eng.* **2022**, *102*, 108156. [CrossRef]
20. Kumar, R.; Kumar, P.; Tripathi, R.; Gupta, G.P.; Gadekallu, T.R.; Srivastava, G. SP2F: A secured privacy-preserving framework for smart agricultural Unmanned Aerial Vehicles. *Comput. Netw.* **2021**, *187*, 107819. [CrossRef]
21. Siddiqi, M.A.; Pak, W. Optimizing filter-based feature selection method flow for intrusion detection system. *Electronics* **2020**, *9*, 2114. [CrossRef]
22. Bugata, P.; Drotar, P. On some aspects of minimum redundancy maximum relevance feature selection. *Sci. China Inf. Sci.* **2020**, *63*, 112103. [CrossRef]
23. Billah, M.; Waheed, S. Minimum redundancy maximum relevance (MRMR) based feature selection from endoscopic images for automatic gastrointestinal polyp detection. *Multimed. Tools Appl.* **2020**, *79*, 23633–23643. [CrossRef]
24. Pourpanah, F.; Wang, R.; Lim, C.P.; Wang, X.Z.; Yazdani, D. A review of artificial fish swarm algorithms: Recent advances and applications. *Artif. Intell. Rev.* **2022**. [CrossRef]
25. Osama, A.; El-Said, S.A.; Hassanien, A.E. Energy-efficient routing techniques for wireless sensors networks. In *Sensor Technology: Concepts, Methodologies, Tools, and Applications*; IGI Global: Hershey, PA, USA, 2020; pp. 917–944.
26. Hamza, N.M.; El-Said, S.A.; Attia, E.R.M.; Abdalla, M.I. Energy aware optimized hierarchical routing technique for wireless sensor networks. In Proceedings of the International Conference on Advanced Machine Learning Technologies and Applications, Cairo, Egypt, 22–24 February 2018; Springer: Cham, Switzerland, 2018; pp. 614–623.
27. Al-Nasser, A.; Almesaeed, R.; Al-Junaid, H. A Comprehensive Survey on Routing and Security in Mobile Wireless Sensor Networks. *Int. J. Electron. Telecommun.* **2021**, *67*, 379–384.
28. Kumar, P.; Kumar, R.; Srivastava, G.; Gupta, G.P.; Tripathi, R.; Gadekallu, T.R.; Xiong, N.N. PPSF: A privacy-preserving and secure framework using blockchain-based machine-learning for IoT-driven smart cities. *IEEE Trans. Netw. Sci. Eng.* **2021**, *8*, 2326–2341. [CrossRef]
29. Welcome to the Adversarial Robustness Toolbox. Available online: <https://adversarial-robustness-toolbox.readthedocs.io/en/latest/> (accessed on 6 December 2022).

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.