Intelligent Automation & Soft Computing
DOI:10.32604/iasc.2022.022856
Article

Tech Science Press

Detection of DDoS Attack in IoT Networks Using Sample Selected RNN-ELM

S. Hariprasad[1,*], T. Deepa[1] and N. Bharathiraja[2]

[1]SRM Institute of Science and Technology, Kattankulathur, 603203, Tamil Nadu, India
[2]Vel Tech Multi Tech Dr. Rangarajan Dr. Sakunthala Engineering College, Chennai, 600062, Tamil Nadu, India
*Corresponding Author: S. Hariprasad. Email: haripras@srmist.edu.in
Received: 20 August 2021; Accepted: 21 December 2021

Abstract: The Internet of Things (IoT) is a global information and communication technology which aims to connect any type of device to the internet at any time and in any location. Nowadays billions of IoT devices are connected to the world, this leads to easily cause vulnerability to IoT devices. The increasing of users in different IoT-related applications leads to more data attacks is happening in the IoT networks after the fog layer. To detect and reduce the attacks the deep learning model is used. In this article, a hybrid sample selected recurrent neural network-extreme learning machine (hybrid SSRNN-ELM) algorithm that uses recurrent neural network (RNN) as a supervised and extreme learning machine (ELM) classifier as unsupervised. In the proposed algorithm sample selected features are extracting from the original dataset using linear regression with recursive feature extraction (LR-RFE) and sequence forward selector (SFS) then RNN is used to learn the behavior of the important features and at end layer the ELM classifier is used. This hybrid intrusion detection algorithm is placed in between the fog layer and its devices. NSL_KDD benchmark is used for detecting the distributed denial-of-service (DDoS) attack in IoT devices after the fog node. The proposed hybrid SSRNN-ELM model exposes the attacks while testing with enhanced accuracy of up to 99% from NSL-KDD data set. Experimental results outperform by using proposed technique when compared with the existing models.

Keywords: Distributed denial of service; internet of things; sample selected; linear regression-recursive feature extraction; sequence forward selector; recurrent neural network; extreme learning machine

1 Introduction

The Internet of Things (IoT) device is a resource-limited system that sense or transmit any information to the internet [1]. Some of the IoT applications are autonomous driving, unmanned mining, smart power grids, health care, and emergency alert systems etc., [2]. To avoid unauthorized user data, transfer the radio frequency identification (RFID) [3] card-based identification is used to communicate the sensor data to the actuator. Many IoT devices can connect with sensors and communicate to actuator with the help of internet to monitor or control any applications. Due to the heterogeneity, larger size hardware and

resource limited the IoT security has become difficult [4]. In smart health care system security breaches can result in degraded performance of IoT devices and degrade life-threatening damages. If any of the IoT device attempts to connect to the target system without validating its own identity is referred as an unauthorized device. Such attempts are called as network classification attack; it can be identified by machine learning algorithms such as K-Means, Random Forest (RF), Decision Tree (DT), Support Vector Machine (SVM), and Bayesian Network [5].

The behavior of the attack model helps locate threat models and determine which attack. The patterned behavior of attacks can be determined by features collected, including the total of bytes sent and received, link time, error count, and the total amount of requests. Manual feature engineering is required in [6]. Deep learning is used to solve the feature extraction problem in ML [7]. Due to its multilayer structure, a profound representation of the raw data can be given to DL and classify or predict the data more accurately relative to ML [8–10]. The DL algorithm has been applied to the categorization of IoT cyber-attacks. Fog computing has been developed for implementation with DL models directly by processing IoT devices to store and analyze vast amounts of low latency and fast response time data on fog nodes [11,12]. The idea is to transfer data using sensors on the layer of edge towards the nearest data source where little time is required to analyze the data using DL.

Security Information and Event Management (SIEM) solution [13] based approaches are used to detect the DDOS attack in gateway and monitoring IoT network. The SIEM solution includes parsing/indexing/secure storage and detection of DDOS attack is based on comparing the number of packets of types such as SYN, ICMP, or DNS. In [14], Deep neural network hyperparameter optimization is an enhancement with three top layers such as the input layer, which has the features in the CICDDoS2019 dataset, the hidden layer is responsible for learning the complex information from the elements, and the output layer is liable to classify the attack label. Intrusion detection system [15] approach with 25 features from the real-time traffic such as source address, destination address and packet ID. Neural Network, Naive Bayes, and Random Forest, KNN, and SVM, datasets processed have different percentages, intending to facilitate classifying the attacks. The detection of normal data packet flow and DDoS attacks packet flow on networks is used variety of training function schemes and hidden layer architecture [16]. This Artificial neural network (ANN) architecture involves several steps such as Extracting data packet flow features, Training the ANN network, and evaluating the performance of ANN. The linear and mesh network topology with various sizes can handle the Distributed denial of service attack without managing channel bandwidth in a software-defined network (SDN) [17]. The SDN uses a round-robin algorithm to load-balance all connections equally between controllers. The Recurrent Neural Network Software Defined Regulation (RNN-SDR) technique is performed by the Software-Defined Network Controller (SDN) *vs.* the NSL-KDD dataset [18]. The RNN is the technique used to classify and analyze data sequences using the RNN-SDR. It shows better results in learning sequences and improves the identification of abnormalities in an SDN environment. In literature [19], a sample selected extreme learning machine (SS-ELM) algorithm is proposed, which combines a sample selection of features and according to the network features of fog computing (FC)/Mobile edge computing (MEC) using the ELM. This approach helped to resolve the issues caused by resource limitations in FC/MEC. The existing security approach that uses DL to track IoT attacks is discussed in this section. Also, the ability to detect and remove latent features in different network settings, various attacks from various DL models is studied. Because of the rapid usage of IoT applications, cyber-attacks have been increased tremendously over the last ten years [20]. Attackers often use cyber-attacks can hack thousands of globally connected IoT devices. For instance, certain websites that use the “Dyn” DNS provider were targeted with the help of a DDoS attack in 2016.

The botnet malware attack is executed in many IoT applications. A decentralized security solution that is continuously controlling IoT devices is needed to detect zero-day attacks. Several known organizations,

including Facebook, Yahoo! based on DL, Twitter, and YouTube, have built many applications to track and analyze enormous amounts of data produced by billions of users [21]. With the new rise of the accessibility of big data, the graphical processing units (GPUs) with DL models are trained by using powerful learning algorithms. LSTM detection-based DL has been applied to analyze traffic on the network by botnet attack detection [22]. For detecting botnets, the behavior of connection between devices with IoT is analyzed. The dataset is spliced into two, namely training and testing the detection model for these two separate datasets were used one is labeled, and another unlabeled is tested with LSTM output of different features using the dataset. In the literature [23], attack detection using a deep network focused on LSTM is fog-to-things was performed. A distributed method is implemented for obtaining parameter initialization and updating from the coordinating node using fog nodes. The LSTM model was trained by each fog node based on the parameters obtained the bias and weight values were sent from one coordinating node to the other coordinating node. After that, it computes the aggregate parameters sent back to the fog nodes. A suggested model shows that a typical ML outperforms DL models in terms of accuracy and scalability compared to a centralized technique. Deep autoencoders are unsupervised DL models that are used to create feature learning for models of detection. An ensemble learning system has been merged with different DL models for learning and detection development. A distributed monitoring framework for attacks is used to identify network traffic at the edge layer; the extreme learning machine (ELM) classifier has been introduced. Besides all-comprehensive resource activities, the model training and development were transferred into the cloud layer using the HPC cluster [24]. Model training was conducted using data anonymized obtained from devices in the edge layer using CNN and LSTM schemes [25]. Then the model of activity on edge servers is used by classifiers. The tests showed that CNN and LSTM had improved accuracy with 98%, 74%, and 95%, respectively, in scanning, contact, and infected host scenarios. However, because of the limited computing, energy capabilities, and storage in IoT devices, the direct deployment of complicated DL models to detect attacks in devices is complex. Within this framework, we have proposed a hybrid SSRNN-ELM DL model.

The main contributions of this work are as follows: A hybrid SSRNN-ELM model is proposed to detect intrusion into IoT networks with great accuracy and less computational complexity. The proposed methodology is verifying the network data and its activity in the actual network in the IoT networks against DDoS attacks. An improved sample selected features are obtained from the original dataset using linear regression with recursive feature extraction (LR-RFE) and sequence forward selector (SFS). With the NSL-KDD data set, the benchmark tested the SSRNN-ELM model is tested and the results are compared with existing solutions. The proposed model achieves high accuracy with less time demonstration compared to other models. This demonstrates an improved model for critical latency in IoT devices by using SSRNN-ELM.

The remaining part of the paper is arranged as follows: Section 2 provides research methods of the proposed methodology; Section 3 presents the results gathered during the analysis and the conclusion is presented in Section 4.

2 Research Method

The proposed architecture consists of three levels (i) data set configuration, (ii) feature data discrimination and (iii) attack detection as described in Fig. 1. The NSL-KDD dataset is used in the proposed hybrid SSRNN-ELM to detect the DDoS attack detection by analyzing the network packets of normal and abnormal packets. The data features NSL-KDD is configured with trained and test dataset given to the feature data discriminators. The feature data discriminator has two parts namely full features data discriminator and sample selected feature data discriminator. For the sample selected features here two feature selection methods are used such as linear regression-recursive feature elimination (LR-RFE)

and sequence forward selector (SFS). LR-RFE is a less computational and less complex feature selection with feature weight coefficients it is based on the tree structure algorithms to reduce the features by using recursively. In the proposed sample selected the LR-RFE and SFS to gather the important features from the NSL-KDD dataset and the output is delivered into recurrent neural network (RNN) and other machine learning algorithms. Finally, the ELM classifier is to classify the normal and DDoS attack.

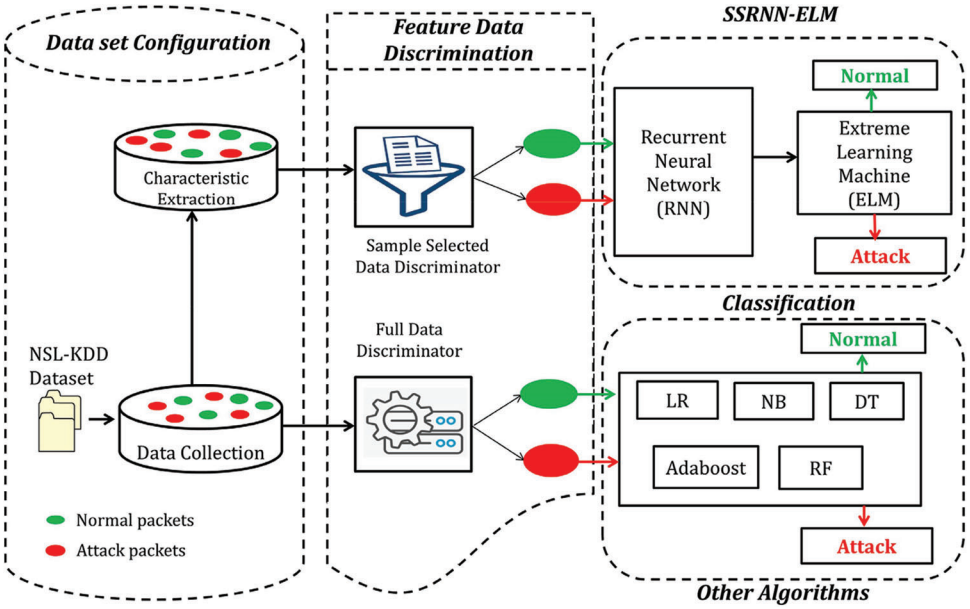

Figure 1: Framework of hybrid SSRNN-ELM

The proposed model has four modules that run together as shown in Fig. 2. Each module's details are discussed below.

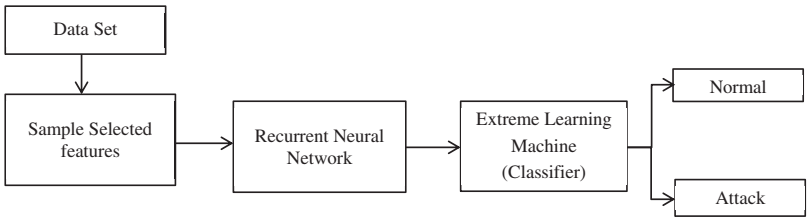

Figure 2: Modules of hybrid SSRNN-ELM method

2.1 Dataset

One of the best publicly available dataset for DoS attack is NSL KDD dataset with the implementation of attack detection according to the requirement is possible. NSL-KDD is derived from the KDD CUP 99 [26] dataset the dimension is reduced in the updated dataset. The updated NSL-KDD dataset can able to detect some types of DDoS attack only such as neptune, smurf, apache2, teardrop etc.,. The NSL KDD has 42 features as described in the Tab. 1. These 42 features if fed into characteristic extortion (sample selected) to obtain some important feature.

Table 1: Feature description of NSL-KDD dataset

Feature number	Feature attribute	Feature number	Feature attribute	Feature number	Feature attribute
1.	duration	15.	su-attempted	29.	same_srv_rate
2.	type_protocol	16.	num_root	30.	diff_srv_rate
3.	service	17.	num_file_creations	31.	srv_diff_host_rate
4.	flag	18.	num_shell	32.	dst_host_count
5.	src_byte_data	19.	num_access_files	33.	dst_host_srv_count
6.	dst_byte_data	20.	num_outbound_cmds	34.	dst_host_same_srv_rate
7.	land	21.	is_host_login	35.	dst_host_diff_srv_rate
8.	wrong_fragment	22.	is_guest_login	36.	dst_host_same_src_port_rate
9.	urgent	23.	count	37.	dst_host_srv_diff_host_rate
10.	host	24.	srv_count	38.	dst_host_serror_rate
11.	num_failed_login	25.	serror_rate	39.	dst_host_srv_serror_rate
12.	logged_in	26.	srv_error_rate	40.	dst_host_rerror_rate
13.	num_compromised	27.	rerror_rate	41.	dst_host_srv_rerror_rate
14.	root_shell	28.	srv_rerror_rate	42.	class

2.2 Characteristic Extraction (Sample Selected Feature Extraction)

The most important stage in the DDoS attack detection method is characteristic extraction. The performance is highly improved with a collection of strong features to reflect the attack behavior. In characteristic extraction the attack behaviors are thoroughly analyzed on the application layer of IoT networks to detect the DDoS attacks. DDoS attackers have generally categorized them into four. The first is a repetitive attack on a single IoT networks. This form of attack sends a lot of repeated requests at a fixed speed or random speed to the target. In second repeated attack from the multiple IoT networks. In this form, attackers submit a lot of server data is send to IoT devices with fix or variable speed. In the third randomly select the IoT devices for attack. This form of attack software will search the IoT device with less authorization and authentication of the target and then initiate an attack by randomly selecting the IoT device as the target. In the fourth randomly select and repeated multiple attack session. Traditional methods could not be able to detect this kind of DDoS attack because they mask themselves by simulating typical user behaviors and submit attack requests repeatedly to IoT devices. With the help of two feature selection methods such as recursive feature elimination (RFE) and sequence feature selector (SFS) are used to obtain the important feature from the original NSL-KDD dataset.

2.2.1 Linear Regression–Recursive Feature Elimination (LR-RFE)

Recursive Feature Elimination (RFE) is a type of feature selection algorithm. An algorithm that selects a subset with most relevant features form the original dataset is referred as feature selection algorithm. This fewer feature can reduce the time complexity of machine learning algorithms and more effective also. The machine learning may not work properly with irrelevant feature to select the most relevant and best feature the linear regression recursive feature elimination (LR-RFE) method is used. It is a type of wrapper-style feature selection which means regression model is combined with RFE for selects the features. The linear regression model is used to ranking the feature by its importance and removes the

least important features; the RFE collects the subset of features with specified number of features remains as described in the Algorithm 1. After using the LR-RFE most important 25 features are extracted from the original NSL-KDD dataset and the import feature number are 2, 4, 7, 8, 10, 11, 12, 13, 14, 15, 16, 22, 25, 26, 27, 28, 29,30, 31, 35, 36, 37, 38, 39, 41 these feature attributes are shown in Tab. 1.

Algorithm 1: LR-RFE Algorithm

Input: TD (X_{tt},Y_{tt}) , Where X_{tt} = [X_{tt1}, X_{tt2} … , X_{ttn}] is the Input features, Y_{tt} = Output Class, n = (1,2, …N) is Feature number and TD is the total NSL-KDD dataset

Output: SS(X_r,Y_r), Where SS is the subset of sample selected features

```
S = [ ]
model = linearregression(X,Y)
rfe = RFE(model)
X_r = rfe.fit_transform(X,Y)
model.fit(X_r)
S = [rfe.index == True]
X_r = X.loc[:,S]
Y_r = Y.loc[:,S]
return SS(X_r,Y_r)
```

2.2.2 Sequential Feature Selection

Sequential feature selection based on wrapper type method it adds and removes the features sequentially from the dataset. It evaluates the individual features and selects the N features from the entire dataset based on the best on the scores. Sequential feature selection has four sub types of namely sequential forward selector (SFS), sequential backward selection (SBS), sequential forward floating selection (SFFS) and sequential backward floating selection (SBFS). For linear variants features the sequential forward selector and sequential backward selector is used whereas floating variants features sequential forward floating selection and sequential backward floating selection is used. Two components involved in SFS one is an objective function which find to minimize the number of overall features to form the subset of features. Other is sequential search algorithm which adds or removes the features by evaluating the objective function criterion. This search algorithm will follow one direction to reduce the number of features from the original feature set as described in Algorithm 2. After using the SFS feature selection methods the most important 30 features are selected form the original NSL-KDD dataset and the features numbers are 1, 2, 3, 4, 6, 8, 10, 11, 12, 14, 19, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 41 these feature attributes are shown in Tab. 1.

Algorithm 2: SFS Algorithm

Input: TD (X_{tt},Y_{tt}), Where X_{tt} = [X_{tt1}, X_{tt2} …. , X_{ttn}] is the Input features, Y_{tt} = Output Class, n = (1,2, …N) is Feature number and TD is the total NSL-KDD dataset

Output: SS1(X_r,Y_r), Where SS1 is the subset from the SFS

Initialization: X_r = NULL, r = 0

```
S1 = [ ]
```

(Continued)

Algorithm 2 (continued)

2 **do**
3 S1 = arg max TD($X_{tt} - x$), Where $x \in X_r$
4 $X_{tt\text{-}n} = X_{tt} - S1$
5 n = n-1
6 **until S1 converges**
7 X_r = X.loc[:,S1]
8 Y_r = Y.loc[:,S1]
9 return SS1(X_r,Y_r)

2.3 Attack Detection

Once the optimized dataset is obtained after the feature selection the data preprocessing is invoked to bring the data on refined form. To preprocess the data, normalization and standardization are used. In normalization, each sample inputs to unit norms based on the cosine similarity of the features of the range between minimum and maximum values. In standardization the data is uniform distributed with Gaussian zero and mean variance. After the preprocessing, by using the machine learning algorithms and proposed SSRNN-ELM algorithm used generate the model and to detect the DDoS attack. Further the RNN, SSRNN, ELM and finally SSRNN-ELM are discussed below.

2.3.1 RNN

RNN is a category of neural networks that allow the use of previous outputs as inputs while hidden states. It has many types like one to one, one to many, many to one, and many to many concerning the input and output. To calculate the current state of the output feature, calculating the activation layer and calculating the following formula were used.

$$q_t = f(q_{t-1}, a_t) \tag{1}$$

where q_t - current state, q_{t-1} - previous state, a_t - input state

$$q_t = tanh(W_{hh}q_{t-1}, W_{xh}q_t) \tag{2}$$

where: w_{hh} - weight at recurrent neuron, w_{xh} - weight at input neuron

$$o_t = W_{hy}q_t \tag{3}$$

where: y_t - output W_{hy} - weight at output layer

2.3.2 Sample Selected RNN (SSRNN)

In RNN inputs and outputs of all data are independent concerning each other and when it is required for output status of attacks to predict it takes the previous output features and therefore is always needed to know the previous data on attacks. Thus, sample selected features LR-RFE and SFS to RNN input layer (SSRNN) is employed with number hidden layer to resolve the simple RNN problem. The most important aspect of SSRNN is the output state which knows the information about all the input and output features must be stored in the memory. It employs the same parameters for all inputs as for the output or hidden layers to generate the same task. In contrast to other neural networks, this minimizes the complexity of parameters and reduction in time.

2.3.3 SSRNN-Extreme Learning Machine (ELM)

Extreme Learning Machines (ELM) is feed forward neural networks. It does not require gradient-based back propagation to work. The algorithm for traditional neural network classification is the slow speed in training the data and over-fitting problems when compared to the ELM. ELM is based on the principle of empirical risk reduction and only requires a single iteration of its learning process. Local minimization and multiple iterations are avoided by the ELM algorithm. The ELM algorithm is configured to set the hidden layer nodes in more numbers hence it does not need to alter the input weights of the network and a hidden bias in the implementation process, and it creates a specific best solution with the reduced time complexity and high performance. With the ELM classification and further incorporated at RNN output layer with ensemble feature selection algorithms. By using a trained SSRNN-ELM model the attack is detected. Algorithm 3 provides detailed steps for SSRNN-ELM training.

Algorithm 3: SSRNNELM Data Training

Input: SS(x_{tt},y_{tt}), SS1(x_{tt1},y_{tt1}) Where X_{tt} = [X_{tt1}, X_{tt2} …., X_{ttn}] is the Input features, Y_{tt} = Output Class, n = (1,2,…N) is Feature number and SS is the sample selected features using LR-RFE and SS1 is the sample selected features using SFS

Output: Training model

Initialize layers (l), count of neurons (H), bias(B), weight (W), learning rate (η), epoch (E); accuracy = { }

Observed ← Input SS,SS1 to RNNELM

Accuracy ← (#(actual == observed) ÷ N) × 100

if (accuracy is not satisfactory)

  change parameters = E, η, H, W

else

  return E, η, H, W

set weight←random, bias B_i where i = 1,2,…, L.

rnnmodel (r) = sequential()

rnnmodel.add(RNN(16))

rnnmodel.add(dropout(0.1))

rnnmodel.add(RNN(16))

rnnmodel.add(dropout(0.1))

rnnmodel.add(dense(1))

rnnmodel.add(activation(sigmoid))

hiddenlayerelm (T) = model(input = rnnmodel.layers [0].output, output = rnnmodel.output)

Calculate Matrix $H = \begin{matrix} r(w_1, b_1, x_1) & \ldots. & r(w_L, b_L, x_L) \\ . & . & . \\ r(w_1, b_1, x_N) & \ldots. & r(w_L, b_L, x_N) \end{matrix}$

Calculate $H_{New} = \dfrac{|H_{ij}|}{\sum_{i,j=1}^{N} |H_{ij}|}$

Calculate Output Layer weight β = H + T

Calculate the Reconstruction error $OF_i = \dfrac{1}{N} \sum_{j=1}^{n} (x_{ij} - t_{ij})^2$

elmmodel(Y) = ELM(T,β)

return trained_model (Yj) from higher to lower

2.3.4 Final Attack Prediction

The trained model is generated by using the input collected data. Algorithm 4 demonstrates the final detection process of the attack. When the model categorizes the regular traffic, it continues with the normal operation. If the traffic and the model categories are abnormal the traffic as an attack immediately activates the module mitigation. The outputs from the model produce the warning "?" immediately sends the mirrored traffic to detect the attack.

Algorithm 4: Detection of Attack

Input: Traffic Features Extracted (TFE)
Output: Attack Label (AL)
1 AL Trained Model (TFE)
2 if (AL == normal) then Continue with normal operation
3 if (AL == attack class) then Return AL
4 if (AL == ?) then Alarm/Create alert. Return AL

3 Results and Discussion

The efficiency of the proposed SSRNN-ELM model and other ML algorithms is achieved using the original and optimized NSL-KDD data set. Initially the NSL-KDD dataset consists of 42 attributes such as time, urgency, bytes, bytes, count, etc. The datasets comprise normal traffic instances and various traffic attacks class instances such as Teardrop, Land, Rootkit, Port Sweep, SPY, IMAP, etc. falling into one of the four attack classes, namely DDoS, Probing, User to Root, and Remote to Local. The features are symbolic or numerical. After the sample selected features by using LR-RFE and SFS the feature is reduced to 25 and 30 features respectively. The optimized NSL-KDD dataset consists of 4,94,021 training samples and 3,11,029 testing sample. Experimental simulation has been carried out using python 3.6, keras, tensor flow, sci-kit learn, numpy and pandas.

In this work the assessment is carried out by general performance metrics such as accuracy, attack detection rate, the attack predictive value, normal prediction value and F1 score. Here true positive (TP), true negative (TN), false positive (FP) and false negative (FN). Finally, evaluate the degree of accuracy of measurements as presented and compared to the literature technique with accuracy, attack detection rate, attack predictive, normal predictive value and F1 score.

3.1 Confusion Matrix

The confusion matrix is an error matrix used to define a classification model (or classifier) performance on a test data set with the true values. It makes it possible to visualize the performance of an algorithm. This enables for easier definition of class confusion, e.g., one class is often mislabeled as the other. The key to the confusion matrix is the number of correct and erroneous forecasts summed with count values and not only the number of errors made by each class is shown in Fig. 3.

3.1.1 Accuracy-(ACC)

The percentage of all requests correctly defined over all the data is referred as Accuracy and the formula is shown as follows.

$$ACC = \frac{TP + TN}{TP + FP + TN + FN} \tag{4}$$

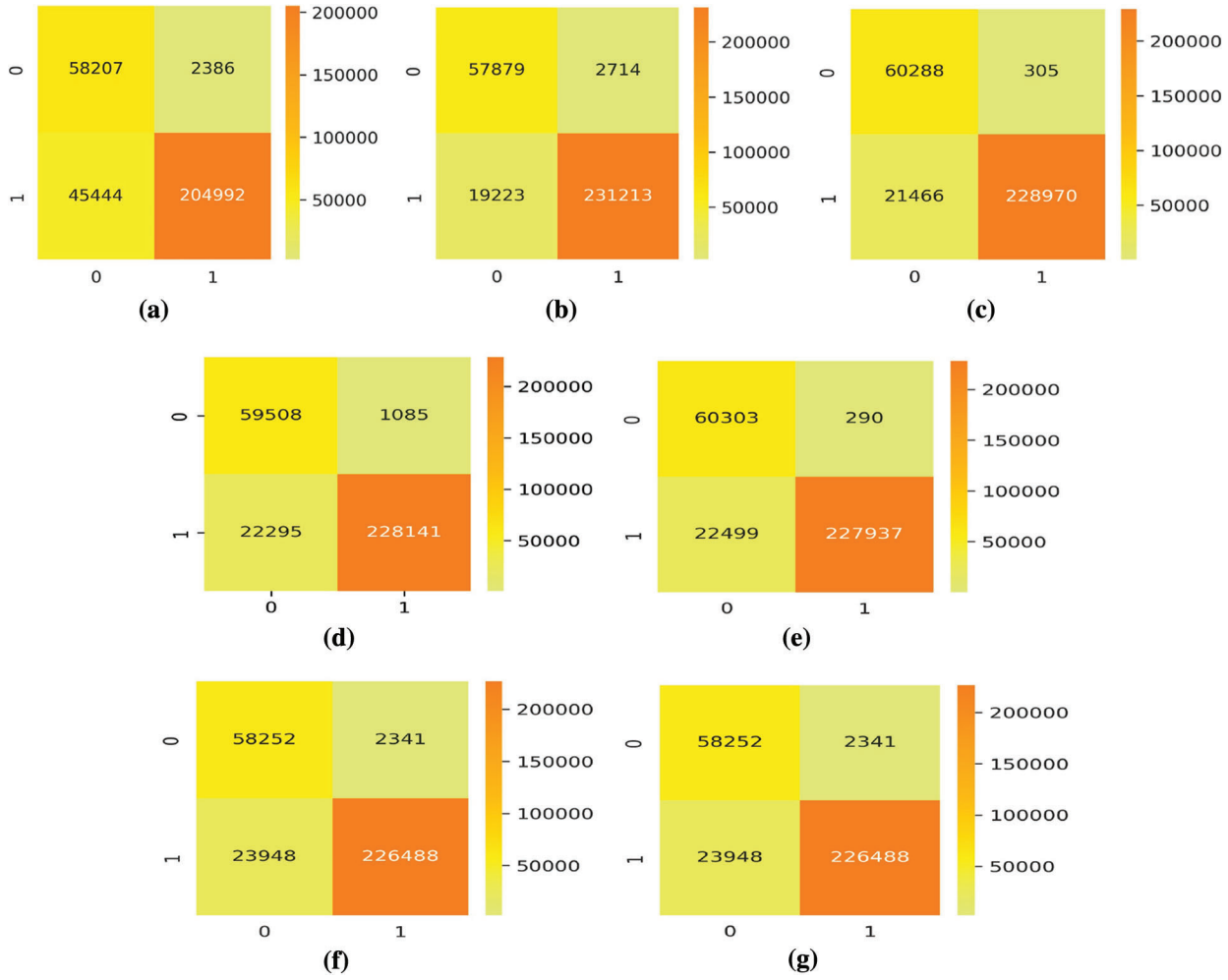

Figure 3: Confusion matrix for (a) LR, (b) NB, (c) DT, (d) AdaBoost, (e) RF, (f) RNN (g) RNNELM

3.1.2 Detection Rate-(DR)

This percentage of the ratio between total attack traffic instances to the correctly identified attack traffic instances is referred as detection rate and the formula is as follows.

$$DR = \frac{TP}{TP + FN} \tag{5}$$

3.1.3 Attack Predictive Value-(APV)

The percentage of the ratio between attack traffic instances correctly detected among the expected cases as an attack.

$$APV = \frac{TP}{TP + FP} \tag{6}$$

3.1.4 Normal Predictive Value-(NPV)

This refers to the percentage of normal traffic instances correctly detected among the real normal traffic instances.

$$NPV = \frac{TN}{FP + TN} \tag{7}$$

3.1.5 F1-Score

The F1 Score rate is the ratio of normal requests that are detected as attacks overall normal requests as follows.

$$F1\ score = \frac{2 * TP}{(2 * TP + FP + FN)} \tag{8}$$

The performance metrics without sample selected features on various ML algorithms with accuracy in LR is 84.62%, NB 92.94%, DT is 93.0%, Adaboost is 92.48%, RF is 92.67%, RNN is 91.54% and RNNELM is 91.54% as depicted in Fig. 4 and the other performance measure such as DR, APV, NPV, F1-Score is also mentioned in Tab. 2.

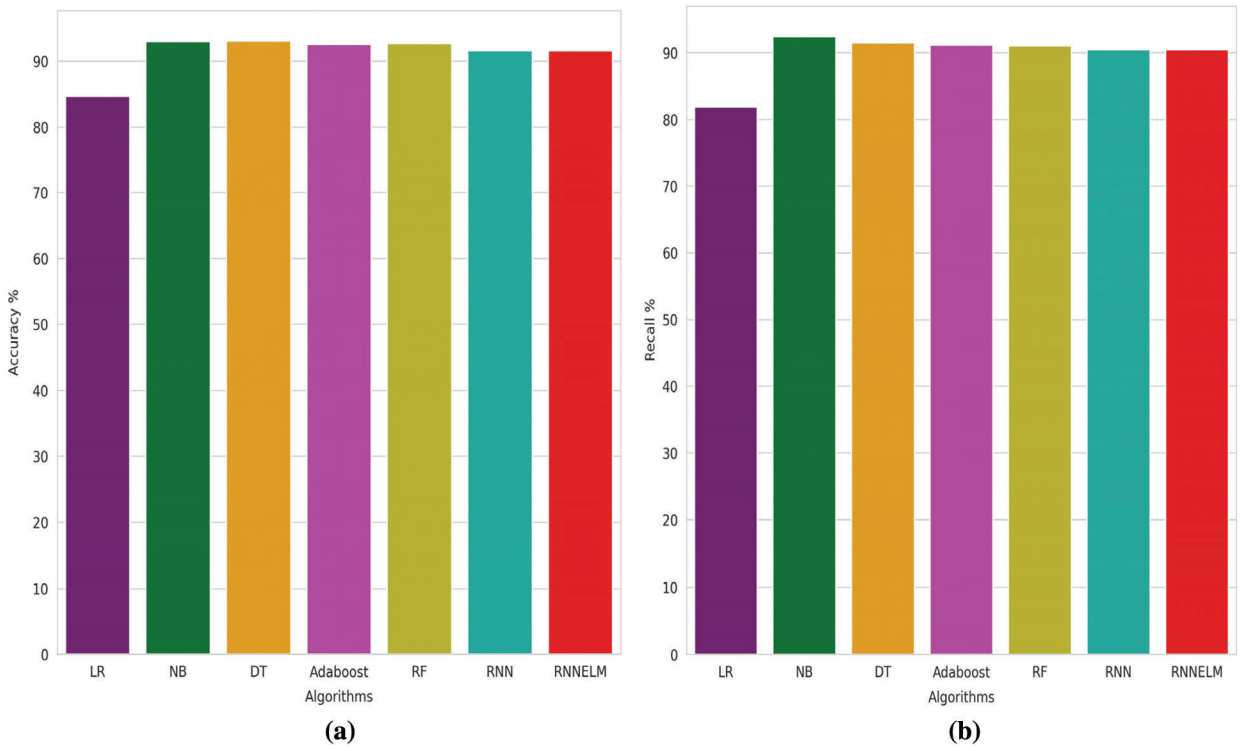

Figure 4: Performance metrics of without sample selected features (a) accuracy (b) detection rate using proposed RNNELM and other ML algorithms

The performance metrics with sample selected features using LR-RFE on various ML algorithms with accuracy in LR is 91.63%, NB 91.42%, DT is 92.83%, Adaboost is 92.61%, RF is 98.06%, RNN is 99.27% and RNNELM is 91.54% as depicted in Fig. 5 and the other performance measure such as DR, APV, NPV, F1-Score is also mentioned in Tab. 3. The performance metrics with sample selected features using SFS on

various ML algorithms with accuracy in LR is 88.73%, NB 91.42%, DT is 92.8%, Adaboost is 92.38%, RF is 92.51%, RNN is 98.44% and RNNELM is 99.25% as depicted in Fig. 6 and the other performance measure such as DR, APV, NPV and F1-Score is also mentioned in Tab. 4. Hence, the proposed SSRNN-ELM has better performance than the other conventional algorithms.

Table 2: Performance metrics using without sample selected feature selection

Algorithm	Accuracy	DR	APV	NPV	F1-Score
Logistic regression	84.62	0.818	0.988	0.960	0.896
Naïve bias	92.94	0.923	0.988	0.955	0.955
Decision tree	93.0	0.914	0.998	0.994	0.955
Ada Boost	92.48	0.910	0.995	0.982	0.951
Random Forest	92.67	0.910	0.998	0.995	0.952
RNN	91.54	0.904	0.989	0.961	0.945
RNN-ELM	91.54	0.904	0.989	0.961	0.945

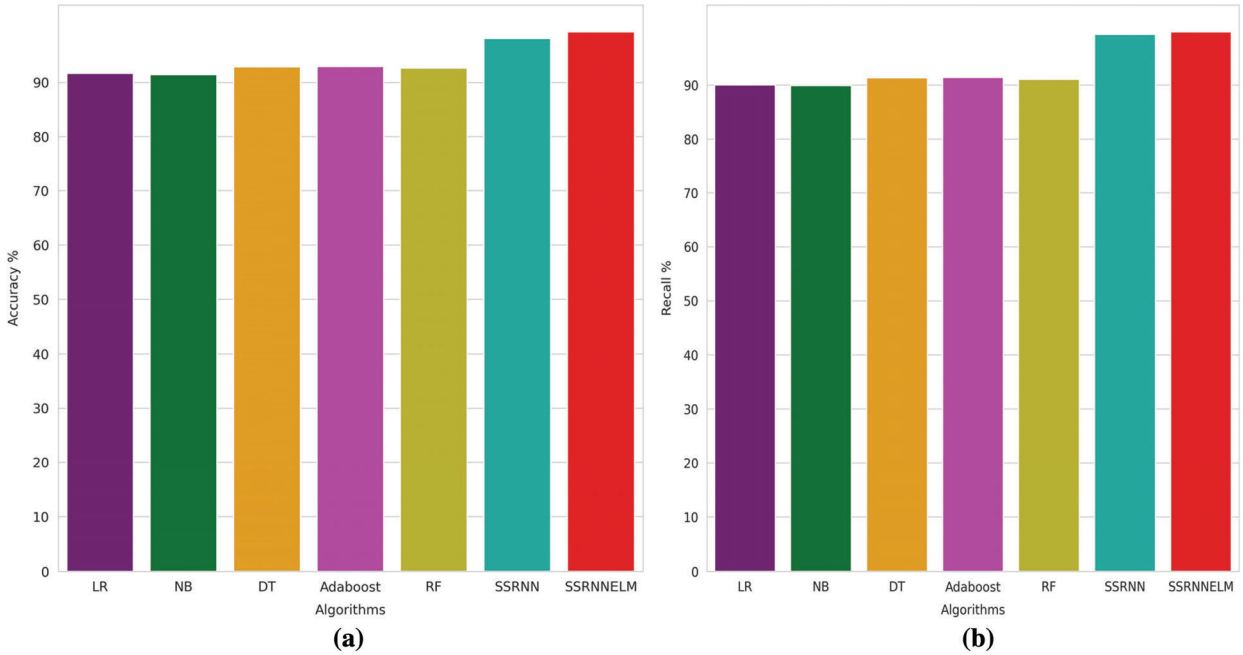

Figure 5: Performance metrics of sample selected features using LR-RFE (a) accuracy (b) detection rate using proposed RNNELM and other ML algorithms

The proposed SSRNN-ELM showing the difference in accuracy with the existing method is up to ~5% due to the sample selected features with ELM classifier.

Table 3: Performance metrics using LR-RFE

Algorithm	Accuracy	DR	APV	NPV	F1-Score
Linear regression	91.63	0.90	0.995	0.984	0.945
Navie Bayes	91.42	0.898	0.994	0.977	0.944
Decision Tree	92.83	0.913	0.997	0.989	0.954
Ada Boost	92.89	0.913	0.997	0.991	0.954
Random Forest	92.61	0.91	0.997	0.99	0.952
SSRNN	**98.06**	**0.993**	**0.982**	**0.927**	**0.988**
SSRNN-ELM	**99.27**	**0.997**	**0.992**	**0.971**	**0.995**

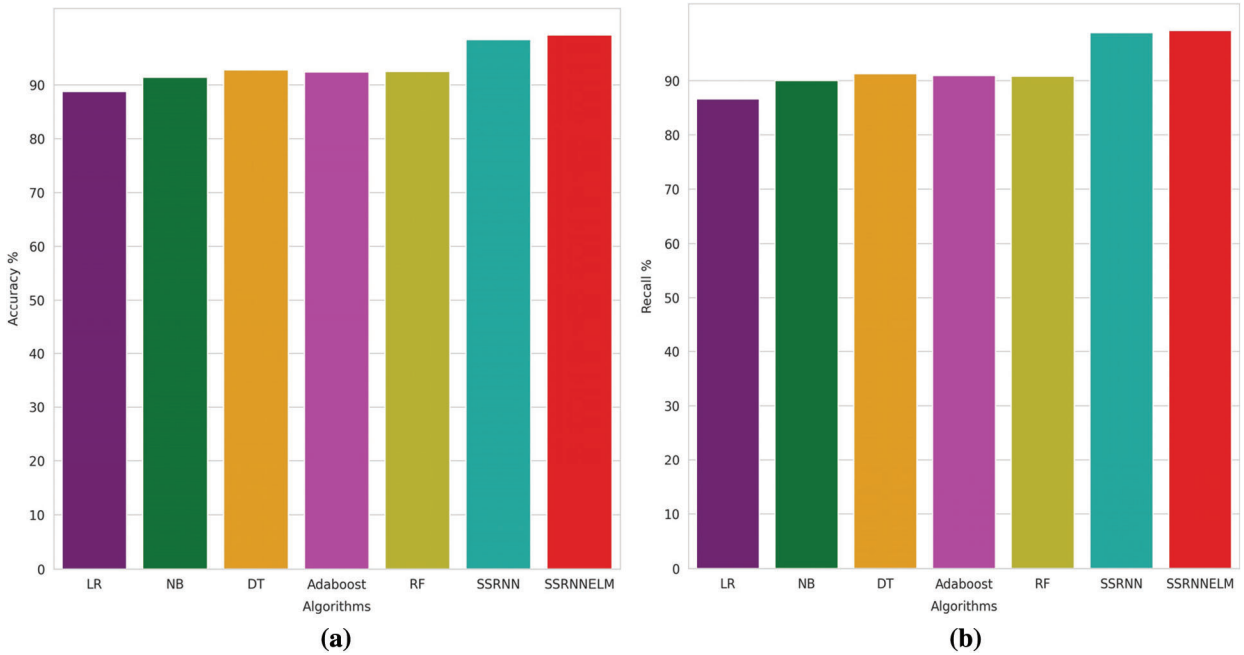

Figure 6: Performance metrics of sample selected feature with SFS (a) accuracy (b) detection rate using proposed RNNELM and other ML algorithms

Table 4: Performance metrics using SFS

Algorithm	Accuracy	DR	APV	NPV	F1-Score
Linear regression	88.73	0.866	0.993	0.974	0.925
Navie Bayes	91.42	0.899	0.992	0.973	0.944
Decision Tree	92.80	0.911	0.998	0.994	0.953
Ada Boost	92.38	0.909	0.996	0.984	0.951
Random Forest	92.51	0.908	0.998	0.995	0.951
SSRNN	**98.44**	**0.988**	**0.992**	**0.969**	**0.990**
SSRNN-ELM	**99.25**	**0.991**	**0.999**	**0.996**	**0.995**

Fig. 7 presents the best accuracy, best F1 Score, the best detection rate of the attack, and best attack prediction value by using various machine learning algorithms. This comparison will help make the best-accuracy attack model mitigate the attack easily with high accuracy. Fig. 8 presents the time complexity can be seen as the measure of how fast or slow an algorithm will perform for the input size it is always given concerning some input size.

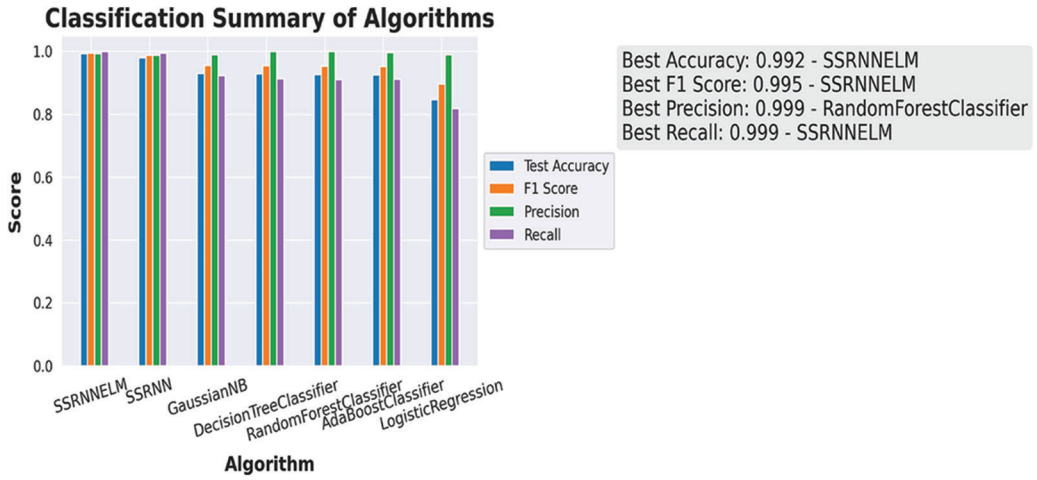

Figure 7: Graph for accuracy, F1 Score, and precision, recall using various algorithms

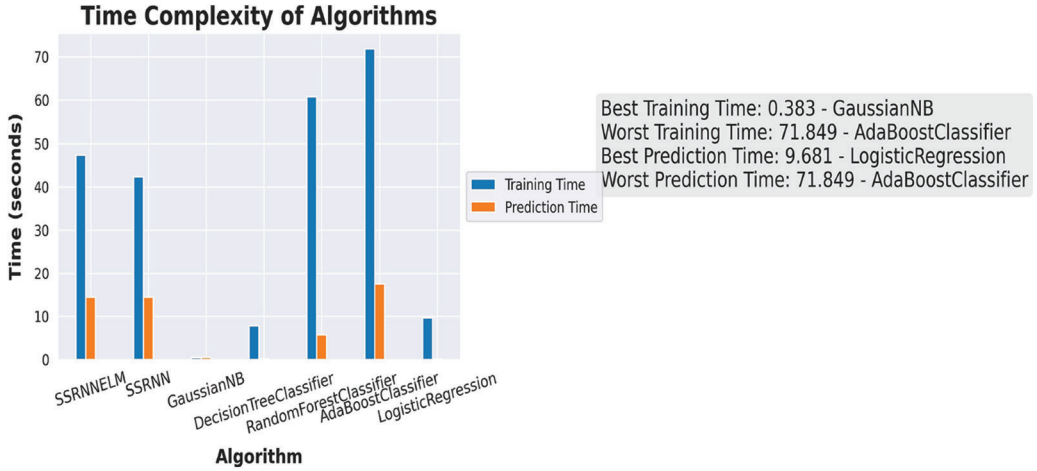

Figure 8: Graph for time complexity using various algorithms

4 Conclusion

This article proposed the hybrid SSRNN-ELM model for an attack and detecting intrusion prevention within the IoT network. The solution has been developed based on a deep neural network that is supervised using RNN and classification as ELM with optimized sample selected features of NSL-KDD dataset called SSRNN-ELM. The LR-RFE and RFE are the feature selection and reduction techniques used in the sample selected features. Then, the model trained has been built on the nodes in the fog layer

which is used to detect the attack in the detection module. The proposed technique has been tested in the IoT-fog testbed and the attack can be identified more quickly than deployed on the cloud. The experimental results show the proposed model is outperforms greater accuracy up to 99% in detecting the attacks in real-time when compared to all other existing algorithms such as LR, NB, DT, Adaboost, and RF.

Acknowledgement: The author would like to thank SRM Institute of Science and Technology for providing facilities to carry out the research work.

Funding Statement: The authors received no specific funding for this study.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] P. Sethi and S. Sarangi, "Internet of things: Architectures, protocols, and applications," *Journal of Electrical and Computer Engineering*, vol. 2017, no. 1, pp. 1–25, 2017.

[2] M. Ahmad, T. Younis, M. Habib, R. Ashraf and S. Ahmed, "A review of current security issues in internet of things," in *Recent Trends and Advances in Wireless and IoT-enabled Networks*, Switzerland, Springer, Cham, pp. 11–23, 2019.

[3] H. Landaluce, L. Arjona, A. Perallos, F. Falcone, I. Angulo *et al.,* "A review of IoT sensing applications and challenges using RFID and wireless sensor networks," *Sensors*, vol. 20, no. 9, pp. 2495, 2020.

[4] L. Tawalbeh, F. Muheidat, M. Tawalbeh and M. Quwaider, "IoT privacy and security: Challenges and solutions," *Applied Sciences*, vol. 10, no. 12, pp. 4102, 2020.

[5] M. Hasan, M. Islam, M. Zarif and M. Hashem, "Attack and anomaly detection in IoT sensors in IoT sites using machine learning approaches," *Internet of Things*, vol. 7, no. 20, pp. 100059, 2019.

[6] Q. Xiao, J. Liu, Q. Wang, Z. Jiang, X. Wang *et al.,* "Towards network anomaly detection using graph embedding," in *International Conference on Computational Science, ICCS 2020, Proceedings: Lecture Notes in Computer Science*, Springer, Cham, pp. 156–169, 2020.

[7] Y. Xin, L. Kong, Z. Liu, Y. Chen, Y. Li *et al.,* "Machine learning and deep learning methods for cybersecurity," *IEEE Access*, vol. 6, pp. 35365–35381, 2018.

[8] M. Kang and J. Kang, "Intrusion detection system using deep neural network for in-vehicle network security," *PLOS ONE*, vol. 11, no. 6, pp. e0155781, 2016.

[9] Y. Chen, Y. Zhang, S. Maharjan, M. Alam and T. Wu, "Deep learning for secure mobile edge computing in cyber-physical transportation systems," *IEEE Network*, vol. 33, no. 4, pp. 36–41, 2019.

[10] R. Vinayakumar, K. Soman and P. Poornachandran, "Deep android malware detection and classification," in *Int. Conf. on Advances in Computing, Communications and Informatics (ICACCI)*, Udupi, India, pp. 1677–1683, 2017.

[11] A. Alrawais, A. Alhothaily, C. Hu and X. Cheng, "Fog computing for the internet of things: Security and privacy issues," *IEEE Internet Computing*, vol. 21, no. 2, pp. 34–42, 2017.

[12] M. Al-Garadi, A. Mohamed, A. Al-Ali, X. Du, I. Ali *et al.,* "A survey of machine and deep learning methods for internet of things (IoT) security," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 3, pp. 1646–1685, 2020.

[13] B. Al-Duwairi, W. Al-Kahla, M. AlRefai, Y. Abedalqader, A. Rawash *et al.,* "SIEM-based detection and mitigation of IoT-botnet DDoS attacks," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 10, no. 2, pp. 2182, 2020.

[14] T. Khempetch and P. Wuttidittachotti, "DDoS attack detection using deep learning," *IAES International Journal of Artificial Intelligence (IJ-AI)*, vol. 10, no. 2, pp. 382, 2021.

[15] A. Maslan, K. Bin Mohamad and F. Binti Mohd Foozy, "Feature selection for DDoS detection using classification machine learning techniques," *IAES International Journal of Artificial Intelligence (IJ-AI)*, vol. 9, no. 1, pp. 137, 2020.

[16] A. Muhammad, C. Foozy and K. Mohammed, "Multischeme feedforward artificial neural network architecture for DDoS attack detection," *Bulletin of Electrical Engineering and Informatics*, vol. 10, no. 1, pp. 458–465, 2021.

[17] B. Mladenov and G. Iliev, "Optimal software-defined network topology for distributed denial of service attack mitigation," *Bulletin of Electrical Engineering and Informatics*, vol. 9, no. 6, pp. 2588–2594, 2020.

[18] M. Albahar, "Recurrent neural network model based on a new regularization technique for real-time intrusion detection in SDN environments," *Security and Communication Networks*, vol. 2019, pp. 1–9, 2019.

[19] X. An, X. Zhou, X. Lü, F. Lin and L. Yang, "Sample selected extreme learning machine based intrusion detection in fog computing and MEC," *Wireless Communications and Mobile Computing*, vol. 2018, pp. 1–10, 2018.

[20] W. Zhou, Y. Jia, A. Peng, Y. Zhang and P. Liu, "The effect of iot new features on security and privacy: New threats, existing solutions, and challenges yet to be solved," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 1606–1616, 2019.

[21] M. Najafabadi, F. Villanustre, T. Khoshgoftaar, N. Seliya, R. Wald *et al.,* "Deep learning applications and challenges in big data analytics," *Journal of Big Data*, vol. 2, no. 1, pp. 1–21, 2015.

[22] P. Torres, C. Catania, S. Garcia and C. Garino, "An analysis of recurrent neural networks for botnet detection behavior," in *2016 IEEE Biennial Congress of Argentina (ARGENCON)*, Buenos Aires, Argentina, pp. 1–6, 2016.

[23] A. Diro and N. Chilamkurti, "Distributed attack detection scheme using deep learning approach for Internet of Things," *Future Generation Computer Systems*, vol. 82, no. 6, pp. 761–768, 2018.

[24] R. Kozik, M. Choraś, M. Ficco and F. Palmieri, "A scalable distributed machine learning approach for attack detection in edge computing environments," *Journal of Parallel and Distributed Computing*, vol. 119, no. 2, pp. 18–26, 2018.

[25] J. Yan, Y. Qi and Q. Rao, "Detecting malware with an ensemble method based on deep neural network," *Security and Communication Networks*, vol. 2018, no. 1, pp. 1–16, 2018.

[26] M. Tavallaee, E. Bagheri, W. Lu and A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in *2009 IEEE Symp. on Computational Intelligence for Security and Defense Applications (CISDA)*, Ottawa, ON, Canada, pp. 1–6, 2009.