Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2021) 29: 1852 – 1870

doi:10.3906/elk-1908-87

# Distributed denial of service attack detection in cloud computing using hybrid extreme learning machine

**Gopal Singh KUSHWAH***, **Virender RANGA**

Department of Computer Engineering, National Institute Of Technology Kurukshetra, Haryana, India

**Received:** 16.08.2019 • **Accepted/Published Online:** 22.03.2021 • **Final Version:** 26.07.2021

**Abstract:** One of the major security challenges in cloud computing is distributed denial of service (DDoS) attacks. In these attacks, multiple nodes are used to attack the cloud by sending huge traffic. This results in the unavailability of cloud services to legitimate users. In this research paper, a hybrid machine learning-based technique has been proposed to detect these attacks. The proposed technique is implemented by combining the extreme learning machine (ELM) model and the blackhole optimization algorithm. Various experiments have been performed with the help of four benchmark datasets namely, NSL KDD, ISCX IDS 2012, CICIDS2017, and CICDDoS2019, to evaluate the performance of our proposed technique. It achieves an accuracy of 99.23%, 92.19%, 99.50%, 99.80% with NSL KDD, ISCX IDS 2012, CICIDS2017, and CICDDoS2019, respectively. The performance comparison with other techniques based on ELM, artificial neural network (ANN) trained with blackhole optimization, backpropagation ANN, and other state-of-the-art techniques is also performed.

**Key words:** Cloud computing, black hole optimization, DDoS attacks, artificial neural networks, extreme learning machine

## 1. Introduction

Cloud computing has become a globally adopted technology [1]. It offers various services, cost-effectively. One of the features of this technology is the availability, meaning that the services must be accessible always to all users. DDoS attacks make cloud services inaccessible to their users [2]. In this attack, multiple machines are used to launch the attack. In the beginning, the attacker searches for vulnerable hosts on the Internet and installs a malicious program on those hosts. These hosts, which are called bots or zombies, also search for other vulnerable hosts and infect those. In this way, a botnet of infected hosts is formed. All the hosts in the botnet are controlled by either the attacker or malicious program installed on hosts at higher levels. To launch the attack, all bots send a large number of requests to the victim server. This makes the server busy serving these fake requests and all the server resources are exhausted. Thus, the server becomes unavailable to legitimate users.

ELM [3] has become a popular machine learning model nowadays. It has many applications in several fields as discussed in [4]. It is an artificial neural network with only one hidden layer. In this model, input to hidden layer weights and hidden layer biases are initialized with random values. For the determination of hidden to output layer weights, the generalized inverse of the matrix is used [13]. In the present work, a new technique by combining extreme learning machine (ELM) with blackhole optimization algorithm [5] has been proposed.

*Correspondence: kushwah.gopalsingh@gmail.com

In this technique, the weights between the input layer and hidden layer along with biases of the hidden layer are optimized using black hole optimization. The weights between the hidden layer and the output layer are calculated using generalized matrix inverse. The blackhole optimization algorithm is based on the black hole phenomenon of the universe. It has been used to solve many problems as described in [6–8]. The reason behind selecting this optimization algorithm over others is that it does not use any parameters. Thus, there is no need for parameter optimization in the proposed technique. The major contributions of this research paper can be summarized below.

- A new hybrid machine learning technique based on ELM and blackhole optimization has been proposed for DDoS attack detection in cloud computing.
- Several experiments using four benchmark datasets have been performed to evaluate the performance of the proposed technique.

The rest of the paper is organized as follows. Some of the recent works proposed in this area are described in the next section. In Section 3, a brief description of ELM and blackhole optimization is presented. The proposed hybrid machine learning-based technique and implemented DDoS attack detection system are discussed in Section 4. In Section 5, experimental results are discussed and the paper is concluded in Section 6.

## 2. Related work

The DDoS attack detection techniques used in cloud computing can be broadly classified into three groups: signature-based, anomaly-based, and hybrid. Some recent works related to each technique have been discussed in this section, a more detailed discussion on proposed works in this area can be found in [9, 10].

### 2.1. Signature based

These techniques use a signature database of previously known attacks. Signatures are the set of rules which define certain characteristics of attack. The network traffic is compared with these signatures. A match with any signature can detect an attack.

In [18], the authors proposed an architecture based on the Intrusion detection message exchange format (IDMEF) and Snort. In [19], the authors proposed a cooperative intrusion detection framework in their proposed research work. An IDS is installed in every region of the cloud, which sends alerts to each other. Based on alerts received from other IDS, new types of attacks are detected. In [20], the authors proposed an approach that uses IDS installed on the virtual switch. It observes the traffic in both directions, inward and outward. In [21], the author proposed a system based on Snort. For detecting known attacks, Snort is used. For detecting attacks derived from known attacks, it uses the signature apriori algorithm.

### 2.2. Anomaly based

These techniques first create a normal profile of the network, and then the network traffic is continuously observed. If the current network profile differs from the normal profile above a predefined threshold, then the attack is detected. These techniques can be further classified as machine learning-based, statistical-based, and software-defined networking (SDN) based.

#### 2.2.1. Machine learning based

In [22], the authors proposed an approach based on probabilistic self-organizing maps. For feature selection and noise removal, principal component analysis and fisher discriminant ratio have been used. In [23], the authors proposed a bi-objective hyper-heuristic framework for support vector machine (SVM) configuration optimization. In this configuration, model complexity and accuracy are considered as two different and conflicting objectives. This SVM configuration is then used for intrusion detection. In [24], the authors proposed a semisupervised approach for DDoS detection. It uses a combination of two algorithms. First is a newly proposed unsupervised algorithm that is based on information gain ratio, entropy estimation, and co-clustering. The second is a supervised algorithm that is an ensemble of extra tree algorithm. In [25], the authors proposed a multilevel method for detection. At the first level, rules are used to identify intrusions from normal traffic. At the second level, the exact categories of attacks are determined by a combination of an SVM and a signal processing technique (DWT). At last, a visual analytic tool is integrated with the system for interactive visual analysis.

In [26], the authors proposed a fuzzy-based semi-supervised learning algorithm that improves the classification performance of the IDS. A single hidden layer feedforward neural network (SLFN) is used as the classifier. A deep learning-based detection system has been proposed in [27]. A recurrent neural network has been used in this system. In [28], the authors proposed a hybrid intrusion detection framework that uses both network-level and host-level activities for detection. It uses a deep neural network for IDS implementation. In [29], the authors proposed a method by combining deep belief network with modified density peak clustering (MDPCA) algorithm. MDPCA is used to reduce the size of the training dataset and the imbalance of samples. A deep belief network is used for classification. In [30], the authors proposed a deep learning-based NIDS, which uses the self-taught learning (STL) technique. In [31], the author used two models for detecting intrusions. One is based on the gravitational search algorithm and artificial neural network (ANN). The other is based on the gravitational search algorithm, particle swarm optimization, and ANN. A DDoS attack detection system based on ANN has been proposed in [7]. The blackhole optimization algorithm is used to train ANN. In [32], the authors proposed a detection system using an ELM classifier for binary as well as multiclass classification. A DDoS attack detection system based on an ensemble of ELMs has been proposed in [51]. In this system, majority voting is used to combine the results of different classifiers.

#### 2.2.2. Statistical based

In [33], a solution based on correlation characteristics of IP and TCP header fields has been proposed. It used a new measure called confidence and based on this, a confidence-based filtering score is calculated. This score is used for attack detection. A modified solution for enhancing the processing speed of the confidence-based filtering method is proposed in [34]. In [35], a detection system based on the covariance matrix model is proposed. A covariance matrix is implemented during the normal period using various parameters. The attack is detected by comparing this covariance matrix with the new covariance matrix.

In [36], the authors proposed a method based on graph-clustering. The criterion for detection is based on the internal and external weights of clusters. In [37], the authors proposed a solution for HTTP/2 based slow rate attacks. For the detection of attacks, the chi-square test has been used, which is used to differentiate between normal packets and attack packets. A dynamic threshold-based detection algorithm is proposed in [38]. Different features of traffic are extracted and, based on these features, four attributes have been calculated. If

attribute values exceed the threshold, then the attack is detected. Here, the threshold value is dynamic instead of constant.

#### 2.2.3. SDN based

In [39], the proposed system uses two components: an attack detection module, and an attack mitigation module. This proposed system also addresses the problem of data shifts. In [40], the authors proposed a system that distributes the traffic load among various traffic processors for scalability. In [41], a detection and mitigation architecture is proposed. It is based on SDN and uses OpenFlow and sFlow tools. It also uses flow rate and flow duration characteristics for evaluation purposes. In [42], a defensive solution for Slowloris and Slow HTTP POST attack has been proposed, which is based on SDN. It uses an application at the SDN controller. When a web server suspects malicious activity, it requests the application for processing, which blocks the sources of malicious activities. In [43], the authors proposed a system that can detect attacks early. It used two levels for detection in the SDN controller. At the first level, enhanced entropy is used, and at the second level Snort is used.

### 2.3. Hybrid techniques

An architecture using both the signature-based and anomaly-based techniques has been proposed in [44]. This is a peer-to-peer architecture. In [45], the authors used the snort and Bayesian classifier both to propose a detection system. In [46], the authors proposed a detection system that uses snort for detecting known attacks and a decision tree for unknown attacks. In [47], the authors proposed a system that is based on snort and three classifiers decision tree, Bayesian, and associative. A score is generated for confirmation of attacks detected by classifiers. In [48], for detecting known attacks C4.5 decision tree algorithm is used. After this, for detecting unknown attacks, several one-class support vector machines are used.

## 3. Background

### 3.1. Extreme learning machine

ELM is a network of processing elements called neurons. It has three layers namely, an input layer, a hidden layer and, an output layer. Each layer is a collection of neurons. The neurons in different layers are interconnected through links having weights. The input to the network is applied through the input layer and the output layer is used to take the output from the network. The hidden layer performs the actual computation of the network. Each neuron uses an activation function for processing. Commonly used activation functions are linear activation function and sigmoid activation function.

ELM does not require iterative training, it is trained in a single step. In this model, weights between the input layer and the hidden layer along with biases of the hidden layer are randomly initialized. The corresponding hidden-output layer weights are analytically determined using Moore-Penrose generalized inverse of a matrix [13]. Due to this, it can be trained quickly with better generalization than other models. Consider, $n$ represents number of input layer neurons, $l$ represents number of hidden layer neurons, and $m$ represents number of output layer neurons. Let U represents weight matrix of input-hidden layers, $b_j$ represents $j^{th}$ hidden neuron bias and, V represents weight matrix of hidden-output layers. The activation function in the hidden layer is $f$. The ELM with the above configuration is represented in Figure 1.

Suppose, $(\mathbf{x}_i, \mathbf{t}_i)$ represents $i^{th}$ training sample. Here, $\mathbf{x}_i = [x_{i1}, x_{i2}, x_{i3}, ..., x_{in}]^T \in R^n$ is the vector of $n$

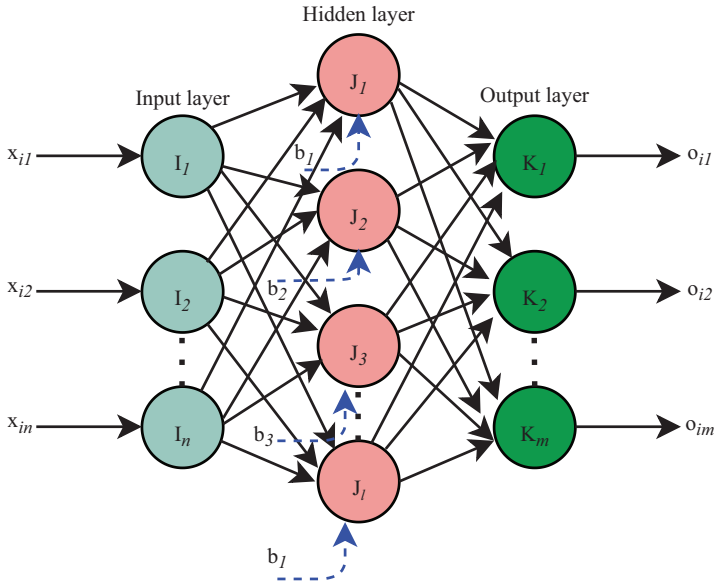



**Figure 1**. Extreme learning machine.

features for $i^{th}$ sample and $\mathbf{t}_i = [t_{i1}, t_{i2}, t_{i3}, ..., t_{im}]^T \in R^m$ is the target vector representing $m$ classes, to which the sample $i$ may belong. If there are total N training samples then output of this network can be modeled as

$$\mathbf{o}_i = \sum_{j=1}^{l} f(\mathbf{u}_j, b_j, \mathbf{x}_i)\mathbf{v}_j, for \;\; i = 1, 2, 3, ..., N. \tag{1}$$

Where, $\mathbf{u}_j \in R^n$ and $b_j \in R$ are the learning parameters of the $j^{th}$ hidden neuron and $\mathbf{v}_j = [v_{j1}, v_{j2}, , v_{jm}]^T$ is the vector of weights which connects $j^{th}$ hidden neuron to $m$ output neurons. The output of $i^{th}$ sample is of the form $\mathbf{o}_i = [o_{i1}, o_{i2}, , o_{im}]^T \in R^m$. We can write the above equation for complete training dataset of N samples as

$$\mathbf{O} = \mathbf{H}V \tag{2}$$

Where,

$$\mathbf{H} = \begin{bmatrix} f(\mathbf{u}_1, b_1, \mathbf{x}_1) & \dots & f(\mathbf{u}_l, b_l, \mathbf{x}_1) \\ \vdots & \vdots & \vdots \\ f(\mathbf{u}_1, b_1, \mathbf{x}_N) & \dots & f(\mathbf{u}_l, b_l, \mathbf{x}_N) \end{bmatrix}, V = \begin{bmatrix} \mathbf{v}_1^T \\ \vdots \\ \mathbf{v}_l^T \end{bmatrix} \;\; and \;\; \mathbf{O} = \begin{bmatrix} \mathbf{o}_1^T \\ \vdots \\ \mathbf{o}_N^T \end{bmatrix}$$

For minimizing the error $||\mathbf{O\text{-}T}||$, where $\mathbf{T} = (\mathbf{t}_1, \mathbf{t}_2, , \mathbf{t}_N)$, we can randomly initialize the parameters $u_j$ and $b_j$ independent of the input values. Now, the hidden-output weights can be determined by finding a least square solution of the following equation.

$$V = \mathbf{H}^{\dagger}\mathbf{T} \tag{3}$$

Where † represents the Moore–Penrose generalized inverse of a matrix. Now, the model is trained as the value of hidden-output weight matrix V is known. Here important to mention that there has been some debate

about the originality of ELM [14, 15]. The creator of ELM has given responses to the comments received from the research community in [16, 17].

### 3.2. Black hole optimization

It is an optimization algorithm inspired by the blackhole phenomenon of the universe. During initialization, a fixed number of solutions, called the initial population, are randomly generated. Then objective function against each solution is calculated. The solution with a minimum (or maximum) value of the objective function is now designated as black hole position and all remaining solutions are designated as the position of stars. The position of each star is changed due to its attraction toward the blackhole. Equation (4) is used to calculate the new position of stars. Here, $Pos_i(new)$ represents a new position and $Pos_i(old)$ represents an old position of $i^{th}$ star. The black hole position is represented by $Pos_{BH}$ and $r$ represents a random number in [0,1].

$$Pos_i(new) = Pos_i(old) + r * (Pos_{BH} - Pos_i(old)) \tag{4}$$

Now, the objective function for new solutions (position of stars) is calculated. If the objective function value of any star at a new position becomes better than blackhole, then positions of that star and blackhole are exchanged. Therefore, the position of blackhole at this point is the optimal solution up to the current iteration. For generating the population of the next generation, the event horizon radius is calculated using equation (5), where $\xi$ represents the number of solutions in the population, and $f_{obj}$ represents objective function. After that, the distance of stars from the black hole is calculated. A star is destroyed if it crosses the event horizon radius and a new star with a random position is created. The group of all stars and blackhole works as the population for the next generation. Equation (6) represents the distance between a star and blackhole.

$$Radius = \frac{f_{obj}(Pos_{BH})}{\sum_{i=1}^{\xi} f_{obj}(Pos_i)} \tag{5}$$

$$Distance_i = f_{obj}(Pos_{BH}) - f_{obj}(Pos_i) \tag{6}$$

The process of black hole optimization is shown in Figure 2. The number of solutions in initial population is 4, which are represented as positions of stars $x_1, x_2, x_3$ and $x_4$. The position of the black hole is $x_4$ with the minimum value of the objective function. The position of stars are changed due to attraction and becomes $x_1', x_2'$, and $x_3'$. Now, the objective function for other stars is recalculated. New blackhole position is $x_3'$. The event horizon radius and distance of each star from the black hole are calculated. The star with position $x_4$ falls under the radius, it is destroyed and a new star at random position $x_4'$ is generated. Now all the stars $x_1', x_2', x_3'$ and $x_4'$ become population for the next generation. This process continues until the termination criteria are met.

## 4. DDoS attack detection

### 4.1. Proposed hybrid extreme learning machine

As the training of ELM uses random weights between input and hidden layers, it may be possible that these random weights are not optimal, instead, there are other weights in the search space, which provide better training accuracy. To overcome this problem, we have used black hole optimization algorithm so that the optimal values of input to hidden layer weights and hidden biases can be selected. The proposed technique is called blackhole ELM (BH-ELM).

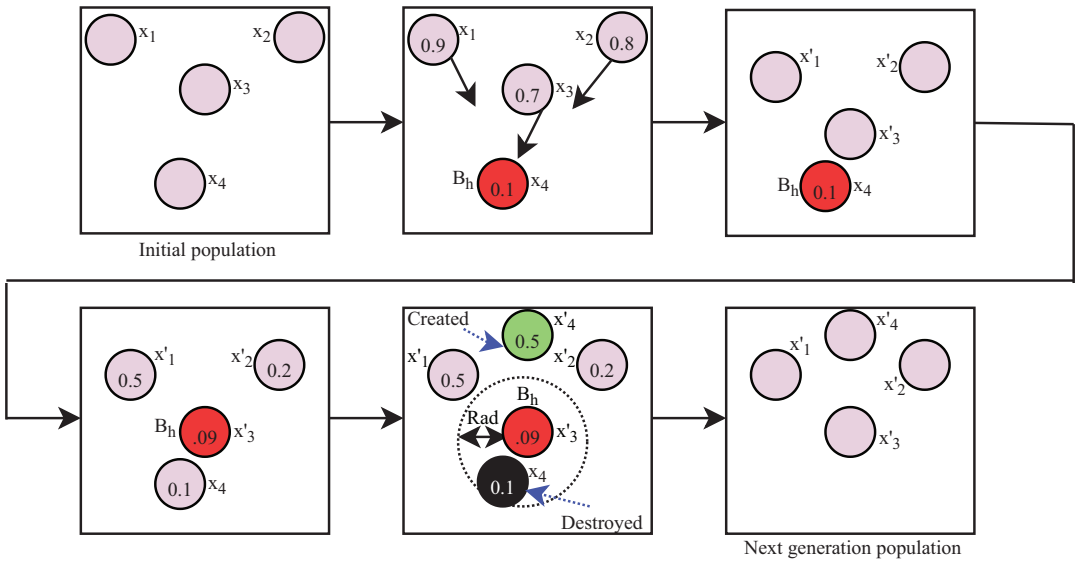



**Figure 2**. Black hole optimization.

Suppose, $\mathbf{UB} = [u_{11}, u_{12}, , u_{1l}, u_{21}, u_{22}, , u_{2l}, , u_{n1}, u_{n2}, ..., u_{nl}, b_1, b_2, , b_l]$ represents the matrix of input-hidden weights and hidden biases in vector form. This vector represents a solution in our technique. Now, randomly initialize a population of **UB** vectors of size $IP_{size}$ (initial population size). Let, N be the total number of training samples. For each vector **UB** in the initial population, the value of corresponding hidden-output weight matrix V is calculated with the help of generalized matrix inverse using equation (3). After this, the output of all training samples is calculated using equation (1). Now, the objective function is calculated. Here, mean squared error (MSE) is used as an objective function that is to be minimized; it is given by equation (7).

$$f_{obj} = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{t}_i - \mathbf{o}_i)^2 \tag{7}$$

The vector **UB** with minimum objective function value represents the position of the black hole, and all other vectors represent the position of normal stars. Due to attraction toward the blackhole, the positions of other stars are changed according to Eq (4). After the change of position, if some star achieves an objective function value less than the blackhole, then the position of that star and blackhole is exchanged. The position of the blackhole at this point represents the best solution up to the current iteration. This means these are the best values of input-hidden weights and hidden biases. For generating the population for next generation, the event horizon radius is calculated using Eq (5). Then, the distance of each star from the blackhole is calculated using Eq (6). If any star crosses this radius, then it is destroyed and a new star with a random position is generated. This set of positions of stars along with blackhole position is passed to the next generation. This process is continued, till the desired value of the objective function is achieved or maximum iterations have reached. When this process is complete, the best solution available till the last iteration, i.e. position of blackhole, along with their corresponding matrix V are returned. The training process of the proposed technique is presented in algorithm 1.

**Algorithm 1** : Blackhole ELM (BH-ELM) training.

**Input:** Initial population size ($IP_{size}$), number of training samples ($N$), number of hidden neurons ($l$), number of iterations ($k$)
**Output:** Trained classifier
1. Randomly generate $IP_{size}$ number of **UB** vectors
2. For *iter*=1 to $k$ do
3. For each individual in population do
4. calculate V=$\mathbf{H}^{\dagger}\mathbf{T}$
5. calculate $\mathbf{o}_i = \sum_{j=1}^{l} f(\mathbf{u}_j, b_j, \mathbf{x}_i)\mathbf{v}_j$, $i$=1,2,...,N
6. calculate $f_{obj} = (\frac{1}{N}\sum_{i=1}^{N}(\mathbf{t}_i - \mathbf{o}_i)^2)$
7. End for
8. The individual with minimum $f_{obj}$ is designated as position of blackhole, represented as $Pos_{BH}$
9. Represent its objective function value as $f_{obj}(Pos_{BH})$
10. All other individuals represent position of normal stars
11. Update positions of each star using $Pos_i(new) = Pos_i(old) + r * (Pos_{BH} - Pos_i(old))$
12. Recalculate $f_{obj}$ of stars (Repeat steps 3 to 7)
13. If a star with less $f_{obj}$ than black hole found, interchange their positions
14. Calculate event horizon radius, Radius $= \frac{f_{obj}(Pos_{BH})}{\sum_{i=1}^{IP_{size}} f_{obj}(Pos_i)}$
15. Calculate distance of each star from black hole using $D_i = f_{obj}(Pos_{BH}) - f_{obj}(Pos_i)$
16. If a star crosses event horizon radius then it is destroyed and a star with random position is generated
17. Now, all stars along with black hole work as population for next iteration
18. End For
19. The position of blackhole in last iteration is the optimal value of solution vector **UB**
20. Return blackhole position (optimal **UB**) and its corresponding matrix V

### 4.2. An example of the proposed technique

Here, we are discussing an example for the clarity of readers. Suppose, there are 3 features in the samples. The number of neurons in the hidden layer and output layer is 2. The initial population size is four. Now the size of vector **UB** will be (3+1)*2 = 8 (3 input features and 1 bias input), and size of matrix V will be 2*2 = 4. Table 1 represents the different operations of the technique. Part (a) represents the four **UB** vectors (represented as rows), and their corresponding V matrix in vector form are represented in part (b) (represented as rows). Part (c) represents the values of objective functions for each solution in the population. The individual with minimum objective function value, i.e. the third vector in part (a) is designated as black hole.

When the blackhole attracts the other stars, then their position changes according to Eq (4). The new position (updated population) of individuals is presented in part (d). Part (e) represents their corresponding V vectors, and part (f) represents objective function value. Now from part (f), it is observed that after a position change, a new individual (fourth) has an objective function value lesser than the black hole, i.e. $0.2649 < 0.2825$. Hence, this individual becomes the black hole, which represents the optimal values of weights till the current iteration for an ELM. Now the event horizon radius is calculated using Eq. (5), which is 0.2334. After that, the distance of each individual from blackhole is calculated using Eq. (6), which is represented in part (g).

As observed from part (g), all the individuals have a distance less than the event horizon radius; hence, all these are swallowed by black hole, and new random individuals are generated. Part (h) represents new individuals with blackhole. These individuals work as the initial population for the next iteration. All these steps are repeated until the desired accuracy is achieved.

**Table 1**. An example of the proposed technique.

(a). Initial population matrix.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0.24017 | 0.0339 | 0.3896 | -0.2799 | 0.4685 | 0.5684 | 0.1818 | 0.6683 |
| 0.3907 | 0.1133 | –0.147 | -0.0915 | –0.1394 | 0.4111 | –0.0812 | –0.9687 |
| 0.4403 | –0.687 | 0.6725 | –0.2272 | 0.3875 | –0.7813 | –0.8993 | 0.7274 |
| –0.3062 | 0.1241 | 0.4627 | 0.5511 | 0.8904 | –0.2201 | –0.5426 | –0.8438 |

(b). Matrix V

| | | | |
|---|---|---|---|
| 1.2162 | –0.4071 | –0.3653 | 1.0886 |
| –2.9202 | 4.7438 | 5.8859 | –5.3748 |
| 2.6126 | –1.5675 | –0.7282 | 1.8105 |
| 2.1153 | –0.9591 | –1.7492 | 3.2573 |

(c). Objective function value

| |
|---|
| 0.5081 |
| 0.3071 |
| 0.2825 |
| 0.2885 |

(d). Updated population matrix.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0.374 | –0.4483 | 0.5789 | –0.2446 | 0.4143 | –0.3346 | –0.5415 | 0.7078 |
| 0.4155 | –0.2869 | 0.2628 | –0.1594 | 0.1241 | –0.1853 | –0.4904 | –0.1202 |
| 0.4403 | –0.687 | 0.6725 | –0.2272 | 0.3875 | –0.7813 | –0.8993 | 0.7274 |
| 0.1205 | –0.3395 | 0.5826 | 0.1062 | 0.6029 | –0.5409 | –0.7465 | 0.0543 |

(e). Updated matrix V.

| | | | |
|---|---|---|---|
| 2.4709 | –1.8473 | –0.9261 | 2.0897 |
| 4.0221 | –3.0046 | –2.5582 | 3.8879 |
| 2.6126 | –1.5675 | –0.7282 | 1.8105 |
| 2.9006 | –1.962 | –1.4786 | 2.8377 |

(f). Updated objective function value

| |
|---|
| 0.3542 |
| 0.4981 |
| 0.2825 |
| 0.2649 |

(g). Distance from blackhole.

| |
|---|
| 0.0893 |
| 0.2332 |
| 0.0176 |
| 0 |

(h). Next generation population matrix.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| –0.7556 | 0.3423 | 0.1991 | –0.888 | –0.8873 | –0.6949 | –0.9607 | –0.1296 |
| 0.6644 | 0.2347 | 0.0402 | 0.7277 | –0.8046 | 0.8161 | –0.7839 | 0.0339 |
| –0.7136 | 0.1187 | –0.9908 | 0.5333 | 0.6974 | 0.8336 | 0.9739 | 0.0102 |
| 0.1205 | –0.3395 | 0.5826 | 0.1062 | 0.6029 | –0.5409 | –0.7465 | 0.0543 |

### 4.3. DDoS attack detection system

Figure 3 represents the block diagram of the proposed DDoS attack detection system. It has two major components: preprocessor, and classifier. All the incoming service requests are applied to the preprocessor. It constructs samples from these requests. The samples are formed in groups. A group of samples is created for incoming traffic collected during each period $t$. The important features from a request packet are extracted.

These features are used to predict a request as a normal request or attack. Some feature values are symbolic constant and they are replaced with numerical values. Normalization is also performed, which scales feature values in [0,1]. Suppose a feature $a$ has maximum value $a_{max}$ and minimum value $a_{min}$ in all the samples, then normalized value of $i^{th}$ sample is calculated by Eq. (8).

$$a^i_{norm} = \frac{a^i - a_{min}}{a_{max} - a_{min}} \tag{8}$$

Where, $a^i$ is the value of $i^{th}$ sample for feature $a$. The Classifier classifies samples received from preprocessor into attack samples or normal samples. We used our proposed technique BH-ELM as a classifier. In this technique, first, training is performed using training data. The training data contains samples with known type. Let the training data has samples of the form $(\mathbf{x}_i, \mathbf{t}_i)$. Where, $\mathbf{x}_i = [x_{i1}, x_{i2}, , x_{in}]$ represents $n$ features of the samples. And, $\mathbf{t}_i = [1, 0]$ or $\mathbf{t}_i = [0, 1]$ represents type of sample. If $\mathbf{t}_i = [1, 0]$, sample belongs to attack class and if $\mathbf{t}_i = [0, 1]$, then sample belongs to normal class.

After training, it is used to classify incoming requests as either attack or normal. Suppose, it receives samples from preprocessor of the form $\mathbf{x}_i = [x_{i1}, x_{i2}, , x_{in}]$. Then, it generates output for each sample of the form $\mathbf{o}_i = [1, 0]$ or $\mathbf{o}_i = [0, 1]$. If the output $\mathbf{o}_i = [1, 0]$, it means the sample belongs to attack category. If the output $\mathbf{o}_i = [0, 1]$, then sample belongs to normal category.

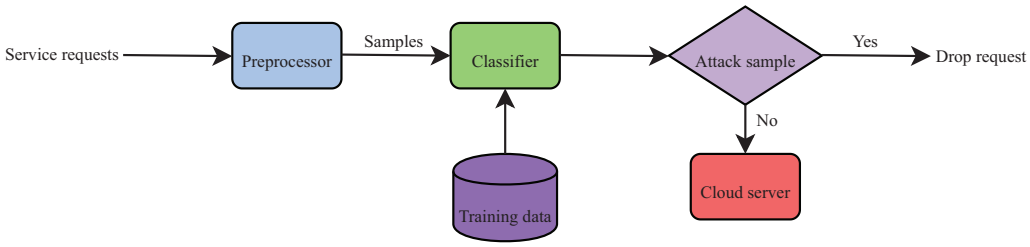



**Figure 3**. Proposed DDoS attack detection system.

## 5. Experimental results and discussions

Several experiments have been performed to evaluate the performance of the proposed technique on a PC with Windows 10 operating system, Intel Core i5 processor, and 16 GB memory. For implementation, MATLAB (MathWorks, Inc., Natick, MA, USA) is used.

### 5.1. Datasets

Four benchmark datasets namely NSL KDD [11], ISCX IDS 2012 [12] , CICIDS2017 [49], and CICDDoS 2019 [50] have been used in experiments. These datasets have 41, 19, 84, and 87 features, respectively. All the features of these datasets are used for experimental purposes. In all the datasets, only DoS/DDoS attack samples are considered. Further details about these datasets are given in Table 2.

**Table 2**. Datasets information.

| Dataset name | Number of features | Number of samples | | | | | |
|---|---|---|---|---|---|---|---|
| | | Training | | | Testing | | |
| | | Normal | Attack | Total | Normal | Attack | Total |
| NSL KDD | 41 | 10000 | 10000 | 20000 | 2500 | 2500 | 5000 |
| ISCX IDS 2012 | 19 | 9600 | 9600 | 19200 | 2400 | 2400 | 4800 |
| CICIDS2017 | 84 | 750 | 750 | 1500 | 500 | 500 | 1000 |
| CICDDoS2019 | 87 | 875 | 875 | 1750 | 375 | 375 | 750 |

### 5.2. Performance metrics

For evaluating the performance of our proposed technique, accuracy, sensitivity, and specificity metrics are used. These are defined and shown by equations (9), (10) and (11), respectively.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \times 100 \tag{9}$$

$$Sensitivity = \frac{TP}{TP + FN} \times 100 \tag{10}$$

$$Specificity = \frac{TN}{FP + TN} \times 100 \tag{11}$$

Where, TP=number of true positives, which is defined as the number of samples that belong to the attack and correctly identified by the classifier. TN=number of true negatives, which is defined as the number of samples that belong to normal and correctly identified by the classifier. FP=number of false positives, which is defined as the number of samples that belong to normal but incorrectly identified as an attack by the classifier. FN=number of false negatives, which is defined as the number of samples that belong to attack but incorrectly identified as normal by classifier.

### 5.3. Performance evaluation

To evaluate the performance of the proposed technique (BH-ELM), we used the sigmoid activation function for the hidden layer. The initial population size (IPsize) is taken as 32. The number of neurons ($l$) is varied from 100 to 1000 in the steps of 100 for NSL KDD, and from 10 to 100 in steps of 10 for the remaining datasets. For each calculated metric, the experiment is repeated 50 times, and the average is taken. The best accuracy achieved by BH-ELM in the above range of neurons for NSL KDD, ISCX IDS 2012, CICIDS2017, and CICDDoS2019 is 99.23%, 92.19%, 99.50%, and 99.80%, respectively. The sensitivity and specificity achieved for these datasets are 99.21%, 86.10%, 100%, 99.80%, and 99.68%, 100%, 99.20%, 99.80%, respectively. Using the same parameters we perform experiments with ELM and ANN trained with blackhole optimization (BH-ANN). We also perform experiments with backpropagation ANN (BP-ANN). Then, the results of BH-ELM are compared with BP-ANN, ELM, and BH-ANN as shown in Table 3. It is observed from the table that BH-ELM shows the highest detection accuracy with all datasets. It also shows the highest or same sensitivity and the highest specificity (except for DDoS 2019 dataset).

**Table 3**. Performance of the proposed technique.

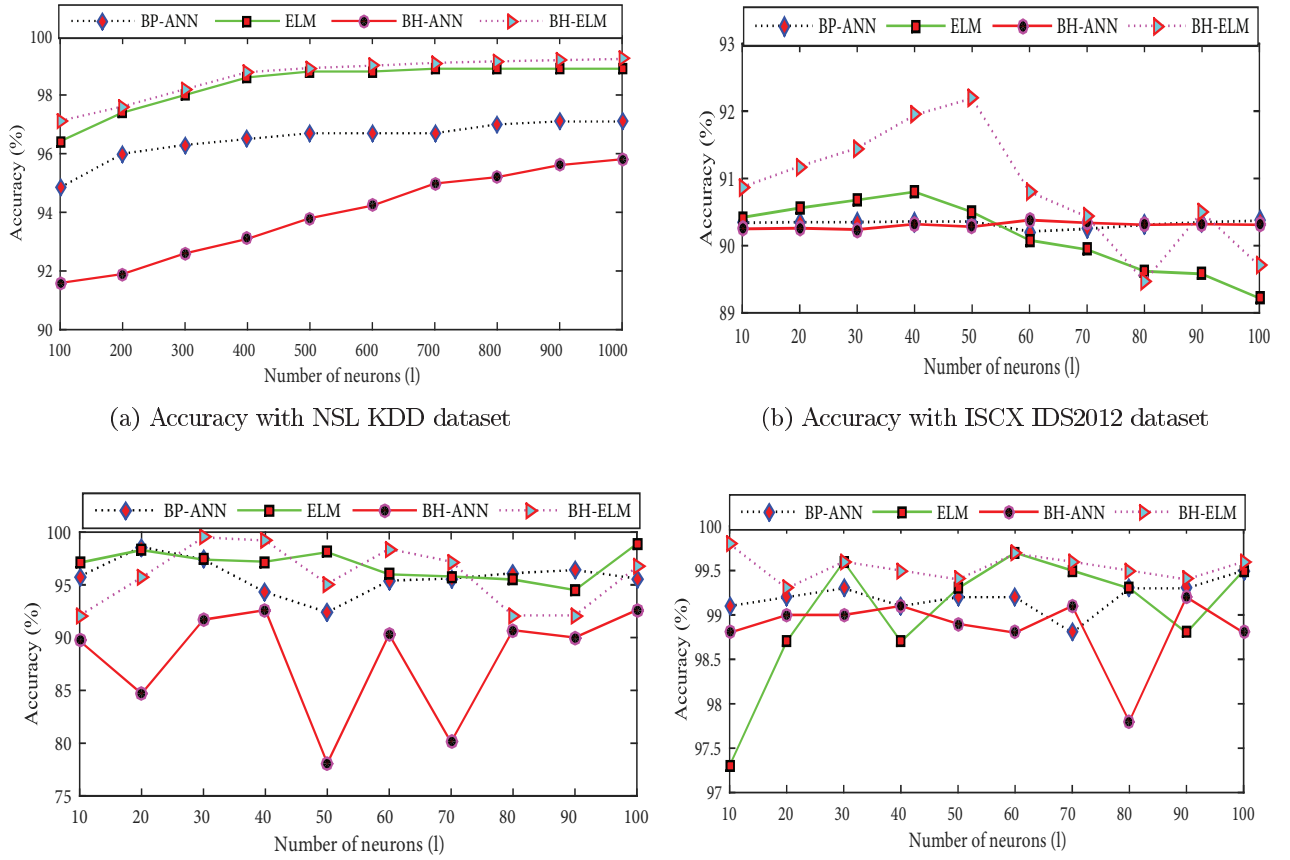
| Dataset | Technique | Accuracy (%) | Sensitivity (%) | Specificity (%) |
|---|---|---|---|---|
| NSL KDD | ELM | 98.90 | 98.90 | 98.45 |
| | BH-ANN | 95.81 | 94.90 | 98.09 |
| | BP-ANN | 97.10 | 96.20 | 97.74 |
| | BH-ELM(proposed) | 99.23 | 99.21 | 99.68 |
| ISCX IDS 2012 | ELM | 90.80 | 82.54 | 99.68 |
| | BH-ANN | 90.38 | 81.21 | 99.50 |
| | BP-ANN | 90.37 | 80.74 | 99.79 |
| | BH-ELM(proposed) | 92.19 | 86.10 | 100 |
| CICIDS 2017 | ELM | 98.90 | 100 | 97.70 |
| | BH-ANN | 92.60 | 100 | 86.70 |
| | BP-ANN | 98.60 | 100 | 98.00 |
| | BH-ELM(proposed) | 99.50 | 100 | 99.20 |
| CICDDoS 2019 | ELM | 99.70 | 99.70 | 100 |
| | BH-ANN | 99.20 | 99.20 | 99.00 |
| | BP-ANN | 99.50 | 99.70 | 99.99 |
| | BH-ELM(proposed) | 99.80 | 99.80 | 99.80 |

### 5.4. Discussion

The performance of BH-ELM is compared with BP-ANN, ELM, and BH-ANN using all metrics for the entire range of neurons. The comparison for accuracy, sensitivity, and specificity is shown by Figures 4(a)–4(d), Figures 5(a)–5(d), and Figures 6(a)–6(d), respectively. The following observations about BH-ELM are made from the figures. Using NSL KDD initially shows a good improvement in accuracy with the number of neurons; however, later only a small improvement is observed. Using ISCX IDS 2012 dataset, accuracy increases with the number of neurons, attains a maximum value, and then starts decreasing, so it shows a zigzag. The accuracy shows a zigzag curve with the number of neurons using CICIDS2017 dataset. For CICDDoS2019 dataset, accuracy starts with a high value, then decreases. After that, it shows zigzag with the number of neurons. Using NSL-KDD, sensitivity increases with the number of neurons in the entire range. With ISCX IDS 2012, sensitivity starts with a high value, then it starts decreasing, and again increases, and then shows zigzag. Using CICIDS 2017, sensitivity remains almost constant in the entire range. Using CICDDoS2019, sensitivity starts with a high value, then decreases. After that, it shows zigzag with the number of neurons. With NSL-KDD, specificity increases with the number of neurons in the entire range. With ISCX IDS 2012, specificity remains almost constant initially, it shows zigzag later. With CICIDS2017, specificity shows a zigzag. For CICDDoS2019, specificity remains almost constant for all values of neurons. The conclusion about the number of neurons ($l$) can be made as follows: A higher value for NSL KDD, a lower value for CICDDoS2019, and a medium value for ISCX IDS 2012, CICIDS2017 is favorable for high accuracy.

The training time comparison of BH-ELM with ELM, BH-ANN, and BP-ANN is shown by Figures 7(a)–8(d). Figure 7(a) shows time comparison for NSL KDD, Fig 7(b) for ISCX IDS 2012, Figure 7(c) for CICIDS 2017, and Figure 7(d) for CICDDoS 2019 datasets, respectively. It is observed from the figures that BH-ELM requires more training time than all other techniques. The reason behind this is due to the hybrid nature of BH-ELM technique, which requires more training time than other proposed techniques having a single model. So, we have achieved higher detection accuracy than other techniques on the cost of training time. BH-ELM is

more suitable and useful where detection accuracy is more important than training time. At last, the accuracy of BH-ELM is compared with other state-of-the-art techniques in Table 4. The comparison is based on the results of NSL KDD dataset. The reason behind this choice is the wide usage of this dataset in the intrusion detection, and DDoS attack detection research works.




(a) Accuracy with NSL KDD dataset

(b) Accuracy with ISCX IDS2012 dataset

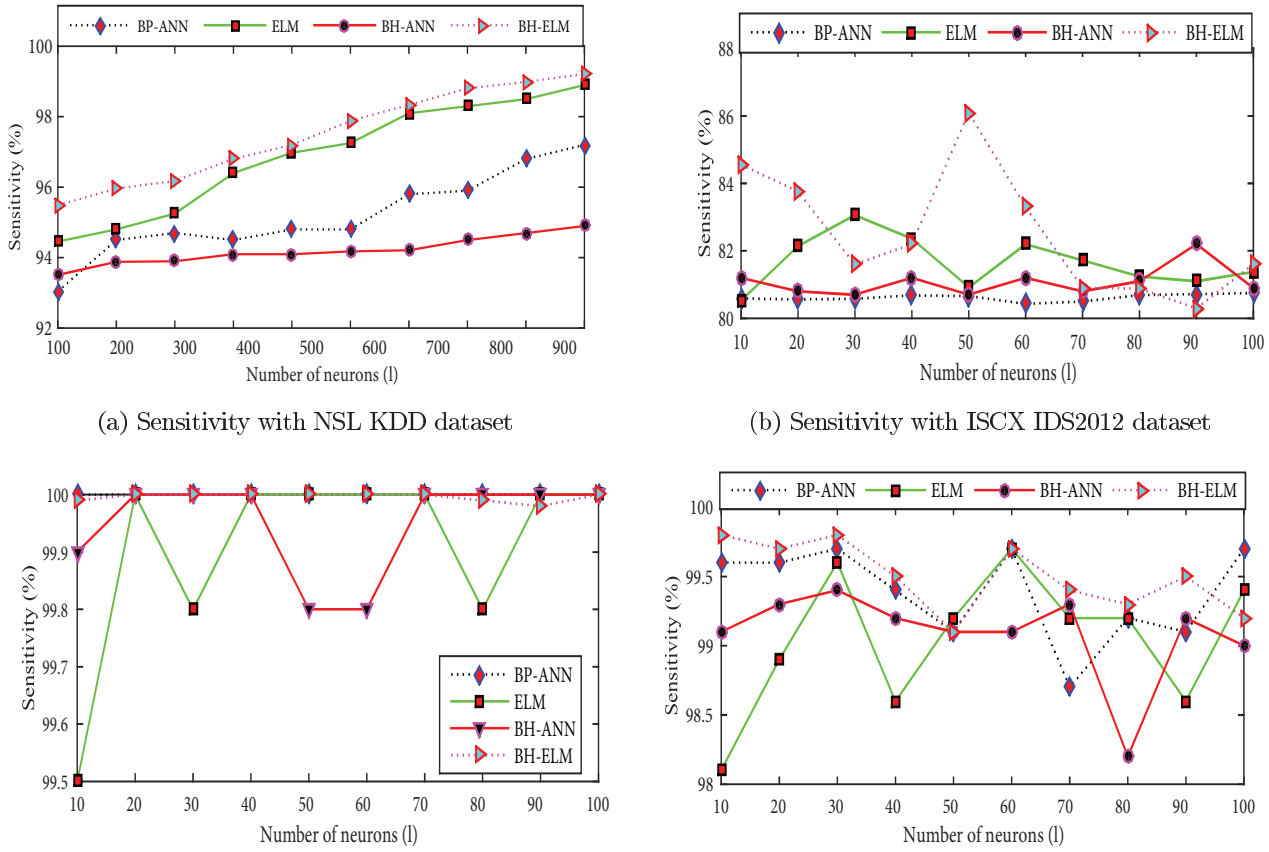
(c) Accuracy with CICIDS2017 dataset

(d) Accuracy with CICDDoS2019 dataset

**Figure 4**. Detection accuracy of the proposed technique with different datasets

## 6. Conclusion and future scope

In this research paper, a solution for DDoS attack detection problem in cloud computing has been proposed. Firstly, a hybrid machine learning technique has been proposed, which combines the ELM model and blackhole optimization algorithm. Then, the proposed technique is used to design a system for detecting DDoS attacks in cloud computing. Various experiments have been performed using four benchmark datasets to evaluate the performance of the proposed technique. The proposed technique detects the attacks with an accuracy of 99.23%, 92.19%, 99.50%, and 99.80% using NSL KDD, ISCX IDS 2012, CICIDS2017, and CICDDoS2019 datasets, respectively. Performance comparison of our proposed technique with other techniques such as ELM, backpropagation ANN, and ANN trained with blackhole optimization shows that the proposed technique detects attacks with better accuracy than these techniques. In the future, we will work on reducing the training time of the proposed technique without affecting its detection accuracy.



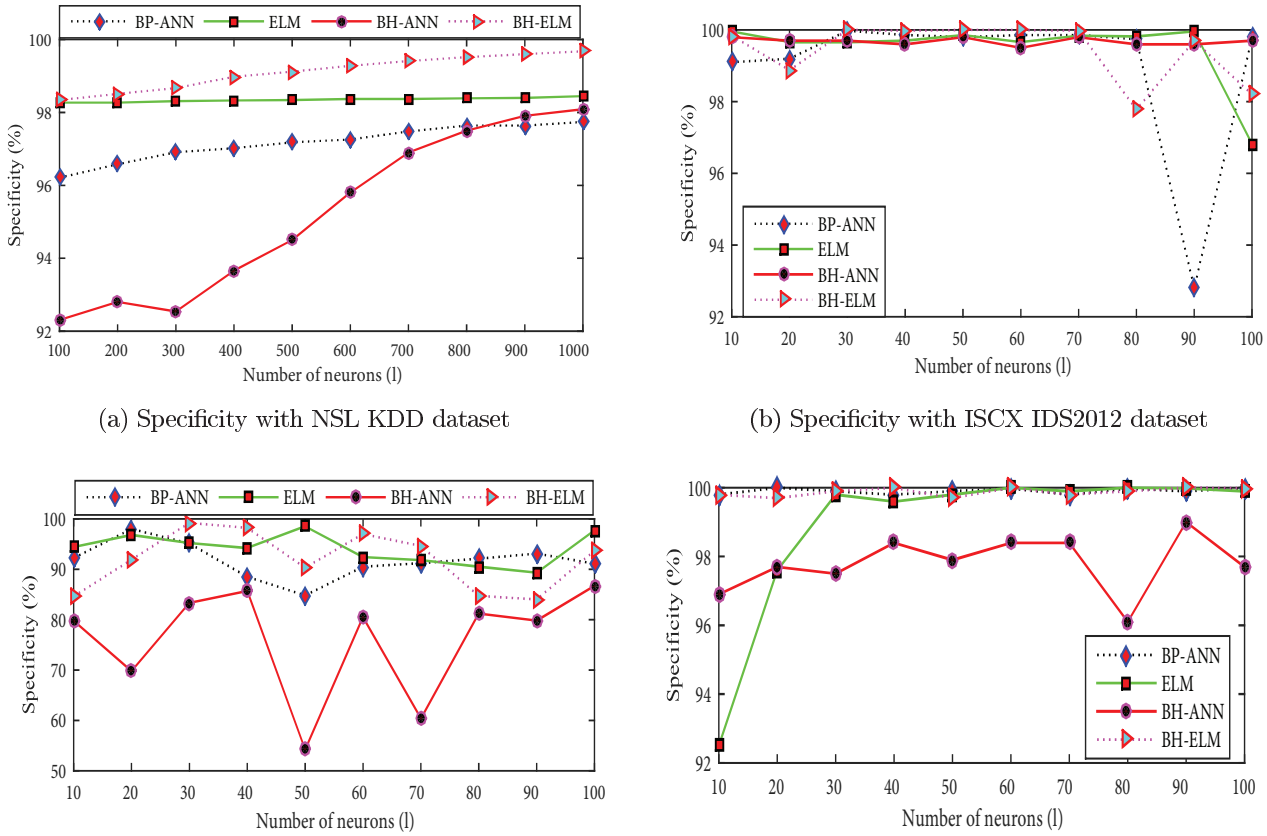

(a) Sensitivity with NSL KDD dataset

(b) Sensitivity with ISCX IDS2012 dataset

(c) Sensitivity with CICIDS2017 dataset

(d) Sensitivity with CICDDoS2019 dataset

**Figure 5**. Sensitivity of the proposed technique with different datasets.



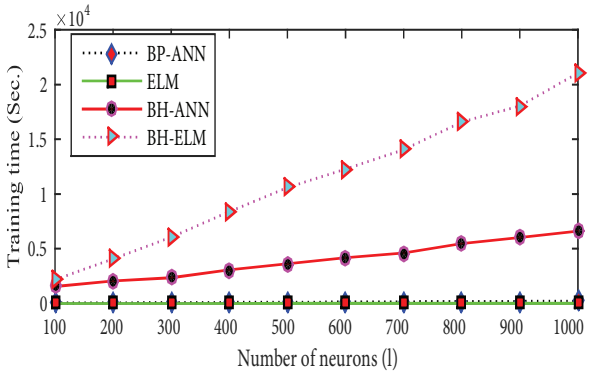

(a) Specificity with NSL KDD dataset

(b) Specificity with ISCX IDS2012 dataset

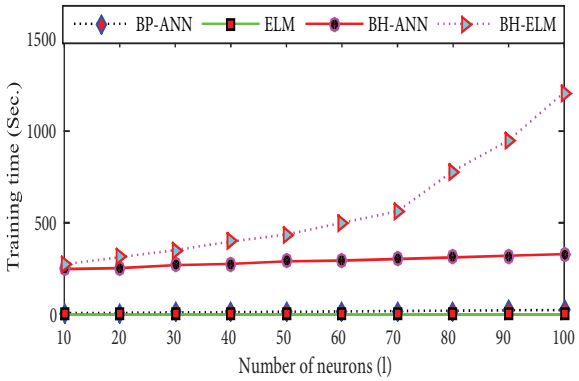
(c) Specificity with CICIDS2017 dataset

(d) Specificity with CICDDoS2019 dataset

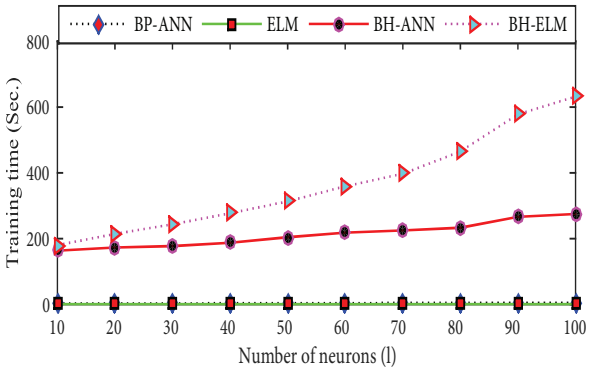
**Figure 6**. Specificity of the proposed technique with different datasets.



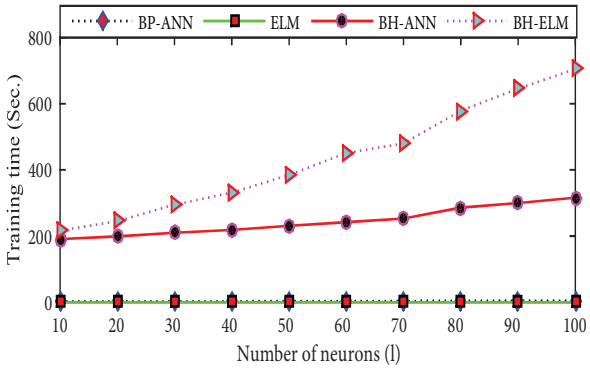

(a) Training time with NSL KDD dataset




(b) Training time with ISCX IDS2012 dataset




(c) Training time with CICIDS2017 dataset




(d) Training time with CICDDoS2019 dataset

**Figure 7**. Training time of the proposed technique with different datasets.

**Table 4**. Detection accuracy comparison with other works (using NSL KDD dataset).

| Work | Detection Accuracy (%) |
|---|---|
| De et al. [22] | 93± 0.01 |
| Sabar et al. [23] | 85.69 |
| Ji et al. [25] | 96.67 |
| Ashfaq et al. [26] | 84.12 |
| Yin et al. [27] | 83.28 |
| Vinayakumar et al. [28] | 80.10 |
| Yang et al. [29] | 82.08 |
| Javaid et al. [30] | 88.39 |
| Dash et al.[31] | 95.26 |
| Kushwah et al. [32] | 98.73 |
| Kushwah et al.[51] | 99.18 |
| Idhammad et al.[24] | 98.23 |
| Present work | 99.23 |

## References


[1] Armbrust M, Fox A, Griffith R, Joseph AD, Katz R et al. A view of cloud computing. Communications of the ACM 2010; 53 (4): 50-8. doi: 10.1145/1721654.1721672

[2] Lau F, Rubin SH, Smith MH, Trajkovic L. Distributed denial of service attacks. In: IEEE 2000 International Conference on Systems, Man and Cybernetics; Nashville, TN, USA; 2000. pp. 2275-2280.

[3] Huang GB, Zhu QY, Siew CK. Extreme learning machine: a new learning scheme of feedforward neural networks. Neural networks 2004; 2: 985-990. doi: 10.1109/IJCNN.2004.1380068

[4] Deng C, Huang G, Xu J, Tang J. Extreme learning machines: new trends and applications. Science China Information Sciences 2015; 58(2): 1-16. doi: 10.1007/s11432-014-5269-3

[5] Hatamlou A. Black hole: A new heuristic optimization approach for data clustering. Information sciences 2013; 222: 175-184. doi: 10.1016/j.ins.2012.08.023

[6] Kumar J, Singh AK. Dynamic resource scaling in cloud using neural network and black hole algorithm. In: IEEE 2016 Fifth International Conference on Eco-friendly Computing and Communication Systems (ICECCS); Bhopal, India; 2016. pp. 63-67.

[7] Kushwah GS, Ali ST. Detecting DDoS attacks in cloud computing using ANN and black hole optimization. In: IEEE 2017 2nd International Conference on Telecommunication and Networks (TEL-NET); Noida, India; 2017. pp. 1-5.

[8] Hatamlou A. Solving travelling salesman problem using black hole algorithm. Soft Computing 2018; 22(24): 8167-8175. doi: 10.1007/s00500-017-2760-y

[9] Agrawal N, Tapaswi S. Defense Mechanisms Against DDoS Attacks in a Cloud Computing Environment: State-of-the-Art and Research Challenges. IEEE Communications Surveys & Tutorials 2019; 21(4): 3769-3795. doi: 10.1109/COMST.2019.2934468

[10] Kushwah GS, Ranga V. Distributed Denial of Service Attacks and Defense in Cloud Computing. In: Singh S, Sharma RM (editor). Handbook of Research on the IoT, Cloud Computing, and Wireless Network Optimization. IGI Global, 2019, pp. 41-59.

[11] Tavallaee M, Bagheri E, Lu W, Ghorbani AA. A detailed analysis of the KDD CUP 99 data set. In: IEEE 2009 Symposium on Computational Intelligence for Security and Defense Applications; Ottawa, ON, Canada; 2009. pp. 1-6.

[12] Shiravi A, Shiravi H, Tavallaee M, Ghorbani AA. Toward developing a systematic approach to generate benchmark datasets for intrusion detection. computers & security 2012; 31(3): 357-374. doi: 10.1016/j.cose.2011.12.012

[13] Rakha MA. On the Moore–Penrose generalized inverse matrix. Applied Mathematics and Computation 2004; 158(1): 185-200. doi: 10.1016/j.amc.2003.09.004

[14] ELM Origin (2004). ELM Origin [online]. Website https://elmorigin.wixsite.com/originofelm [accessed 01 March 2020]

[15] Wang LP, Wan CR. Comments on "The extreme learning machine". IEEE Transactions on Neural Networks 2008; 19(8): 1494-1495. doi: 10.1109/TNN.2008.2002273

[16] Huang GB. Reply to "comments on "the extreme learning machine"". IEEE Transactions on Neural Networks 2008; 19(8): 1495-1496. doi: 10.1109/TNN.2008.2002275

[17] Huang GB. What are extreme learning machines? Filling the gap between Frank Rosenblatt's dream and John von Neumann's puzzle. Cognitive Computation 2015; 7(3): 263-278. doi: 10.1007/s12559-015-9333-0

[18] Roschke S, Cheng F, Meinel C. An extensible and virtualization-compatible IDS management architecture. In: IEEE 2009 Fifth International Conference on Information Assurance and Security; Xi'an, China; 2009. pp. 130-134.

[19] Lo CC, Huang CC, Ku J. A cooperative intrusion detection system framework for cloud computing networks. In: IEEE 2010 39th International Conference on Parallel Processing Workshops; San Diego, CA, USA; 2010. pp. 280-284.

[20] Bakshi A, Dujodwala YB. Securing cloud from ddos attacks using intrusion detection system in the virtual machine. In: IEEE 2010 Second International Conference on Communication Software and Networks; Singapore; 2010. pp. 260-264.

[21] Modi CN, Patel DR, Patel A, Rajarajan M. Integrating signature apriori based network intrusion detection system (NIDS) in cloud computing. Procedia Technology 2012; 6: 905-912. doi: 10.1016/j.protcy.2012.10.110

[22] De LHE, Ortiz A, Ortega J, Prieto B. PCA filtering and probabilistic SOM for network intrusion detection. Neurocomputing 2015; 164: 71-81. doi: 10.1016/j.neucom.2014.09.083

[23] Sabar NR, Yi X, Song A. A bi-objective hyper-heuristic support vector machines for big data cyber-security. IEEE Access 2018; 6: 10421-10431. doi: 10.1109/ACCESS.2018.2801792

[24] Idhammad M, Afdel K, Belouch M. Semi-supervised machine learning approach for DDoS detection. Applied Intelligence 2018; 48(10): 3193-3208. doi: 10.1007/s10489-018-1141-2

[25] Ji SY, Jeong BK, Choi S, Jeong DH. A multi-level intrusion detection method for abnormal network behaviors. Journal of Network and Computer Applications 2016; 62: 9-17. doi: 10.1016/j.jnca.2015.12.004

[26] Ashfaq RAR, Wang XZ, Huang JZ, Abbas H, He YL. Fuzziness based semi-supervised learning approach for intrusion detection system. Information Sciences 2017; 378: 484-497. doi: 10.1016/j.ins.2016.04.019

[27] Yin C, Zhu Y, Fei J, He X. A deep learning approach for intrusion detection using recurrent neural networks. IEEE Access 2017; 5:21954-21961. doi: 10.1109/ACCESS.2017.2762418

[28] Vinayakumar R, Alazab M, Soman KP, Poornachandran P, Al-Nemrat A et al. Deep learning approach for the intelligent intrusion detection system. IEEE Access 2019; 7: 41525-41550. doi: 10.1109/ACCESS.2019.2895334

[29] Yang Y, Zheng K, Wu C, Niu X, Yang Y. Building an effective intrusion detection system using the modified density peak clustering algorithm and deep belief networks. Applied Sciences 2019; 9(2): 238. doi: 10.3390/app9020238

[30] Javaid A, Niyaz Q, Sun W, Alam M. A deep learning approach for network intrusion detection system. In: 2016 9th EAI International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS); New York City, United States; 2016. pp. 21-26.

[31] Dash T. A study on intrusion detection using neural networks trained with evolutionary algorithms. Soft Computing 2017; 21(10): 2687-2700. doi: 10.1007/s00500-015-1967-z

[32] Kushwah GS, Ali ST. Distributed denial of service attacks detection in cloud computing using extreme learning machine. International Journal of Communication Networks and Distributed Systems 2019; 23(3): 328-351. doi: 10.1504/IJCNDS.2019.101915

[33] Dou W, Chen Q, Chen J. A confidence-based filtering method for DDoS attack defense in cloud environment. Future Generation Computer Systems 2013; 29(7): 1838-1850. doi: 10.1016/j.future.2012.12.011

[34] Negi P, Mishra A, Gupta BB. Enhanced CBF Packet Filtering Method to Detect DDoS Attack in Cloud Computing Environment. International Journal of Computer Science Issues (IJCSI) 2013; 10(2): 142-146.

[35] Aborujilah A, Musa S. Cloud-based DDoS HTTP attack detection using covariance matrix approach. Journal of Computer Networks and Communications 2017; doi: 10.1155/2017/7674594

[36] Karimpour J, Lotfi S, Siahmarzkooh AT. Intrusion detection in network flows based on an optimized clustering criterion. Turkish Journal of Electrical Engineering & Computer Sciences 2017; 25(3): 1963-1975. doi: 10.3906/elk-1601-105

[37] Tripathi N, Hubballi N. Slow rate denial of service attacks against HTTP/2 and detection. Computers & security 2018; 72: 255-272. doi: 10.1016/j.cose.2017.09.009

[38] David J, Thomas C. Efficient DDoS flood attack detection using dynamic thresholding on flow-based network traffic. Computers & Security 2019; 82: 284-295. doi: 10.1016/j.cose.2019.01.002

[39] Wang B, Zheng Y, Lou W, Hou YT. DDoS attack protection in the era of cloud computing and software-defined networking. Computer Networks 2015; 81: 308-319. doi: 10.1016/j.comnet.2015.02.026

[40] Joldzic O, Djuric Z, Vuletic P. A transparent and scalable anomaly-based DoS detection method. Computer Networks 2016; 104: 27-42. doi: 10.1016/j.comnet.2016.05.004

[41] Buragohain C, Medhi N. FlowTrApp: An SDN based architecture for DDoS attack detection and mitigation in data centers. In: IEEE 2016 3rd International Conference on Signal Processing and Integrated Networks (SPIN); Noida, India; 2016. pp. 519-524.

[42] Hong K, Kim Y, Choi H, Park J. SDN-assisted slow HTTP DDoS attack defense method. IEEE Communications Letters 2017; 22(4): 688-691. doi: 10.1109/LCOMM.2017.2766636

[43] Tsai SC, Liu IH, Lu CT, Chang CH, Li JS. Defending cloud computing environment against the challenge of DDoS attacks based on software defined network. In: 2017 Advances in Intelligent Information Hiding and Multimedia Signal Processing; Kaohsiung, Taiwan; 2017. pp. 285-292.

[44] Kholidy HA, Baiardi F. CIDS: A framework for intrusion detection in cloud systems. In: IEEE 2012 Ninth International Conference on Information Technology-New Generations; Las Vegas, NV, USA; 2012. pp. 379-385.

[45] Modi CN, Patel DR, Patel A, Muttukrishnan R. Bayesian Classifier and Snort based network intrusion detection system in cloud computing. In: IEEE 2012 Third International Conference on Computing, Communication and Networking Technologies (ICCCNT'12); Coimbatore, India; 2012. pp. 1-7.

[46] Modi C, Patel D, Borisanya B, Patel A, Rajarajan M. A novel framework for intrusion detection in cloud. In: 2012 Fifth International conference on security of information and networks; Jaipur, India; 2012. pp. 67-74.

[47] Modi CN, Patel D. A novel hybrid-network intrusion detection system (H-NIDS) in cloud computing. In: IEEE 2013 Symposium on Computational Intelligence in Cyber Security (CICS); Singapore; 2013. pp. 23-30.

[48] Kim G, Lee S, Kim S. A novel hybrid intrusion detection method integrating anomaly detection with misuse detection. Expert Systems with Applications 2014; 41(4): 1690-1700. doi: 10.1016/j.eswa.2013.08.066

[49] Sharafaldin I, Lashkari AH, Ghorbani AA. Toward generating a new intrusion detection dataset and intrusion traffic characterization. In: 2018 4th International Conference on Information Systems Security and Privacy (ICISSP); Funchal, Madeira, Portugal; 2018. pp. 108-116.

[50] Sharafaldin I, Lashkari AH, Hakak S, Ghorbani AA. Developing Realistic Distributed Denial of Service (DDoS) Attack Dataset and Taxonomy. In: IEEE 2019 International Carnahan Conference on Security Technology (ICCST); Chennai, India; 2019. pp. 1-8.

[51] Kushwah GS, Ranga V. Voting extreme learning machine based distributed denial of service attack detection in cloud computing. Journal of Information Security and Applications 2020; 53: 102532. doi: 10.1016/j.jisa.2020.102532