Hindawi
Scientific Programming
Volume 2023, Article ID 7210034, 15 pages
https://doi.org/10.1155/2023/7210034

Research Article

Intrusion Detection System Using the G-ABC with Deep Neural Network in Cloud Environment

Nishika Gulia,[1] **Kamna Solanki**,[1] **Sandeep Dalal**,[2] **Amita Dhankhar**,[1] **Omdev Dahiya**,[3] **and N. Ummal Salmaan**[4]

[1]*Department of Computer Science and Engineering, University Institute of Engineering and Technology, Maharshi Dayanand University, Rohtak, India*
[2]*Department of Computer Science and Applications, Maharshi Dayanand University, Rohtak, India*
[3]*School of Computer Science and Engineering, Lovely Professional University, Phagwara, Punjab, India*
[4]*Department of Automotive Engineering, Aksum University, Axum, Ethiopia*

Correspondence should be addressed to N. Ummal Salmaan; ummalsalmaan90@gmail.com

Received 1 October 2022; Revised 6 November 2022; Accepted 24 November 2022; Published 28 April 2023

Academic Editor: Punit Gupta

Cloud computing plays a pivotal role in sharing resources and information. It is challenging to secure cloud services from different intruders. Intrusion detection system (IDS) plays a vital role in detecting intruder attacks, and it is also used to monitor the traffic in the network. The paper is aimed to control the attacks using the machine learning (ML) technique integrated with the artificial bee colony (ABC) named Group-ABC (G-ABC). The IDS detector has been implemented and further simulation results have been determined using the G-ABC. The evaluation has been carried out using the measures such as precision, recall, accuracy, and *F*-measure. Different attacks such as user to root (U2R), probe, root to local (R2L), backdoors, worms, and denial-of-service (DoS) attacks have been detected. The simulation analysis is performed using two datasets, namely, the NSL-KDD dataset and UNSW-NB15 dataset, and comparative analysis is performed against the existing work to prove the effectiveness of the proposed IDS. The objective of the work is to determine the intruder attacker system using the deep learning technique.

1. Introduction

In today's scenario, cloud computing plays a pivotal role in sharing resources and information and computing with predefined services such as software as a service (SAAS), platform as a service (PAAS), and infrastructure as a service (IAAS) [1]. It is challenging to secure these cloud services in multitenet, static, and dynamic environments. Therefore, cloud security is provided by using different security controls that prevent intruders by reducing the intensity level of the respective intruders [2, 3]. In the cloud, there are four different security controls, namely, detective controls, deterrent controls, corrective controls, and preventive controls [4, 5]. In these controls, deterrent control is used by alerting the system which ultimately reduces the attack level. Preventive control is used to provide robustness to the preventive actions and methods against the intruder attack. The corrective control process helps in the identification of the threat and uses measures to retrain the system to avoid future attacks [6]. The intrusion detection system (IDS) is a part of the detective control in the cloud environment used for security control mechanisms [7]. IDS can be a hardware system that includes the use of physical devices to detect the threat or a software system in which threat monitoring has been done automatically through identification and alerting the system. The classification of IDS has been done based on their functioning, location, and type of action. There are three types of IDS: anomaly-based IDS, signature-based IDS, and hybrid IDS. As per [8], the attack is known and it can be mitigated using machine learning techniques in anomaly-based IDS. This technique detects the packet attacked by the intruders using unknown patterns or signatures. It is also named a behavior-based detection system due to the modeling of user, host, and network behavior, and thus

alarm has been generated when the behavior of the system changed from the normal behavior. In signature-based IDS, the signature of the previous known threat or attack is saved in the database. The packet and signature patterns can be used for the identification of packets from intruders. The detection accuracy in this technique is very high. However, this method fails when the arriving packet signature is not available in the system. Hybrid IDS is a combination of both former methods as discussed, but it is used rarely as this method fails to provide the desired detection accuracy.

Figure 1 depicts the IDS in the cloud based on location such as network-based IDS (NIDS) and host-based IDS (HIDS). It exemplifies the placement of NIDS in the network as such that the cloud is connected via the internet to access the services by the cloud users. NIDS is used to scan the entire network to determine the data flow and detect the packets that came from which location (intruders or normal packets). If the data packets are detected by intruders, then an alert message is sent to the network administrator for prevention. As per the cloud security mechanism, NIDS acts as a safeguard against intruder attacks while IDS inspects the traffic in the network at the packet level and does not block the information. Moreover, NIDS operated at a network node and data analysis has been done in real time.

Consequently, Deshpande et al. developed HIDS-based detection system with an average sensitivity of about 96%, specificity of about 42.5%, and accuracy of about 90% with a 20 threshold level [9]. The authors considered each host with the HIDS to monitor the incoming and outgoing network traffic. HIDS is used for monitoring the data packets from the host machine or from the intruders and also initiates the alert message when there is an intruder attack. But, the drawback of the study is the requirement of the module to use the integrated system that detects the attacks for updated data logs. Devarakonda et al. integrated the HIDS and NIDS and implemented the improved dragonfly optimization technique [10]. The detection rate was 96% but limited to determining the attacks for desired results in terms of precision. Although several swarm intelligence (SI) and machine learning (ML) algorithms were developed in the literature, controlling intruder attacks in an efficient manner still is demanding [11]. The objective of the work is to determine the intruder attacker system using the deep learning technique. The contribution of the proposed work by understanding the limitations is given as follows:

(i) This paper introduced the ML techniques based on IDS with optimization technique ABC named Group-ABC (G-ABC).

(ii) The proposed study plays a pivotal role in detecting intruder attacks, and it is also used to monitor the traffic in the network with the intent to intimate the suspicious activity and intruder threats to the administrator.

(iii) A novel distributed detector is implemented, and further simulation results in terms of performance metrics such as precision, recall, accuracy, and *F*-measure have been computed. The evaluation has been carried out by comparing it with the existing work.

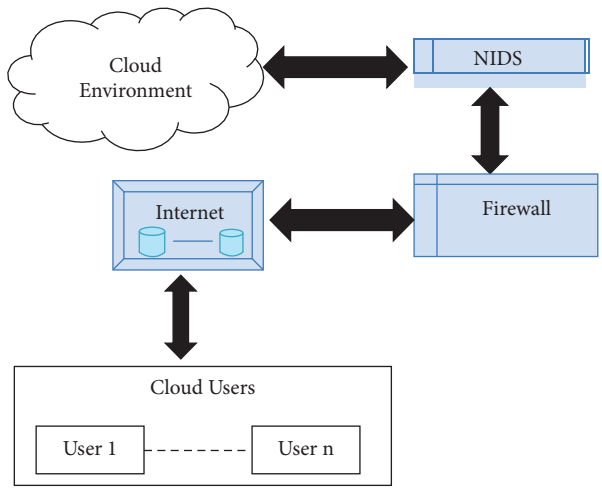

Figure 1: IDS in the cloud environment on the basis of location.

The organization of the paper is as follows. In the beginning, the introduction of different malicious attacks has been highlighted followed by the types of attacks. The next part includes the study of different past techniques based on IDS using the ML and swarm intelligence (SI) methods. The proposed methodology is depicted in the further part in which different algorithms and their working are illustrated. In the next section, the results and discussion are described with the demonstration of performance metrics. The final part ends with a valid conclusion.

2. Literature Review

The network systems are used to deploy the IDS to monitor the sources of malicious attacks by intruders. The existing research considers two different IDSs such as host-based IDS and network-based IDS. Jaber et al. presented a HIDS-based prevention model to avoid distributed denial-of-service attacks (DDoS). The authors used the hybrid prevention model of principal component analysis (PCA) and linear discriminant analysis (LDA). Further, the ant lion optimization technique was used further to select the feature and the attacks were classified using the ANN technique. The statistical techniques depict a 100% detection and prevention rate [12].

Deshpande et al. presented a detection model based on HIDS that alerts cloud users about malicious activities. The study was illustrated by dividing the work into the framework in which the detection model was developed in conjunction with OpenNebula. The data logging module and management module were illustrated well, and the KNN classifier was used to trace the call helpful to make the network more scalable. The detection accuracy of the developed model of 96% was obtained for high sensitivity. Although high sensitivity was obtained, presented work is limited to illustrating the adaptive management module and adaptive alert system [9].

Malik and Khan presented a hybrid model using the PSO and decision tree to classify the intruder attacks. In this work, the authors used the PSO for node pruning and the decision tree was used for classification purposes. The proposed model had been applied to the intruder problem considering the well-known dataset. The hybrid model was further compared with other classifiers, and performance evaluation is performed in terms of false positive rate (FPR) of about 0.136%, precision was 99.98%, detection rate was 92.71%, and accuracy of about 96.65%. The system is complex as the pruning of the decision tree for arbitrary nodes demands more generalization [13].

Practitioners developed a hybrid IDS model using the glow swarm optimization and Tabu search. The optimization was done with aim of reducing the convergence time and avoiding the trapping situation in local optima. The IDS model worked on the multilayer in which the structure was optimized for reliable communication between the users [14]. Garg et al. proposed the ABC-based malicious node identification system in conjunction with the machine learning-based Kalman filter. The experimental results were tested using the NSL-KDD dataset and further compared with the SVM and ML-based IDS system [15]. Chaturvedi et al. overcame the limitations by presenting the alert system model based on HIDS used to secure several applications and data from intruder attacks. Most internet applications required a high level of security for the business and government sectors. The proposed model was suitable to prevent attacks on cloud computing host machines [16].

Ghosh et al. presented robust IDS using the grouse mating algorithm using the NSL-KDD dataset. The authors reduced the features using this algorithm that provided an accuracy of about 81%. The simulation outcomes show that the developed IDS outperforms in comparison to other metaheuristic techniques. The scanning process is lengthy as it is difficult to detect the intrusion for large datasets [17]. Sreelatha et al. solved the security and privacy problem using the efficient cloud-based IDS using the feature selection technique named sandpiper. The extended form using the deep transfer learning classification model was developed, and features were reduced using the optimization algorithm. The behavior of the network traffic and different attacks was classified using the NSL-KDD dataset, and performance measures in terms of detection rate and false alarm rate were measured [18]. The existing studies as described above show that despite existing research in the field of IDS, still there is a need for robust IDS to overcome the limitations of complexity and alert systems in case of malicious node attacks. Moreover, researchers used the intelligent behavior of CSA in combination with the *K*-means clustering algorithm. The optimal solution was obtained globally considering the different datasets. The accuracy of the CS algorithm was measured considering the 25-feature set, and performance was evaluated using the 19 features. The precision of the developed CS algorithm is better than that of the existing ones [19]. This paper developed a robust IDS system using the ML and SI technique by interpreting the activities of the malicious nodes.

3. Proposed Methodology

The proposed methodology is divided into two phases. The first framework is designed using the swarm-based artificial bee colony (ABC) recognized as the G-ABC technique to select the most optimal features and reduce the redundant features. The dataset used to develop the IDS model is NSL-KDD. In the second framework, DNN is used as a classifier for reparability and further optimized to achieve better outcomes. The block diagram of the proposed model is given in Figure 2.

3.1. IDS Using the G-ABC. In this section, the G-ABC algorithm has been implemented to select the best features from the dataset. ABC is a SI technique inspired by the foraging nature of the artificial bees swarming over the search space. "Swarm" is named for a colony in which bees fly randomly to find a food source with a large amount of nectar. Specifically, there are three types of bees: employed, onlooker, and scout bees. The bees fall under the category of employed bees when they get the food source. The bees waiting in the hive are onlooker bees for the information exploited by the employed bees. However, the bees that have not had any experience are called scout bees. In the proposed algorithm, the scout bees worked as an explorer, whereas employed and onlooker bees were categorized as exploiters. The entire process of searching the intruders using the G-ABC is elaborated in Algorithm 1.

The flowchart of the proposed algorithm is depicted in Figure 3. It is seen that employed bee has been grouped named G-ABC to select the food source with a high amount of nectar content. In the proposed G-ABC algorithm, initially, a group of employed bees has been created. The food source is considered as the solution and the nectar content level is considered as fitness level for the particular solution. There is a single food source for each bee to be employed. Thus,

$$\text{Number of employed bees} = \text{Number of solutions}. \quad (1)$$

In this algorithm, the search cycles C_s are represented as

$$s = \{1, 2, 3, \ldots\ldots, x\}. \quad (2)$$

An initial population is distributed randomly having p solutions generated with solution s_n which is an n-dimensional vector.

$$n = \{1, 2, \ldots, z\}. \quad (3)$$

The feature of the employed bee has been selected in the solution space arranged in the form of rows and columns using the following equation:

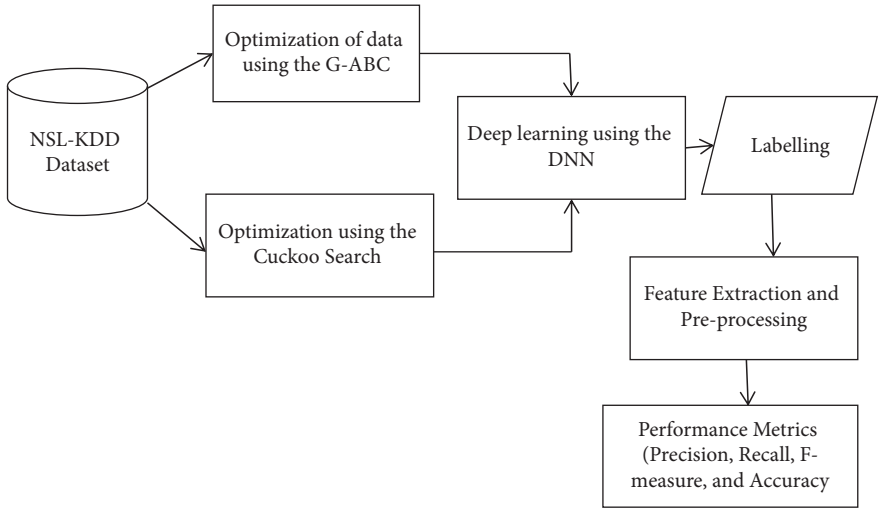

Figure 2: Block diagram of the proposed work.

$$\text{Ebee} = [\text{Current Feature Col}(1), \text{Other five Current Feature Col}(\text{Randomly})]. \tag{4}$$

Each onlooker bee uses the information from the employed bee and looks for the solution in the search space as depicted in the given equation.

$$O_{\text{bee}} = \frac{\sum_1^J E_{\text{bee}}(J)}{\text{Number of } E_{\text{bee}}}. \tag{5}$$

The fitness solution is a function of E_{bee} and O_{bee}:

$$\text{All Fit} = \text{fitness function}\left(E_{\text{bee}}, O_{\text{bee}}\right),$$
$$\text{Fit Status} = \begin{cases} 1, \text{if } E_{\text{BEE}} > O_{\text{BEE}}, \\ 0, \text{Else}. \end{cases} \tag{6}$$

In the proposed G-ABC algorithm, the local search information has been collected from the E_{bee} and O_{bee}, and global search information has been collected from scout bees and O_{bee}. Thus, the exploration process to detect the intruders has been carried out efficiently.

3.2. IDS Using the Cuckoo Search Algorithm (CSA). CS is a metaheuristic technique used to enhance the performance of the parasite. The working of the CSA majorly depends upon three rules:

(i) Cuckoo is free to choose the odd location and can lay an egg at any time.

(ii) For the new cuckoo, the highest quality of the egg is transferred to the best home.

(iii) The host is likely to settle the quantity and find the egg.

The best solution is given in an equation in which normalized data are recorded and preprocessed.

$$G_{\text{BEST}} = \frac{\sum_{i=1}^n \text{NZ}_{\text{RECORD}}(i)}{n}. \tag{7}$$

The proposed CS algorithm implies different steps. The number of features is initialized as an initial value in the dataset that has been selected. The fitness value has been initialized considering the egg values. An initial position has been evaluated considering the fitness values of the eggs. If the fitness value of the cuckoos is good or better than the host egg, then the host egg is replaced; otherwise, repeat the steps until the optimal solution is obtained.

Figure 4 shows the flowchart of the proposed CS to detect intruders in the system. CS is a global optimal solution introduced to attain the detection of attacks so that performance of the system has been improved. In the proposed algorithm, initialization has been done, and then the global best solution has been obtained. Further, the fitness of the function has been evaluated considering the threshold value. If the fitness value is satisfied, then the best global solution has been updated and the data selected are considered intruder free.

3.3. Deep Neural Network (DNN). DNN is a back-propagation NN (BPNN) that consists of one input layer, one output layer, and three hidden layers. In the input layer, the node count is directly linked to the number of features in the defined vector space extracted from the dataset and fed as input to the DNN. In the hidden layer, the number of nodes has been generated using G-ABC, and the output layer includes one node. The developed system provides a value of 1 when the classification pattern has been obtained as normal and provides a 0 value in case of intrusion.

The proposed model is divided into different phases. The first phase is used to preprocess the dataset and apply the data label to the input dataset. The second phase was used to

Input: N_{DATA} ⟵ Normalized data after preprocessing
Output: S_{DATA} ⟵ Selected data from normalized data based on their fitness
(1) Calculate Size, [Row, Col] = Size (N_{DATA})
(2) Final Record = []
(3) Count = 1
(4) **For I in range (N_{DATA} , Col)**
(5) Current Feature Col = N_{DATA} (All Row, I)
(6) All Grouped Bee Records = []
(7) **For *J* in range (5)**
(8) Ebee = [Current Feature Col (1), Other five Current Feature Col (Randomly)]
(9) Obee = $\sum_1^J$Ebee (J)/Number of Ebee
(10) Define fitness function of G-ABC
(11) All Fit Record = []
(12) Fit Status = 0
(13) **For K in range (Ebee)**
(14) **If Ebee (K) > Obee**
(15) Fit Status = 1
(16) **Else**
(17) Fit Status = 0
(18) **End–If**
(19) All Fit Record (K) = Fit Status
(20) **End–For**
(21) **End–For**
(22) **All Fit = fitness function (Ebee, Obee)**
(23) If count of non-zeros in All Fit > 1
(24) Bee Status = 1
(25) **Else**
(26) Bee Status = 0
(27) **End–If**
(28) All Bee Record (J) = Bee Status
(29) **End–For**
(30) **If count of non-zeros in All Bee Record > Average (All Bee Record)**
(31) Final Record (count) = I
(32) Count = Count + 1
(33) **End–If**
(34) **End–For**
(35) Select data from normalized data according to selected index by G-ABC
(36) S_{DATA} = N_{DATA} (All Row, Final Record)
(37) **Return**: S_{DATA} as a selected data
(38) **End–Algorithm**

Algorithm 1: G-ABC.

normalize the NSL-KDD dataset using the min-max technique. The third phase is used for splitting purposes in which data are divided into training, validation, and test sets. The fourth phase is used to construct the DNN classifier with enhanced classifiers in the prior class. The final phase is used for evaluation purposes for each model using the different parameters.

3.3.1. Labelling. Labelling has been done to label the extracted features and to recognize the intruders and normal data. There are basically four types of labelling to detect intruders or attacks. The multiclass labels assigned to the extracted features are DoS, R2L, U2R, probe, backdoor, worms, exploit, and normal data. If the label belongs to back, Neptune, smurf', teardrop, pod, and land, then the data include the denial-of-service attack (DoS) while if the data correspond to warezclient, guess_passwd, warezmaster, imap, ftp_write, multihop, phf, and spy, then detected data are supposed to be attacked by R2L. If the feature corresponds to buffer_overflow, rootkit, and loadmodule, then data are affected by U2R attack. If the feature corresponds to satan, ipsweep, portsweep, and nmap, then data are affected by probe attack; otherwise, data are normal. The labelling algorithm is designed in Algorithm 3.

3.3.2. Steps of the Proposed Algorithm. The different steps of the proposed algorithm are described as follows.

Step 1. Feature extraction: This stage is further divided into two parts. The first step is used in which the dataset contains the missing values which cannot be applied to the deep learning models and features are extracted accordingly. Therefore, the missing data are assigned the value 0, and data having information are assigned the numerical value.

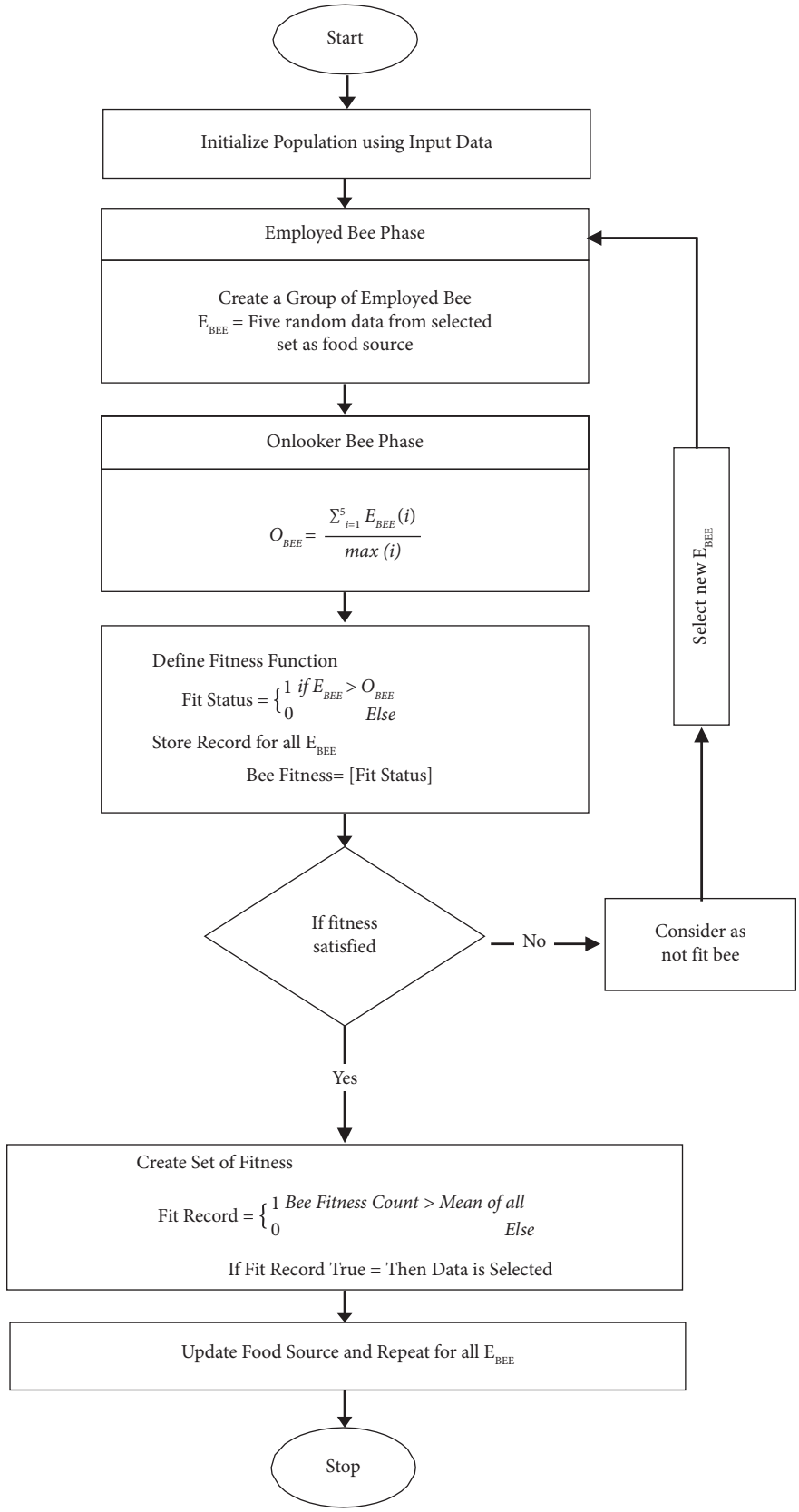

Figure 3: Flowchart of the Group-ABC.

Input: N_{DATA} ⟵ Normalized data after preprocessing
Output: S_{DATA} ⟵ Selected data from normalized data based on their fitness
(1) Calculate Size, [Row, Col] = Size (N_{DATA})
(2) Find out the non-zero element to set Global Best
(3) NZ_{RECORD} = []//Empty array to store record
(4) **For I in range ((N_{DATA} , Col)**
(5) NZ_{RECORD} (I) = $N_{DATA} > 0$
(6) **End–For**
(7) Set Global Best, $G_{BEST} = \sum_{i=1}^{n} NZ_{RECORD}(i)/n$, **Where, n is number of data**
(8) Define fitness function
(9) Define Threshold, $T = ((75 \times GBEST/100))$
(10) **If** $P_{BEST} \geq T$
(11) Fit Status = 1
(12) **Else**
(13) Fit Status = 0
(14) **End–If**
(15) Fit Index = []//To store fit value
(16) Count1 = 1
(17) **For I in range ((N_{DATA} , Col)**
(18) Data = NDATA (I)
(19) P_{BEST} = Data >0
(20) *F*-value = CSA (Fitness Function, P_{BEST}, G_{BEST})
(21) **If F-value = = 1**
(22) Fit Index (Count1) = I
(23) Count1 = Count1 + 1//Increment
(24) **End–If**
(25) **End–For**
(26) $S_{DATA} = N_{DATA}$ (:, Fit Index)
(27) **Return:** S_{DATA} as a selected data
(28) **End–Algorithm**

Algorithm 2: CSA.

Step 2. Preprocessing: This step is divided into data conversion and data normalization. The assigning of values is also referred to as encoding the non-numerical features and handling the numerical data easily. In this step, the normalization of data by setting the levels from max to min has been set. This helps the neural network to consistently build data normalization. The proposed algorithm to preprocess the data is depicted in Algorithm 4.

Step 3. Dataset: in this step, the main dataset is divided into the training set, validation set, and test set with the percentage of 70% for 1st set and 15% for both next sets.

Step 4. DNN-IDS: We proposed IDS based on three types of models: G-ABC, G-ABC-DNN, and CS-IDS. The proposed DNN-based IDS with G-ABC is illustrated in Algorithm 5.

3.4. Datasets. The efficiency of the proposed algorithm for intrusion detection systems has been investigated by using two datasets, the NSL-KDD dataset and the UNSW-NB15 dataset. The dataset includes the 43 different features used per text against DoS, R2L, U2R, and probe attack out of which 41 are the traffic inputs and the last two features are used for labelling (in case of normal attack). The NSL-KDD dataset has been used to enhance the performance of KDD data and resolve intruder problems. The second dataset is UNSW-NB15, from where attack statistics of DoS along with backdoors, exploits, DoS, and worms attacks were extracted. From both the datasets, the intruder features have been extracted without considering the payload and packet header and thus only looking for the basic information. The characteristics of the dataset are 1–9, and the content features have important information. The device easily accesses the payload information, and the features in the prescribed group lie in the range of 10–22. However, time-based features include the two-second window to analyze the traffic information and the group includes 23–31 characteristics. Such characteristics were used to determine the attacks and stretched over seconds.

4. Results and Discussion

The performance of the proposed IDS has been determined by evaluating the different parameters such as accuracy, precision, recall, and *F*-measure in terms of true positive rate (TPR), false positive rate (FPR), true negative rate (TNR), and false negative rate (FNR).

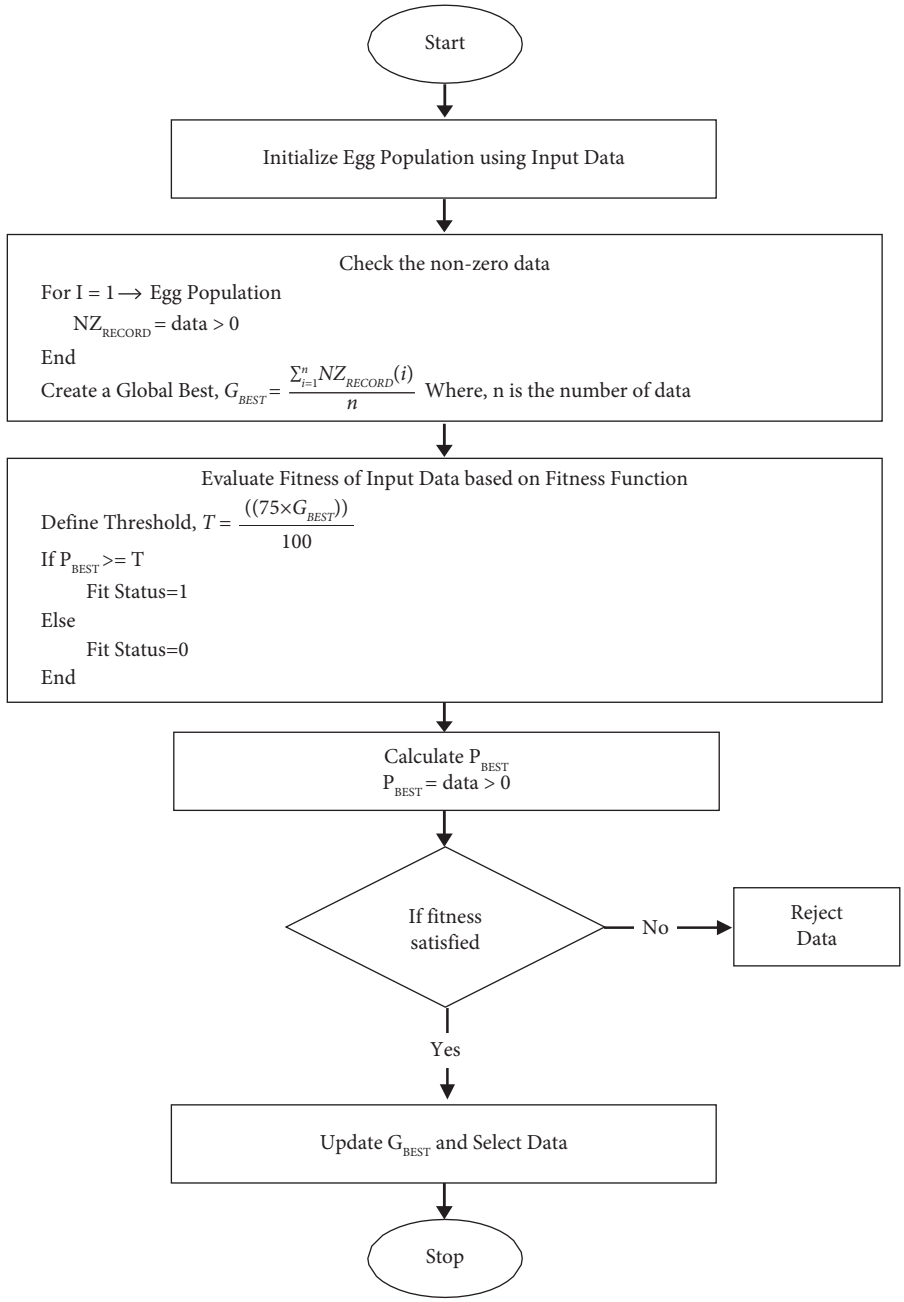

Figure 4: Flowchart of the cuckoo search for IDS.

$$\begin{aligned}
\text{Accuracy} &= \frac{\text{sum of TPR and TNR}}{\text{sum of TPR, TNR, FPR, and FNR}},\\
\text{Recall or Detection Rate} &= \frac{\text{TPR}}{\text{sum of TPR and FNR}},\\
\text{precision} &= \frac{\text{TPR}}{\text{sum of TPR and FNR}},\\
\text{F} - \text{measure} &= \frac{2 \times \text{recall} \times \text{precision}}{\text{recall} + \text{precision}}.
\end{aligned} \tag{8}$$

Table 1 depicts the accuracy of the proposed algorithm in comparison to other techniques for U2R attacks. The accuracy of the proposed approach is 99.01% while with CSA, it is 95.44%. A similar trend for precision as the proposed system shows about 99.94% while without using DNN, it is 94.11%. The recall and *F*-measure also show promising results in comparison to other techniques. Thus, it is observed that the accuracy, precision, recall, and *F*-measure of the proposed IDS are better than those of other techniques.

Table 2 depicts the accuracy, precision, recall, and *F*-measure of the proposed algorithm in comparison to the existing techniques for R2L attack. It is seen that the

```
(1) Start
(2) Calculate size of D_FILE, [Row, Col] = size (D_FILE)
(3) Label Extraction from D_FILE
(4) All Label = []//Create a blank matrix to store Labels from D_FILE
(5) For I in range (D_FILE (Row))
(6) Label (I) = D_FILE (I, Col)
(7) If Label (I) not belongs to Label
      (a) All Label (Count) = Label (I)
(8) End–If
(9) End–For
(10) Convert all Labels into five generic labels according to subclass
(11) Unique Label = []
(12) For I in range (D FILE (Row))
(13) Label (I) = All Label (I)
(14) If Label (I) ε {back, Neptune, smurf', teardrop, pod, land}
      a Unique Label (Count) = DoS
(15) Else If Label (I) ε {warezclient, guess_passwd, warezmaster, imap, ftp_write, multihop, phf, spy}
      (a) Unique Label (Count) = R2L
(16) Else If Label (I) ε {buffer_overflow, rootkit, loadmodule}
      (a) Unique Label (Count) = U2R
(17) Else If Label (I) ε {satan, ipsweep, portsweep, nmap}
      (a) Unique Label (Count) = Probe
(18) Else
      (a) Unique Label (Count) = Normal
(19) End–If
(20) End–For
```

Algorithm 3: Multiclass labelling.

```
(1) Apply preprocessing on D_FILE
(2) Preprocessed D_FILE = Replace (String with PDF (D_FILE))
(3) Apply Normalization to scale data in same range
(4) Normalized Data = []//To store normalized data
(5) For I in range (Col)
(6)    Selected Data = Preprocessed D_FILE (I)
(7)    Maximum = max (Selected Data)//Maximum value in D_FILE
(8)    Minimum = min (Selected Data)//Minimum value in D_FILE
(9)    If Maximum > 1
(10)      For J in range (Selected Data, Row)
(11)         If Selected Data (J) = = 0
(12)            Normalized Data (J, I) = 0
(13)   Else
(14)            Normalized Data (J, I)=(Selected Data (J)-Minimum)/(Maximum-Minimum)
(15)      End–If
(16)   End–For
(17) Else
(18)      For J in range (Selected Data, Row)
(19)         Normalized Data (J, I) = Selected Data (J)
(20)      End–For
(21)   End–If
(22) End–For
```

Algorithm 4: Preprocessing.

accuracy, precision, recall, and *F*-measure of the proposed system are better than those of other techniques. The accuracy of the proposed approach for R2L attack is 98.81% while with CSA, it is 95.18%, with Jaber and Rehman [7], it is 98.46%, and with Pajouh et al. [8], it is 84.68%. A similar trend for precision as the proposed system shows about 99.74% while without using DNN, it is 95.11%. The recall with Jaber and Rehman [7] is 99.3% and with Pajouh et al. [8], it is 84.47%. The *F*-measure also shows promising results in comparison to other techniques for R2L attacks.

Input: D_{FILE} ⟵ Dataset file for the simulation
Output:IDS-Net ⟵ Final Route from VS to VD in VANET CP ← Classification Parameters
(1) Final Record = []
(2) Count = 1
(3) **For I in range (Normalized Data, Col)**
(4) Current Feature Col = Normalized Data (All Row, I)
(5) All Grouped Bee Records = []
(6) **For J in range (5)**
(7) Ebee = [Current Feature Col (1), Other five Current Feature Col (Randomly)]
(8) Obee = $\sum_1^J$Ebee (J)/Number of Ebee
(9) Define fitness function of G-ABC
    (i) All Fit Record = []
    (ii) Fit Status = 0
    (iii) **For K in range (Ebee)**
    (iv) **If Ebee (K) > Obee**
    (v) Fit Status = 1
    (vi) **Else**
    (vii) Fit Status = 0
    (viii) **End–If**
  (b) All Fit Record (K) = Fit Status
  (c) **End–For**
(10) **End–For**
(11) All Fit = fitness function (Ebee, Obee)
(12) **If count of non-zeros in All Fit >1**
(13) Bee Status = 1
(14) **Else**
(15) Bee Status = 0
(16) **End–If**
(17) All Bee Record (J) = Bee Status
(18) **End–For**
(19) **If count of non-zeros in All Bee Record > Average (All Bee Record)**
(20) Final Record (count) = I
(21) Count = Count + 1
(22) **End–If**
(23) **End–For**
(24) Select Data from Normalized Data according to selected index by G-ABC
(25) Selected Normalized Data = Normalized Data(All Row, Final Record)
(26) Create Target for Model Training
(27) Target = []//Create an empty variable to store Target
(28) **For I in range (Selected Normalized Data, Row)**
(29) **If ε 1st Unique Label**
(30) Target (1, I) = 1
(31) **Else if ε 2nd Unique Label**
(32) Target (2, I) = 2
(33) **Else if ε 3rd Unique Label**
(34) Target (3, I) = 3
(35) **Else if ε 4th Unique Label**
(36) Target (4, I) = 4
(37) **Else if ε 5th Unique Label**
(38) Target (5, I) = 5
(39) **End–If**
(40) **End–For**
(41) Call Deep Neural Network with 10 Hidden Layers
(42) IDS-Net = pattern net (10)
(43) IDS-Net = train (IDS-Net, Selected Normalized Data, Target);
(44) Test Model using IDS-Net
(45) Result = sim (IDS-Net, Test Data)
(46) CP = Calculate Parameters (Result, Unique Label)
(47) **Return:** CP as Classification Parameters
(48) **End–Algorithm**

Algorithm 5: IDS with G-ABC with DNN.

Table 1: Proposed G-ABC with DNN for U2R over the machine learning metrics.

Parameters	With CSA (%)	Without DNN (only G-ABC) (%)	Proposed IDS with G-ABC and DNN (%)
Accuracy	96.38	95.44	99.01
Precision	94.22	94.11	99.94
Recall	95.1	94.71	99.86
F-measure	94.65795479	94.40904671	99.89998398

Table 2: Proposed G-ABC with DNN for R2L over the machine learning metrics.

Parameters	With CSA (%)	Without DNN (only G-ABC) (%)	Proposed IDS with G-ABC and DNN (%)	Jaber and Rehman [7]	Pajouh et al. [8]
Accuracy	95.18	94.41	98.81	98.46	84.68
Precision	93.12	95.11	99.74	99.6	84.12
Recall	96.1	95.71	99.79	99.3	84.47
F-measure	94.58653419	95.4090567	99.76499374	99.4497738	84.52

Table 3: Proposed G-ABC with DNN for probe over the machine learning metrics.

Parameters	With CSA (%)	Without DNN (only G-ABC) (%)	Proposed IDS with G-ABC and DNN (%)	Jaber and Rehman [7]	Pajouh et al. [8]
Accuracy	96.68	95.56	98.89	98.8	79.76
Precision	95.12	95.18	99.92	99	79.3
Recall	97.1	96.81	99.85	99.8	79.5
F-measure	96.09980231	95.98808063	99.88498774	99.4	79.11

Table 4: Proposed G-ABC with DNN for DoS over the machine learning metrics.

Parameters	With CSA (%)	Without DNN (only G-ABC) (%)	Proposed IDS with G-ABC and DNN (%)	Jaber and Rehman [7]	Pajouh et al. [8]
Accuracy	95.63	94.56	99.42	99.1	82.68
Precision	96.21	95.81	99.67	99.6	81.99
Recall	98.32	97.43	99.85	99.7	81.39
F-measure	97.25355678	96.61320948	99.75991881	99.2	81.3

Table 3 depicts the accuracy, precision, recall, and *F*-measure of the proposed algorithm in comparison to the existing techniques for probe attacks. It is seen that the accuracy, precision, recall, and *F*-measure of the proposed system are better than those of other techniques. The accuracy of the proposed approach for probe attack is 98.89% while with CSA, it is 95.56%, with Jaber and Rehman [7], it is 98.80%, and with Pajouh et al. [8], it is 79.76%. A similar trend for precision as the proposed system shows about 99.924% while without using DNN, it is 95.18%. The recall obtained by Jaber and Rehman [7] is 99.80% and by Pajouh et al. [8], it is 79.5%. The *F*-measure also shows promising results in comparison to other techniques for probe attacks.

Table 4 depicts the accuracy, precision, recall, and *F*-measure of the proposed algorithm in comparison to the existing techniques for DoS attacks. It is seen that the accuracy, precision, recall, and *F*-measure of the proposed system are better than those of other techniques. The accuracy of the proposed approach for probe attack is 99.42% while with CSA, it is 95.63%, with Jaber and Rehman [7]it is 99.10%, and with Pajouh et al. [8], it is 82.68%. A similar trend for precision as the proposed system shows about 99.67% while without using DNN, it is 97.43%. The recall obtained by Jaber and Rehman [7] is 99.70% and by Pajouh et al. [8], it is 81.39%. The *F*-measure also shows promising results in comparison to existing techniques for DoS attacks.

Figure 5 depicts the accuracy, precision, recall, and *F*-measure of the proposed algorithm for different attacks. It is seen that the precision of the U2R attack is better than that of the other attacks. The accuracy of the proposed approach for DoS attacks is better than that of other attacks. The recall value of the probe and U2R is almost the same, but it is less for an R2L attack. The recall value of the *F*-measure also shows promising results for DoS attacks. Similar outcomes are also obtained with the UNSW-NB15 dataset. The overall performance statistics obtained for UNSW-NB15 are shown in Figure 6. It is observed that average accuracy for different attacks remained at 98.23%, precision was 99%, recall was

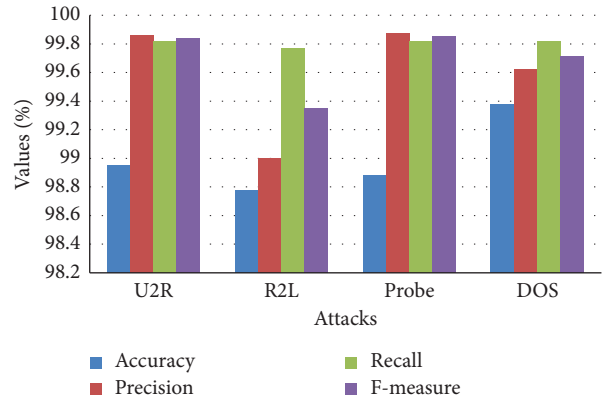

Figure 5: Proposed IDS using G-ABC with DNN for different attacks from the NSL-KDD dataset.

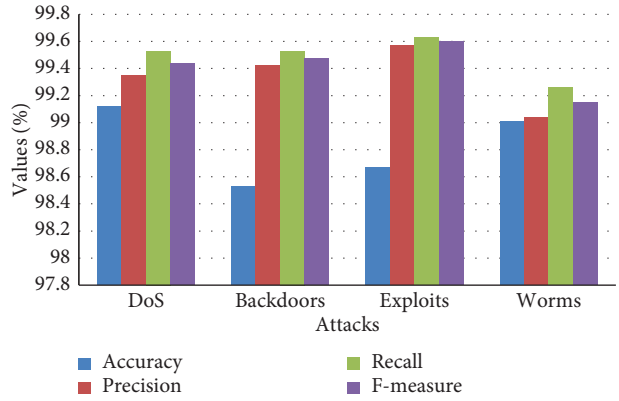

Figure 6: Proposed IDS using G-ABC with DNN for different attacks from the UNSW-NB15 dataset.

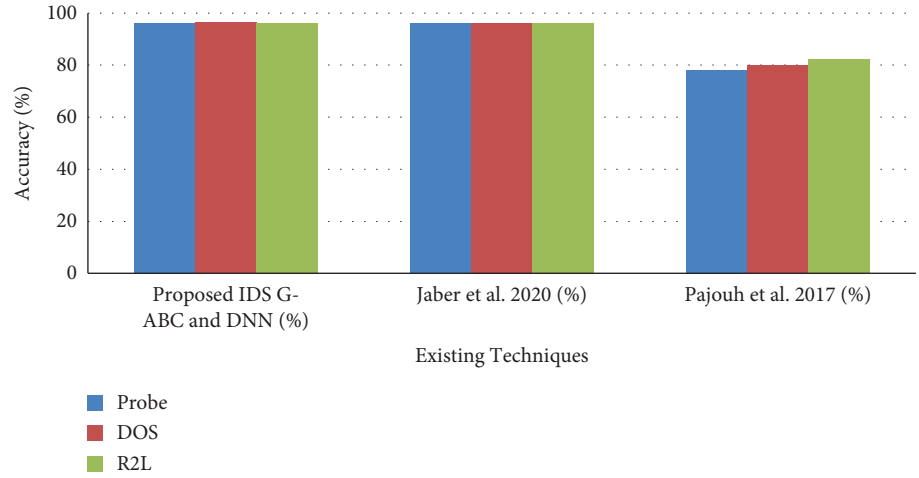

Figure 7: Comparison of accuracy of the proposed IDS with the existing techniques.

99%, and *F*-measure was 99%. The variation for the four performance parameters against four different attacks is further illustrated in Figure 7.

The overall evaluation of the proposed IDS using accuracy comparison and the recall comparison against the existing work is shown in Figures 7–10.

Figure 7 depicts the accuracy comparison of the proposed algorithm to the existing techniques for DoS, probe, and R2L attacks. It is seen that the accuracy of the proposed system is better than that of other techniques. The accuracy of the proposed approach for probe attack is 99.42% while with Jaber and Rehman [7], it is 99.10% and with Pajouh

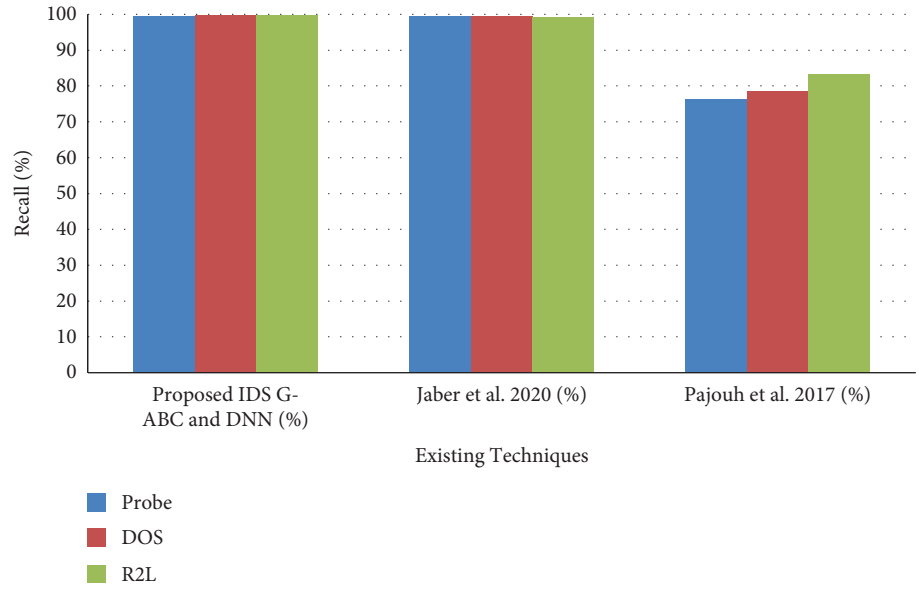

Figure 8: Comparison of recall of the proposed IDS with the existing techniques.

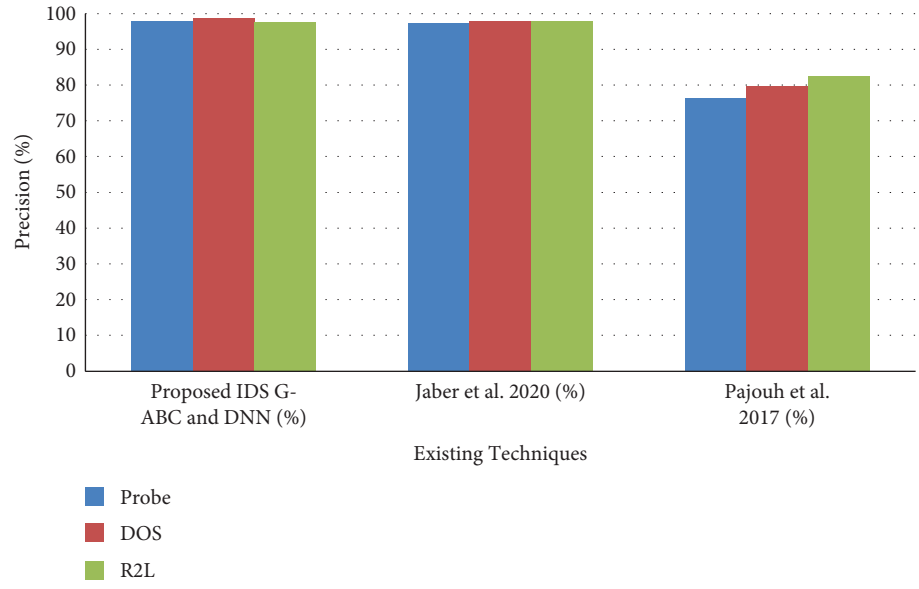

Figure 9: Comparison of precision of the proposed IDS with the existing techniques.

et al. [8], it is 82.68%. Thus, there is a 0.32% improvement in comparison to Jaber and Rehman [7]. Thus, the proposed approach shows better results in comparison to existing techniques.

Figure 8 depicts the recall comparison of the proposed algorithm to the existing techniques for DoS, R2L, and probe attacks. It is seen that the recall of the proposed system is better than that of other techniques. The recall obtained by Jaber and Rehman [7] is 99.70% and by Pajouh et al. [8], it is 81.39%. The proposed technique is improved by 0.15% in comparison to existing techniques.

Figure 9 depicts the precision of the proposed algorithm in comparison to the existing techniques for different attacks. It is seen that the precision of the proposed system is better than that of other techniques. The precision of the proposed system for probe attack shows about 99.924% while for Jaber and Rehman [7] and Pajouh et al. [8], it is 99% and 89.7%. The precision shows promising results in comparison to other techniques for probe attacks.

Figure 10 is used to depict the variations for the *F*-measure that depends upon the recall and precision. It is seen that the proposed system shows better results in comparison to existing techniques employed by Jaber and Pajouh for different attacks. In other words, it can also be understood that the better value of the precision and recall improves the *F*-measure of the proposed IDS. Overall, it is

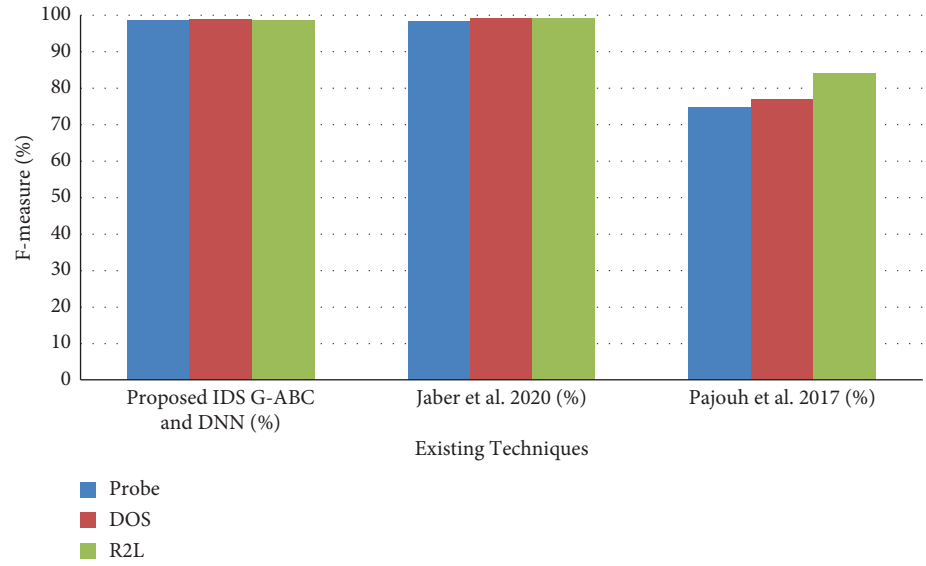

Figure 10: Comparison of *F*-measure of the proposed IDS with the existing techniques.

observed from the detailed simulation analysis that for seven attack statistics obtained from the two datasets, the proposed IDS outperformed and proved its effectiveness. This improved performance is mainly due to the integration of the two swarm intelligence algorithms, namely, ABC and CS, along with the deep learning neural architecture.

5. Conclusion

In the present work, the authors presented improved IDS for the cloud using the G-ABC with the DNN. The proposed system includes the development of the G-ABC with DNN to detect the different attacks. The G-ABC algorithm has been implemented to select the best features from the dataset. The proposed model is divided into different phases to normalize and label the input NSL-KDD and UNSW-NB15 datasets using the min-max technique. The performance of the proposed IDS has been determined by evaluating the different parameters such as accuracy, precision, recall, and *F*-measure against seven attack statistics extracted from two datasets. The proposed IDS is evaluated for the attacks, namely, DoS, probes, R2L, U2R, backdoors, worms, and exploit attacks. The simulation analysis performed using two datasets has demonstrated the success and superiority of the proposed IDS. This is mainly due to the integration of ABC and CS along with the deep learning neural architecture in the proposed IDS. In the future, more attacks will be evaluated to further extend the effectiveness of the proposed IDS.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] C. M. Mohammed and S. R. M. Zeebaree, "And others, "Sufficient comparison among cloud computing services: IaaS, PaaS, and SaaS: a review," *Int. J. Sci. Bus.*,vol. 5, no. 2, pp. 17–30, 2021.

[2] C. Modi, D. Patel, B. Borisaniya, H. Patel, A. Patel, and M. Rajarajan, "A survey of intrusion detection techniques in Cloud," *Journal of Network and Computer Applications*, vol. 36, no. 1, pp. 42–57, Jan. 2013.

[3] S. Iqbal, M. L. Mat Kiah, B. Dhaghighi et al., "On cloud security attacks: a taxonomy and intrusion detection and prevention as a service," *Journal of Network and Computer Applications*, vol. 74, pp. 98–120, Oct. 2016.

[4] S. Velliangiri and J. Premalatha, "Intrusion detection of distributed denial of service attack in cloud," *Cluster Computing*, vol. 22, no. 5, pp. 10615–10623, 2019.

[5] B. Khadka, C. Withana, A. Alsadoon, and A. Elchouemi, "Distributed denial of service attack on cloud: detection and prevention," in *Proceedings of the 2015 International Conference and Workshop on Computing and Communication (IEMCON)*, Vancouver, BC, Canada, December 2015.

[6] T. Tsegaye and S. Flowerday, "Controls for protecting critical information infrastructure from cyberattacks," in *Proceedings of the 2014 World Congr. Internet Secur. WorldCIS*, pp. 24–29, London, UK, January 2014.

[7] A. N. Jaber and S. U. Rehman, "FCM--SVM based intrusion detection system for cloud computing environment," *Cluster Computing*, vol. 23, no. 4, pp. 3221–3231, 2020.

[8] H. H. Pajouh, G. Dastghaibyfard, and S. Hashemi, "Two-tier network anomaly detection model: a machine learning approach," *Journal of Intelligent Information Systems*, vol. 48, no. 1, pp. 61–74, 2017.

[9] P. Deshpande, S. C. Sharma, S. K. Peddoju, and S. Junaid, "HIDS: a host based intrusion detection system for cloud computing environment," *Int. J. Syst. Assur. Eng. Manag.*,vol. 9, no. 3, pp. 567–576, 2018.

[10] N. Devarakonda, S. Anandarao, and R. Kamarajugadda, "Detection of intruder using the improved dragonfly optimization algorithm IOP Conference Series: materials Science and Engineering," *IOP Conference Series: Materials Science*

and Engineering, vol. 1074, no. 1, Article ID 012011, February 2021.

[11] M. Ramaiah, V. Chandrasekaran, V. Ravi, and N. Kumar, "An intrusion detection system using optimized deep neural network architecture," *Transactions on Emerging Telecommunications Technologies*, vol. 32, no. 4, Article ID e4221, 2021.

[12] A. N. Jaber, M. F. Zolkipli, H. A. Shakir, and M. R. Jassim, "Host based intrusion detection and prevention model against DDoS attack in cloud computing," in *Proceedings of the International Conference on P2P, Parallel, Grid, Cloud and Internet Computing*, pp. 241–252, Taichung, Taiwan, October 2017.

[13] A. J. Malik and F. A. Khan, "A hybrid technique using binary particle swarm optimization and decision tree pruning for network intrusion detection," *Cluster Computing*, vol. 21, no. 1, pp. 667–680, 2018.

[14] M. Manickam and S. P. Rajagopalan, "A hybrid multi-layer intrusion detection system in cloud," *Cluster Computing*, vol. 22, no. S2, pp. 3961–3969, 2019.

[15] S. Garg, K. Kaur, S. Batra et al., "En-ABC: an ensemble artificial bee colony based anomaly detection scheme for cloud environment," *Journal of Parallel and Distributed Computing*, vol. 135, pp. 219–233, 2020.

[16] A. K. Chaturvedi, P. Kumar, and K. Sharma, "Proposing host-based intruder detector and alert system (HIDAS) for cloud computing," in *Proceedings of the International Conference on Communication and Artificial Intelligence*, pp. 579–587, Mathura, India, October 2021.

[17] P. Ghosh, Z. Alam, R. R. Sharma, and S. Phadikar, "An efficient SGM based IDS in cloud environment," *Computing*, vol. 104, no. 3, pp. 553–576, 2022.

[18] G. Sreelatha, A. V. Babu, and D. Midhunchakkaravarthy, "Improved security in cloud using sandpiper and extended equilibrium deep transfer learning based intrusion detection," *Cluster Computing*, vol. 25, pp. 1–16, 2022.

[19] M. Imran, S. Khan, H. Hlavacs, F. A. Khan, and S. Anwar, "Intrusion detection in networks using cuckoo search optimization," *Soft Computing*, vol. 26, no. 20, pp. 10651–10663, 2022.