Computer Systems
Science and Engineering

Tech Science Press

DOI: 10.32604/csse.2023.036267
Article

Adaptive Butterfly Optimization Algorithm (ABOA) Based Feature Selection and Deep Neural Network (DNN) for Detection of Distributed Denial-of-Service (DDoS) Attacks in Cloud

S. Sureshkumar[1,*], G .K. D. Prasanna Venkatesan[2] and R. Santhosh[3]

[1]Department of Computer Science and Engineering, Karpagam Academy of Higher Education, Coimbatore, 641021, India
[2]Dean Engineering, Faculty Engineering, Karpagam Academy of Higher Education, Coimbatore, 641021, India
[3]Department of Computer Science and Engineering, Faculty of Engineering, Karpagam Academy of Higher Education, Coimbatore, 641021, India

*Corresponding Author: S. Sureshkumar. Email: sureshkumarcse2022@gmail.com
Received: 23 September 2022; Accepted: 07 April 2023; Published: 26 May 2023

Abstract: Cloud computing technology provides flexible, on-demand, and completely controlled computing resources and services are highly desirable. Despite this, with its distributed and dynamic nature and shortcomings in virtualization deployment, the cloud environment is exposed to a wide variety of cyber-attacks and security difficulties. The Intrusion Detection System (IDS) is a specialized security tool that network professionals use for the safety and security of the networks against attacks launched from various sources. DDoS attacks are becoming more frequent and powerful, and their attack pathways are continually changing, which requiring the development of new detection methods. Here the purpose of the study is to improve detection accuracy. Feature Selection (FS) is critical. At the same time, the IDS's computational problem is limited by focusing on the most relevant elements, and its performance and accuracy increase. In this research work, the suggested Adaptive butterfly optimization algorithm (ABOA) framework is used to assess the effectiveness of a reduced feature subset during the feature selection phase, that was motivated by this motive Candidates. Accurate classification is not compromised by using an ABOA technique. The design of Deep Neural Networks (DNN) has simplified the categorization of network traffic into normal and DDoS threat traffic. DNN's parameters can be fine-tuned to detect DDoS attacks better using specially built algorithms. Reduced reconstruction error, no exploding or vanishing gradients, and reduced network are all benefits of the changes outlined in this paper. When it comes to performance criteria like accuracy, precision, recall, and F1-Score are the performance measures that show the suggested architecture outperforms the other existing approaches. Hence the proposed ABOA + DNN is an excellent method for obtaining accurate predictions, with an improved accuracy rate of 99.05% compared to other existing approaches.

Keywords: Cloud computing; distributed denial of service; intrusion detection system; adaptive butterfly optimization algorithm; deep neural network

1 Introduction

Cloud computing is a developing trend in the Information Technology (IT) industry since it provides customers with valuable services on demand. In essence, public clouds are available over the Internet [1]. Private clouds reside in an organization's premises. Hybrid clouds combine private and public clouds to deliver services. For enterprises to move to the cloud, cloud providers must ensure a high level of security for their clients. Cloud service providers must assure client security through firewalls and IDS by upgrading cloud architecture [2]. Although cloud computing has many advantages, the Internet's open architecture makes it vulnerable to cyber-attacks. A DDoS attack consumes resources and denies legitimate cloud users access [3]. In its simplest form, a handler recruits online zombies to launch a coordinated attack on a predetermined host. The onslaught has grown in size and sophistication, and extortion is one of the critical motivations [4]. Intrusion Detection System (IDS) assesses the safety condition of the network by collecting and evaluating diverse threats and stores them as detailed records, that forms the primary base for checking and assessing the network [5]. Data confidentiality, availability, and integrity are all compromised by intrusion. An attacker compromises network security by gaining access, stealing assets, gaining privileges, exceeding performance limits, and injecting harmful activities into network traffic. IDS were formed to analyze/monitor network traffic and determine its normality [6]. However, contemporary IDS exhibit an increased rate of false alarms, resulting in a vast volume of duplicated IDS log data. Reducing data size becomes an essential processing step. With fewer dimensions and faster categorization, feature selection is known to decrease massive data DDoS assault and creates a considerable volume of harmful data on the network, making it difficult to detect. Data mining for cyber analytics helps detect intrusions. Many techniques based on machine learning have been developed [7]. Feature selection approaches also help reduce dataset dimensionality. The data utilized in this survey comes from NSL-KDD and CICIDS2017. The first approach utilizes ABOA to identify features, and the second uses DNN to classify them. The specified ABOA features were used for classification using DNN, and the results were compared for DDoS attack detection. This work's essential contribution and motivation are:

- A unique DNN architecture with ABOA for learning useful feature representation from network traffic is proposed.
- The proposed architecture has been compared to three different conventional machine learning algorithms using NSL-KDD and CICIDS2017 datasets. The suggested scheme also outperforms or outperforms competitors in metrics such as accuracy, precision, recall, and F1-Score.

Section 2 reviews the current research on DDoS attack detection techniques. Section 3 proposes an ABOA and DNN model. Section 4 explains the findings and discussion of the experiments. Section 6 provides a conclusion for the work and suggests further research topics.

2 Related Work

This chapter contains research on recognizing DDoS assaults in network traffic. Intruders keep developing new ways to fool defense systems, inflict harm to service providers, and illegally abuse existing software. Recent DDoS attacks have exploited emerging technology, such as the Internet of Things (IoT), to great advantage [8]. Some earlier research focused on using ML to counteract obsolete attacks that no longer affect service providers. Moreover, several studies on DDoS assaults concentrated primarily on cloud computing environments [9–11], with little data on the deployed Machine Learning (ML) methods and no consideration of other modern networking contexts.

Bhardwaj et al. [12] suggested an efficient Auto Encoder (AE)/DNN architecture for DDoS attack classification. Initially, a simple AE and DNN model is generated with random hyperparameter values. This standard model is optimized to produce an AE and DNN model that improves classification outcomes. These decreased and important properties are sent into an enhanced DNN with more excellent detection and false rate for classification.

Chiba et al. [13] presented a DNN-based hybrid optimization method. The hybrid optimization uses the Improved Genetic Algorithm (IGA) and Simulated Annealing Algorithm (SAA). Genetic Algorithm (GA) is enhanced by parallel processing and fitness value hashing. Shone et al. [14] utilized Nonsymmetric Deep AE (NDAE) (deep method) and Random Forest (RF) to classify incursions. The encoded representation of two NDAEs has been given to RF for classification. This method's RF qualities, including reduced bias, overfitting reduction, and resilience to outliers, are particularly beneficial, although the training period is reduced by 98.81%. The Simulated Raindrop Algorithm (SRD) is used for modeling the defense mechanism in the form of a mathematical optimization assignment by Bhagat et al. [15]. The method maximizes throughput. Simulation of SRD with Intelligent Waterdrop Algorithm (IWD) showed that SRD requires roughly 40%–45% fewer iterations to reach convergence than IWD. The results indicate that the suggested strategy is very efficient at limiting Assaults and delivers more robustness.

Bhushan et al. [16] discussed many fundamental properties of Software-defined networking (SDN), making it an excellent networking solution for cloud computing. Furthermore, express the flow table space of a switch by utilizing a queuing theory-based mathematical model. Additionally, offer a unique flow-table sharing strategy to prevent the SDN-based cloud from overflow table overloading DDoS threats. Amma et al. [17], tuned vector convolutional deep neural network (TVCDNN) is developed by adjusting the topology and parameters of DNN, applying binary and real cumulative incarnation (CuI) correspondingly. Incoming traffic is classified using an adaptive hybrid technique presented in [18]. The study uses the NSL-KDD database to evaluate the merits of the approaches under consideration. The Mean Absolute Deviation technique with Random Forest classifier (MAD-RF) excels the other ensembles, according to the results of this study. Four filter approaches in [19] are combined to produce an ensemble-based multi-filter feature selection technique. Kushwah et al. [20] proposed an enhanced Self-adaptive evolutionary extreme learning machine for detecting DDoS attacks (SaE-ELM). Two additional characteristics have been added to the SaE-ELM model.

First and foremost, it can adapt to the optimal crossover operator. Second, it can automatically calculate the optimal amount of neurons in the hidden layer. The method's learning and classification abilities are enhanced because of these features. The Oppositional Crow Search Algorithm (OCSA), that merges the Crow Search Algorithm (CSA) and Opposition Based Learning (OBL) approach, was introduced in [21] as a useful DoS attack detection solution. An OCSA-based FS and Recurrent Neural Network (RNN) classifier are employed in a suggested system. OCSA algorithm is helpful in choosing the most critical features, that are then passed to the RNN classifier. Detection of these assaults is increasingly being done using deep learning techniques. A suitable deep neural network structure for detecting DDoS attacks in existing deep feature learning algorithms has resulted in low efficiency regarding the accuracy and false alarms. Hence, the performance of deep Learning is commendable on the massive volume of data compared with conventional machine learning, that functions remarkably well applying the rules over the meagre data. In Machine Learning (ML), feature selection has to be carried out by specialists and ML performance relies on the features extracted for learning purposes. However, in the case of DL, the extraction of high-level features is directly performed from the data. This way, DL decreases the cost incurred during feature selection, and DL algorithms consume more

time for training but reduced time for testing. If ML is considered, only some algorithms are contrary to it, i.e., taking reduced training time and more testing time. Deep learning with feature selection for DDoS recognition is suggested in this study.

3 Proposed Methodology

However, there are several techniques for detecting DDoS attacks. The rising number and intensity of attacks recently, as well as the ever-expanding attack paths, make it essential to improve existing detection systems. As shown in Fig. 1, this research proposes a unique architecture combining ABOA for feature selection with DNN to classify network traffic into benign and DDoS attack traffic. The reconstructed network is smaller and less prone to overfitting while reducing the reconstruction error. Experiments on the CICIDS2017 and NSL-KDD standard datasets have been carried out for testing purposes. Existing algorithms do better on the NSL-KDD dataset than the proposed strategy, producing competitive results on the CICIDS2017 dataset.

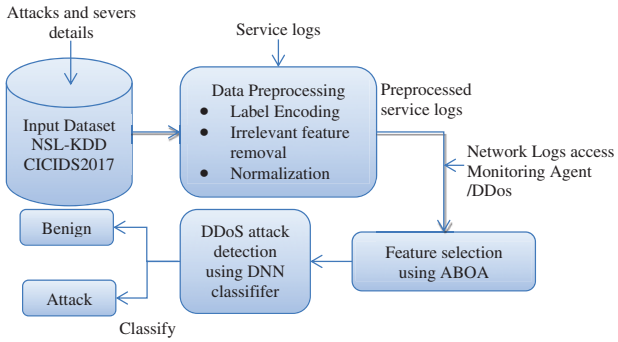

Figure 1: Architecture diagram of proposed ABOA with DNN based DDoS attack detection

3.1 Input Dataset

NSL-KDD: The University of New Brunswick's KDD'99 dataset has been updated, cleaned, and reworked into the NSL-KDD dataset. In 1999, the KDD cup was an international competition for data mining and knowledge discovery tools. The aim was to create IDS that could tell the difference between a good and a wrong network. The KDD dataset was created by compiling a significant volume of network traffic. Using the KDD database, NSL-KDD has been created In the NSL-KDD database, there are 43 varied features. More than 40 of these features are related to traffic, and the remaining two are the class designation (normal or attack) and the severity of the attack. DoS, probe, U2R, and R2L are just four of the 37 possible threats.

CICIDS2017: The CICIDS dataset has been selected as the second dataset for the proposed research investigations. This paper can use this dataset because it contains recent attacks and all the features needed. The data set includes traces of network traffic, both good and bad. There are a total of 84 features in this labeled dataset. As a final feature, a class label can be used to identify whether the traffic is malicious or non-harmful.

Data Preprocessing: The first stage is to increase the quality of the input data, that again improves the outcome's efficiency and the mining procedure's performance. Label encoding, feature elimination, and normalizing are the three preprocessing techniques that have been applied.

3.2 Feature Selection Based on ABOA

This is a client-side assault on an existing knowledge base, with rules and diagnoses suitable for network protocols for successful feature selection optimization on attack detection. As a result, these events can be simply updated without affecting the network flow rules. Suppose the user has exhibited reasonable conduct over an extended period. In that case, the DNN subset groupings can be used to assess whether or not they are legitimate (data maintained in a learning base). Repetitive feature groups are divided into portioned groups in the hypothesis dimensionality space to improve the performance achieved with feature learning algorithms. This diminishes the complexity requirement for dividing the data by inner and outer trained sets in the sub-category of capturing features. The high performance of feature evaluation criterion independent of tasks and the capacity of meta-heuristic optimization ways to identify the optimal feature subsets encourage ABOA to use an innovative phenomenon of modifying the sensory modality of BOA during the optimization process to obtain improved FS outcomes. The suggested system is based on a collective behavior method that uses random traffic models acquired by network devices through stream protocols to derive inferences about traffic through virtualized monitored data logs.

BOA: A novel optimization approach mimicking butterflies' food-forging strategy is employed in this work. Based on how butterflies create food, BOA is a metaheuristic algorithm inspired by nature [22]. For each aroma in BOA, there's a distinct scent and unique flavor. The policy benchmarks which set BOA apart from other meta-heuristics are based on this principle. Three key terms distinguish and care for the procedure: sI, sM, and pE. Tactic modalities are employed in sensory modality methodologies to measure and compare the types of vitality, and the method alludes to raw information used by sensors. Current modes include light, sound, and temperature; however, in BOA, modality is the scent of a butterfly, and sI is how much physical stimulus intensity there is. sI concerns how well a solution or butterfly will fly in a BOA competition. When a butterfly emits a large amount of perfume, it attracts other butterflies in the region. The increase in intensity indicates the butterfly's or solution's power. At the same time, pE is the parameter that allows for normal expression and the butterfly's or solution's changing degree of absorption. Authors have performed various stimulus estimating studies on bugs, critters, and people, and they assume that as the amount is increased, insects are becoming less sensitive to environmental variations. The scent of BOA has been identified as in Eq. (1) [23]:

$$frag = sMsI^{pE} \tag{1}$$

In which '$frag_i$' denotes the apparent level of smell. The global and local search phases constitute the two most important stages of the process. An appreciated domain's aroma is amplified. As a result, butterfly B emits a scent that can be detected virtually anywhere in the vicinity. With each step forward, it moves closer to its goal, the fittest butterfly fB, which can be shown as in Eq. (2)

$$x_i^{iter+1} = x_i^{iter} + (rand^2 \times fB - x_i^{iter}) \times frag_i \tag{2}$$

where in x_i^t is indeed the ith butterfly's solution vector x_i in iteration number *iter*. The best current solution discovered between all solutions in the present phase is denoted by *fB*. $frag_i$ indicates the scent of ith butterfly, while rand refers to a random number within the range [0, 1]. $rand < sP$ is also used here, with *sP* denoting the switching probability. The neighbourhood or local search step is described as in Eq. (3):

$$x_i^{t+1} = x_i^t + \left(rand^2 \times x_j^{iter} - x_k^{ite}\right) \times frag_i \tag{3}$$

x_j^{iter} and x_k^{iter} are the jth and kth butterflies in the search space, correspondingly. If x_k^t includes a site with an identical swarm and *rand* refers to a random number within the range [0, 1], x_i^{iter+1} becomes a neighbourhood random walk. Local and global butterfly searches for food, and mates can co-occur. As a consequence, BOA alternates among regular global quests and local searches using a switch probability sP.

Proposed ABOA for feature selection: The feeding habits of butterflies inspired the design of the Butterfly Optimization Algorithm. The primary aim of the BOA method is to identify the best solution in a complex, multi-dimensional environment. Butterfly positions are randomly assigned in the environment with specific constant characteristics to obtain local and global optimal places. The algorithm adjusts a parameter to maximize the position of the particles (butterflies). The three primary parameters of the butterfly algorithm govern the movement of particles (butterflies) from their initial random location to the best possible solution. Butterflies employ their sense of smell and scent to communicate with one another. Butterflies communicate using the simple concept of sense, which is based on the three parameters. Sensor modality refers to the parameter whereby sensors measure and analyze the form of energy. More than one butterfly can interact with each other and travel toward a better one by emitting a smell that is distinct from the bad one, thanks to sensor modalities. The sI parameter is used to indicate a physical/actual stimulus. To attract more butterflies, a better butterfly must have a more pungent scent and a higher value in terms of fitness. The region's particles can respond compressed thanks to the setting pE. The particle's sensitivity to stimulus changes decreases with the increasing strength of the particle's stimulus. The critical concerns for the variance of sI and formulation of fragrance *frag* are the fragrance *frag* and the stimulus intensity sI. The objective function provides the values of sI that are encoded in the aim function, whereas *frag* is a relative value that is computed as in Eq. (4)

$$frag = sMsI^{pE}\left(1 - \frac{N_t - N_f}{N_t}\right) \tag{4}$$

DNN recognition rate was employed as a quality metric in the set of features used in the goal function design in this study. Here the overall size of features is represented by (N_t), and the size of the feature subset is defined as (N_f). This option in the BOA sets the butterflies' sensory modality to "smell" other butterflies in the search arena and directs them to those who release the most scent throughout their search. There was a time when sM was assumed to be constant in the BOA algorithm [23,24], but this has been replaced by c being adjusted depending on how many times iterations are performed in the enhanced BOA. The algorithm's performance improves as the sM parameter's value rises. The method of increasing performance by adjusting the sM parameter during execution by increasing the number of iterations has been demonstrated using various benchmark functions. Sensory modality c values are adjusted during the iterative search procedure using an adaptive BOA algorithm introduced in this study. As per Eq. (5), new SM values would seriously influence BOA's efficiency and performance. This means that the BOA will perform significantly faster than the standard BOA.

$$sM^{iter} = SM^{iter-1} * \frac{\left(10.0 - \frac{5}{0.9}\right)^2}{MaxGen} \tag{5}$$

where t refers to the current iteration number when running the method, and MaxGen indicates the method's highest number of iterations. Adaptive mechanisms for one sensory modality are created and implemented in an algorithm in this suggested study. Sensory modality sM adjustment mechanisms

give the algorithm useful qualities and aid in obtaining diversified search in search space. BOA method performs better because of the c's dynamic behavior. Butterflies in BOA, which are equipped with the sensory modality, are capable of a wide range of exploration and can discern the scent of other butterflies. Fig. 2 shows the ABOA flowchart, and Algorithm 1 explains the pseudo-code of ABOA.

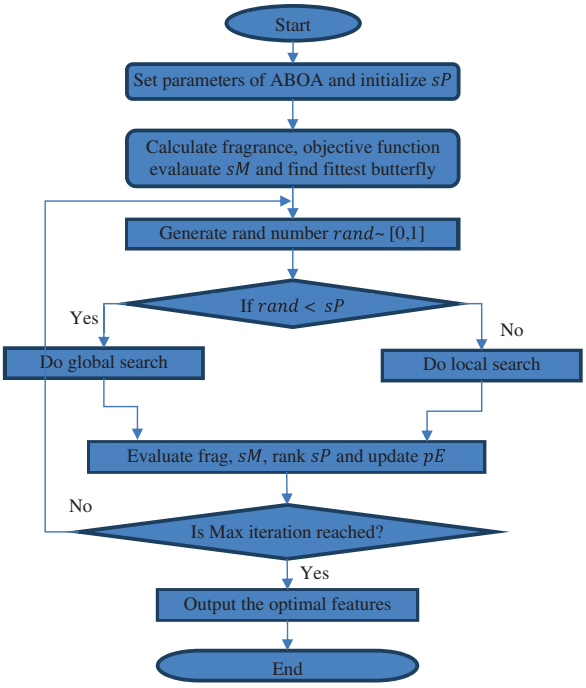

Figure 2: Flowchart of proposed ABOA-based feature selection

Algorithm 1: Pseudo-Code for ABOA for optimal feature selection

Input: Input Dataset (KDD-NSL and CICIDS)
Output: Feature subset selection (optimal features)
Begin
split Dataset into Training and Test set;
Generate initial population of N butterflies $x_i = (i = 1, 2, \ldots, n)$
Stimulus intensity sI at x_i is determined by $frag\,(x_i)$
Describe sensor modality sM, power exponent pE and switch probability sP
while stopping criteria not met do
for every butterfly B in a population do
Calculate fragrance for B using Eq. (4)
end for
Find the best B
for every butterfly B in a population, do
generate a random number *rand* from [0,1]
if $rand < sP$
Travel towards best butterfly/solution through Eq. (2)
else

(Continued)

Algorithm 1: Continued

Move in random applying Eq. (3)
end if
end for
Revise the value of sM
end while
while max iterations attained or the termination criterion is not satisfied, do objective function of butterflies assessed depending on Eq. (5); accuracy of the test set by DNN classifier
Compute classification accuracy of a test set of the chosen feature subset;
return best butterflies (the selected feature subset);
End while
End

DDoS attack detection using DNN classifier

Deep learning is employed in this project to identify the DDoS attack. For the most part, DNN comprises an input, a hidden, and an output layer. Every artificial neuron or node in a DNN is connected to every other neuron or node in the same layer.

Feed-forward DNN processes and propagates the information from the input layer to its final output layer using hidden layers. Each layer of the device has a processing element (PE). In the plan, hubs are referred to as PEs. In Fig. 3, an input node is illustrated as a result of the suggested DDoS attack detection parameters. Each layer of the device has its own set of weights. During the preparation, the importance of the sheet is balanced. Before achieving the output layer, information is transmitted and prepared by heading. The output is generated using the forward method after subtracting the desired output from the actual output, and an error value can be determined [25].

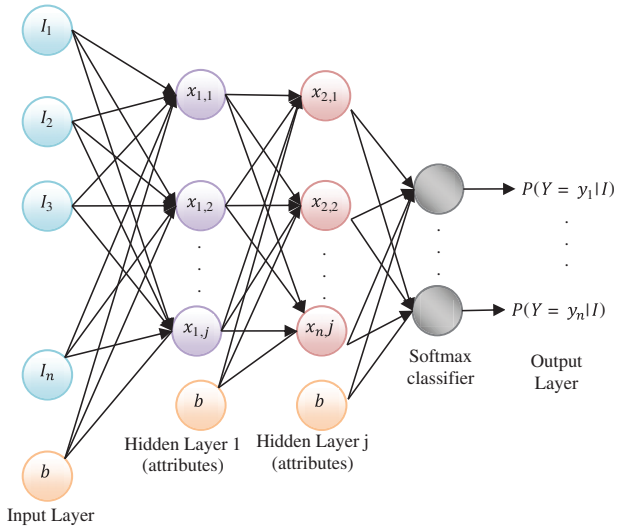

Figure 3: DNN structure

DNN is put into action in the steps below:

Step 1: The first layer of the DNN receives input data from ABOA, and the outputs of this layer can be regarded as reflecting the existence of low-level properties such as lines and edges in the image. In the DNN structure, the input i and secret h units are introduced sequentially.

$$F(i,h) = -\sum_{u=1}^{U}\sum_{v=1}^{V} I_{uv} i_u h_v - \sum_{u=1}^{U} bi_u - \sum_{u=1}^{U} bh_v$$

Here I_{uv} (input features) is a symmetrical concept of interaction between the i_u input unit and the hidden unit h_v; b means the input and hidden units bias, u, v means hidden units.

Step 2: To quantify the existence of higher-level qualities, these characteristics are coupled with subsequent layers, e.g., lines are connected into forms that are further combined into sets of forms.

Step 3: The hidden and input units are parallel and modified in the supplied visible and hidden units. Every iteration of the training process builds on the previous one, creating a multi-layered structure. The weight of the current network and biases are used to split the DNN layer values achieved by the units.

Step 4: Lastly, all this information suggests that the network may or may not have a corrupted image. In the early stages of threat recognition, DNNs can achieve greater efficiency because of this deep hierarchy.

4 Experimental Results and Discussion

Network intrusion detection datasets NSL-KDD and CICIDS2017 were used in this study. Four forms of attacks are launched on the NSL-KDD dataset: DoS, Probe, R2L, and U2R. Additionally, each of these categories has a total of 37 possible attack types. Consequently, collected data from the dataset that corresponds to DDoS attacks, including Back, Land, Neptune, and Pod; Smurf; Teardrop; Mailbomb; Processtable; Udpstorm; Apache2; and Worm are used in this work. After extraction, there are 148517 records and 43 feature columns in the KDDTrain and KDDDTest datasets. When there are categorical values in three columns, one-hot encoding is helpful to transform them into numeric values. During this step, 43 feature columns are mapped onto 124. The min-max scaler is used to standardize the values between 0 and 1. A sample of the input training dataset is later considered for training DNN to avoid overfitting. To reduce generalization error, the training of DNN is performed on a random sample having 25000 data. ABOA generates data that is optimally encoded. DNN uses these optimal data to categorize attack and non-attack traffic. The database is split into two sections: training and testing. DNN is fed a 75% training set that has 40 input features.

CICIDS2017 has 225720 data records and 85 feature columns. They are not fed to AE and are removed from the flow ID. These features are recognized as informative and will be provided directly to DNN. Fifteen thousand records are chosen at random. The column values are standardized to [0,1] by applying the min-max scaler. This gives the AE 24994 records and 78 columns to train. The AE splits these 78 columns into 23. The ideal characteristics are provided to DNN for categorization. The dataset is divided into 75% training and 25% testing. DNN is fed a 75% training set containing input features. The tests were done on a Windows 10 64-bit PC with 16 GB RAM and an Intel(R) Core-i7 CPU. VMware Workstation was used to create a cloud server with several virtual machines. The detection system's performance is defined by how well it can classify incoming communications. The proposed approach [ABOA+DNN] was compared to current methods such as AE+SVM, Naive AE+DNN, and optimized AE+DNN. These measures are described below.

Recall: Calculates the number of useful class predictions based on positive examples in the database as in Eq. (6).

$$Recall = \frac{TP}{TP + FN} \tag{6}$$

Precision: Calculates the percentage of positive class predictions that are truly positive as in Eq. (7).

$$Precision = \frac{TP}{TP + FP} \tag{7}$$

F-measure: Calculates a single score that accounts both for precision and recall issues as in Eq. (8):

$$F - Measure = \frac{(2 * Precision * Recall)}{(Precision + Recall)} \tag{8}$$

Accuracy: It's a ratio of the total samples that were rightly classified to the overall number of samples as in Eq. (9).

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

4.1 Precision Result Comparison

Table 1 shows the precision comparison of suggested ABOA + DNN, AE + SVM, Naive AE + DNN, and refined AE + DNN classifiers.

Table 1: Precision comparison

Input data	AE + SVM	Naive AE + DNN	Optimized AE + DNN	ABOA + DNN
5	81	85	86	90
10	79	82	84.5	92
15	76	84	86.15	93
20	85	86	90.15	94.5
25	80	88	91.5	96

The Precision comparison findings of suggested ABOA + DNN, AE + SVM, Naive AE + DNN, and refined AE + DNN classifiers are shown in Fig. 4. According to the results, the proposed ABOA+DNN achieves a high Precision rate of 96% for 25 input data. When comparing the Precision rate between the existing approaches, AE + SVM, Naive AE + DNN, and optimized AE+DNN provide lower rates, demonstrating that the suggested scheme can provide good attack identification outcomes than the previous techniques.

4.2 F-Measure Result Comparison

Table 2 shows the F-measure comparison of suggested ABOA + DNN, AE + SVM, Naive AE + DNN, and refined AE + DNN classifiers.

The F-measure comparison findings of suggested ABOA + DNN, AE + SVM, Naive AE + DNN, and refined AE + DNN classifiers are shown in Fig. 5. According to the results, the proposed ABOA + DNN attains an increased F-measure rate of 91%. Comparing the F-measure rate between the existing approaches, AE+SVM, Naive AE+DNN, and optimized AE+DNN provide lower rates of

73.15%, 84.5%, and 87%, respectively, demonstrating that the suggested scheme can provide good attack identification outcomes than the previous techniques. The justification for this is that the ABOA + DNN network is typically significantly faster to train than the AE + SVM, Naive AE + DNN, and efficient AE + DNN networks. It also includes efficient preprocessing steps, due to which the f-measure value is improved.

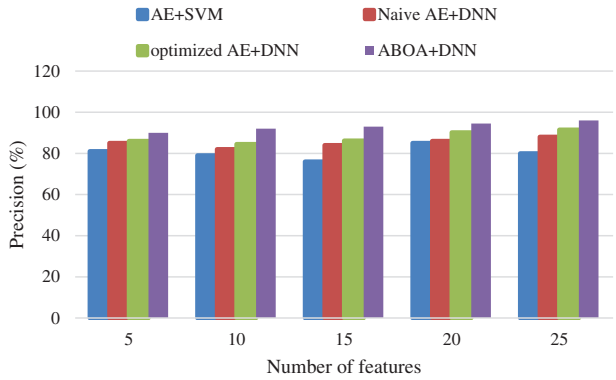

Figure 4: Precision performance comparison

Table 2: F-measure comparison

Input data	AE + SVM	Naive AE + DNN	Optimized AE + DNN	ABOA + DNN
5	66.52	72.5	73	75
10	68.21	74.01	75	78
15	70.12	76	77	81
20	72	79	80	85
25	73.15	84.5	87	91

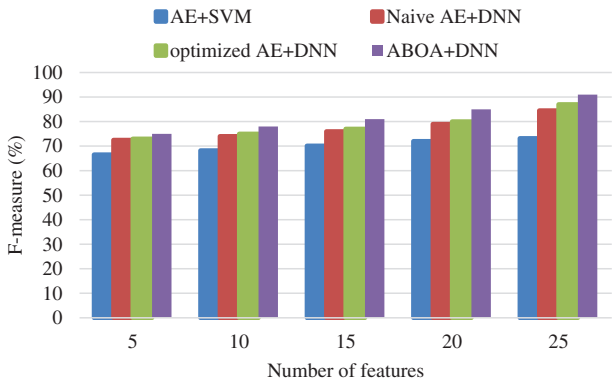

Figure 5: F-measure performance comparison

4.3 Recall Result Comparison

Table 3 shows the Recall comparison of suggested ABOA + DNN, AE + SVM, Naive AE + DNN, and refined AE + DNN classifiers.

Table 3: Recall comparison

Input data	AE + SVM	Naive AE + DNN	Optimized AE + DNN	ABOA + DNN
5	77	80	82.11	85.3
10	78	82	83.5	88.21
15	80	85	86.1	90.11
20	81	88	88.95	93.2
25	84	92	93	96.31

The recall comparison results for suggested ABOA + DNN, A E + SVM, Naive AE + DNN, and enhanced AE + DNN classifiers are shown in Fig. 6. The proposed approach provides a very high recall rate of 96%. According to the results, the suggested ABOA + DNN has a high recall rate value, showing good attack recognition accuracy. When comparing the recall rates of the existing approaches, AE + SVM, Naive AE + DNN, and optimized AE + DNN provide recall rates of 84%, 92%, and 93%, respectively, demonstrating that the suggested scheme can provide good attack recognition outcomes than the previous techniques.

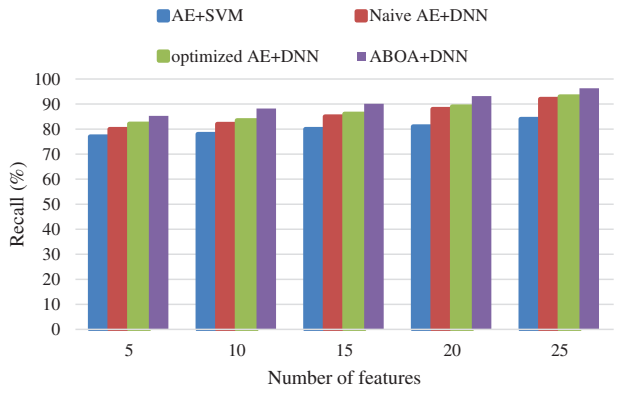

Figure 6: Recall performance comparison

4.4 Accuracy Result Comparison

Table 4 shows the Accuracy comparison of suggested ABOA + DNN, AE + SVM, Naive AE + DNN, and refined AE + DNN classifiers.

The graph in Fig. 7 above illustrates the accuracy comparison for attack detection. Methods such as AE + SVM, Naive AE + DNN, Optimized AE + DNN, and ABOA + DNN multiclass classifiers are used. ABOA + DNN is an excellent method for obtaining accurate predictions, with a high accuracy rate of 99.05%. When comparing the accuracy of previous techniques such as AE + SVM, Naive AE + DNN, and optimized AE + DNN, the rates are as follows: 80%, 95%, and 98%, respectively. ABOA + DNN learning techniques exhibit relative resistance to noise in training data, improving the accuracy when avoiding the local optima problem. Also, compared with other

algorithms, ABOA depicts a better convergence capability when avoiding early convergence; therefore, the recognition rate is increased.

Table 4: Accuracy (%) comparison

Input data	AE + SVM	Naive AE + DNN	Optimized AE +DNN	ABOA+DNN
5	73	81	85	88.21
10	75	87	88	90.12
15	77	88	90	92.21
20	78	92	96	96.55
25	80	95	98	99.05

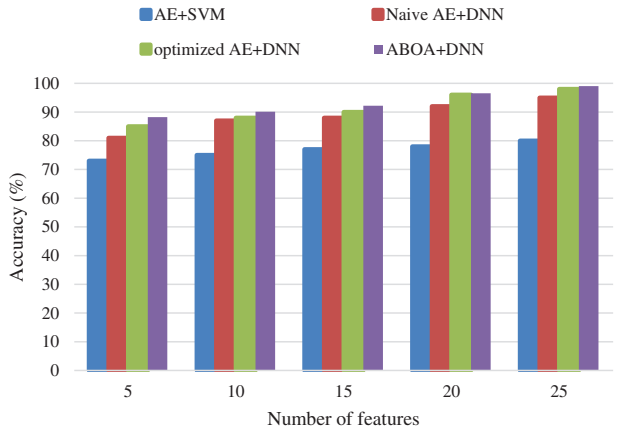

Figure 7: Accuracy performance comparison

5 Conclusions and Future Work

DDoS attack classification using ABOA and DNN architecture is proposed in this work. First, a naive ABOA model is proposed to yield optimal features. DNN classifies the reduced features. To improve DNN, we used ABOA with intelligent learning rate determination and optimization. The suggested mechanism protects cloud infrastructure from unwanted traffic and improves service quality for cloud users by detecting flooding attacks in the private cloud using machine and deep learning. This methodology outperforms previous s DDoS identification approaches on the NSL-KDD dataset with 98.43 percent accuracy. The accuracy of CICIDS2017 is 98.92 percent, comparable to other modern methods. On the NSL-KDD database, the proposed strategy beats all others in precision, recall, and F1-score. It indicated that the proposed unique technique is an effective and original mechanism for supervising DDoS attacks in cloud environments through experiments.

Applying the provided mechanism, the proposed model yielding the highest accuracy is precise and desirable for the diagnosis of DDoS attacks in IoT. In the case of future works, it is recommended that other techniques such as GRU also can yield improved accuracy such as LSTM techniques. Future work includes designing and implementing a scalable deep learning-based DDoS assault recognition system that can be virtualized in a software-defined network controller. The presented approach may be developed to build a novel multimachine recognition scheme in data center networks that acts as a

recovery algorithm. While deep learning may be an efficient DDoS detection solution, existing systems are restricted by their applicability for online applications in resource-constrained contexts.

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

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

References

[1] A. Nayyar, *Handbook of Cloud Computing: Basic to Advance Research on the Concepts and Design of Cloud Computing*. India: BPB Publications, 2019. Available at https://www.google.co.in/books/edition/Handbook_of_Cloud_Computing/ICiwDwAAQBAJ?hl=en&gbpv=1

[2] O. Osanaiye, K. K. R. Choo and M. Dlodlo, "Distributed denial of service (DDoS) resilience in cloud: Review and conceptual cloud DDoS mitigation framework," *Journal of Network and Computer Applications*, vol. 67, no. 1, pp. 147–165, 2016.

[3] R. Bhadauria and S. Sanyal, "Survey on security issues in cloud computing and associated mitigation techniques," arXiv preprint arXiv:1204.0764, vol. 47, pp. 777–780, 2012.

[4] O. A. Osanaiye, "DDoS defence for service availability in cloud computing," Ph.D. Dissertation, University of Cape Town, Cape Town, 2016.

[5] H. T. Elshoush and I. M. Osman, "Alert correlation in collaborative intelligent intrusion detection systems—A survey," *Applied Soft Computing*, vol. 11, no. 7, pp. 4349–4365, 2011.

[6] P. Mishra, V. Varadharajan, U. Tupakula and E. S. Pilli, "A detailed investigation and analysis of using machine learning techniques for intrusion detection," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 1, pp. 686–728, 2018.

[7] N. Agrawal and S. Tapaswi, "Defense schemes for variants of distributed denial-of-service (DDoS) attacks in cloud computing: A survey," *Information Security Journal: A Global Perspective*, vol. 26, no. 2, pp. 61–73, 2017.

[8] Cloud in the Crosshairs NETSCOUT's 14th Annual Worldwide Infrastructure Security Report, Apr. 2020., [Online]. Available: https://www.netscout.com/sites/default/files/2019-03/SECR_005_EN-1901%E2%80%93WISR.pdf

[9] Q. Yan, F. Richard Yu, Q. Gong and J. Li, "Software-defined networking (SDN) and distributed denial of service (DDoS) attacks in cloud computing environments: A survey some research issues and challenges," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 602–622, 2016.

[10] G. Somani, M. S. Gaur, D. Sanghi, M. Conti and R. Buyya, "DDoS attacks in cloud computing: Issues taxonomy and future directions," *Computer Communication*, vol. 107, no. 8, pp. 30–48, 2017.

[11] N. Agrawal and S. Tapaswi, "Defense mechanisms against DDoS attacks in a cloud computing environment: State-of-the-art and research challenges," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 4, pp. 3769–3795, 2019.

[12] A. Bhardwaj, V. Mangat and R. Vig, "Hyperband Tuned deep neural network with well posed stacked sparse AutoEncoder for detection of DDoS attacks in cloud," *IEEE Access*, vol. 8, pp. 181916–181929, 2020.

[13] Z. Chiba, N. Abghour, K. Moussaid, A. El Omri and M. Rida, "Intelligent approach to build a deep neural network-based IDS for cloud environment using combination of machine learning algorithms," *Computer Security*, vol. 86, no. 3, pp. 291–317, 2019.

[14] N. Shone, T. N. Ngoc, V. D. Phai and Q. Shi, "A deep learning approach to network intrusion detection," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 2, no. 1, pp. 41–50, 2018.

[15] S. Bhagat and S. K. Pasupuleti, "Simulated raindrop algorithm to mitigate DDoS attacks in cloud computing," in *Proc. of the Sixth Int. Conf. on Computer and Communication Technology*, India, pp. 412–418, 2015.

[16] K. Bhushan and B. B. Gupta, "Distributed denial of service (DDoS) attack mitigation in software defined network (SDN)-based cloud computing environment," *Journal of Ambient Intelligence and Humanized Computing*, vol. 10, no. 5, pp. 1985–1997, 2019.

[17] N. G. Amma and S. Selvakumar, "Optimization of vector convolutional deep neural network using binary real cumulative incarnation for detection of distributed denial of service attacks," *Neural Computing and Applications*, vol. 34, pp. 2869–2882, 2021.

[18] P. Verma, S. Tapaswi and W. W. Godfrey, "An adaptive threshold-based attribute selection to classify requests under DDoS attack in cloud-based systems," *Arabian Journal for Science and Engineering*, vol. 45, no. 4, pp. 2813–2834, 2020.

[19] O. Osanaiye, H. Cai, K. K. R. Choo, A. Dehghantanha, Z. Xu *et al.,* "Ensemble-based multi-filter feature selection method for DDoS detection in cloud computing," *EURASIP Journal on Wireless Communications and Networking*, vol. 1, no. 1, pp. 1–10, 2016.

[20] G. S. Kushwah and V. Ranga, "Voting extreme learning machine based distributed denial of service attack detection in cloud computing," *Journal of Information Security and Applications*, vol. 53, no. 1, pp. 102532, 2020.

[21] R. SaiSindhuTheja and G. K. Shyam, "An efficient metaheuristic algorithm-based feature selection and Recurrent neural network for DoS attack detection in cloud computing environment," *Applied Soft Computing*, vol. 100, no. 1, pp. 106997, 2021.

[22] S. Arora and S. Singh, "Butterfly algorithm with levy flights for global optimization," in *Int. Conf. on Signal Processing, Computing and Control*, India, pp. 220–224, 2015.

[23] S. Arora and S. Singh, "Butterfly optimization algorithm: A novel approach for global optimization," *Soft Computing*, pp. 1–20, 2018.

[24] Ö. Kasim, "An efficient anfd robust deep learning-based network anomaly detection against distributed denial of service attacks," *Computer Network*, vol. 180, no. 4, pp. 107390, 2020.

[25] Y. Yang, K. Zheng, B. Wu, Y. Yang and X. Wang, "Network intrusion detection based on supervised adversarial variational auto-encoder with regularization," *IEEE Access*, vol. 8, pp. 42169–42184, 2020.