Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 9855022, 14 pages
https://doi.org/10.1155/2022/9855022

Research Article

Autoencoder for Design of Mitigation Model for DDOS Attacks via M-DBNN

Ankit Agrawal,[1] **Rajiv Singh**,[1] **Manju Khari**,[2] **S. Vimal**,[3] **and Sangsoon Lim**[4]

[1]*Department of Computer Science, Banasthali Vidyapith, Banasthali, 304022 Rajasthan, India*
[2]*School of Computer and System Sciences, Jawaharlal Nehru University, New Delhi, India*
[3]*Department of AI & DS, Ramco Institute of Technology, North Venganallur Village, Rajapalayam, 626117 Tamilnadu, India*
[4]*Department of Computer Engineering, Sungkyul University, Anyang 14097, Republic of Korea*

Correspondence should be addressed to Sangsoon Lim; slim@sungkyul.ac.kr

Received 15 January 2022; Revised 3 March 2022; Accepted 7 March 2022; Published 7 April 2022

Academic Editor: Nima Jafari Navimipour

Distributed Denial of Service (DDoS) attacks pose the greatest threat to the continued and efficient operation of the Internet. It can lead to website downtime, lost time and money, disconnection and hosting issues, and website vulnerability. Conventional machine learning methodologies are being harmed by reduced recognition rates and greater false-positive rates due to the emergence of new threats. As a result, high-performance machine learning classifiers with low false-positive rates and high prediction accuracy are required for the DDoS detection system. Here, a deep belief neural network is preferred, upgraded to the modified deep neural network (M-DBNN) to accurately detect DDoS attacks from the network. Enable the database to change a specific format and range, which helps the M-DBNN classifier easily predict the class. An advanced Chimp Optimization Algorithm (ChOA) is used to minimize the error to find the best weight of the M-DBNN classifier; this leads to accurate DDOS attack detection and predict the classes effectively. The proposed method is evaluated for CAIDA "DDoS Attack 2007" dataset. The accuracy of the proposed method is 0.87%, and the outcome is compared with those of other existing methods of deep neural network (DNN), support vector machine (SVM), artificial neural network (ANN), and neural network (NN). The proposed method demonstrates great detection accuracy with a low error.

1. Introduction

Distributed Denial of Service (DDoS) attacks are presently the most common and sophisticated danger to enterprises, and they are becoming progressively difficult to overcome. For example, GitHub became a victim of one of the greatest DDoS attacks ever in 2018 [1]. In February, attacks on eBay, Amazon, and Yahoo were undertaken [2]. In July 2008, a DDoS attack on the Georgian.gov website brought down multiple Georgian servers. The attack on Register.com in January 2001 was carried out by abusing Domain Name System (DNS) servers as reflectors [3]. Previous software protection techniques and intrusion detection systems (IDSs) can recognize and block the attack or intrusion which is happening or already has to a specified extent. However, with the arrival of an era of big data and the resulting problem of uninterrupted huge data flow, previous software protection techniques and IDSs have been faced with new dares. For today's high-speed networks, classic network intrusion detection systems (NIDS) have a high rate of packet loss and a high rate of missed detection [4].

Conventional machine learning algorithms are based on shallow learning. Usually, emphasize feature engineering and selection models, making them unable to properly deal with the challenge of a dynamically growing huge amount of data, resulting in erroneous detecting attacks. Shallow learning is not suited for intelligent systems to predict malevolent behaviour in huge amounts of data in particular [5]. Deep learning has advanced quickly in recent years, overcoming many of the shortcomings of conventional machine learning models and providing a model that is well fitted for IDS [6]. It is commonly utilized in various

applications, including speech recognition, picture recognition, machine translation, and a variety of others. It can analyze vast amounts of data once again. Because deep learning approaches can extract high-dimensional features from original data by numerous nonlinear transformations, and the features collected have a hierarchical structure, there is a movement put on them in the construction of novel intrusion detection algorithms [7].

A comprehensive literature assessment on deep learning approaches for intrusion detection has been published. Some of the deep learning approaches for intrusion detection is DNN-based DDoS detection in software-defined networks [8], and CNN-based IDS against DDoS [9]. In the proposed technique, a Modified Deep Belief Neural Network (M-DBNN) [10], like most classical deep learning systems, uses a multilayer restricted Boltzmann machine (RBM) to recognize the DDoS attack. To make the network better fit the training data and increase efficiency in intrusion detection, optimization is employed to identify a suitable weight of M-DBNN. The M-DBNN classifier predicts whether the class is an attack or not attack from the dataset. Whenever a network is attacked and M-DBNN is detected, the user receives an alert notification. On the other hand, the network is secure, and the dataset can be fed into the autoencoder [11], which compacts the income data into a latent-space format and then restructured this data to give an output. The outcome data of the autoencoder is stored in cloud storage [12].

1.1. Main Contribution. Detect intrusion or DDOS attacks from the network to protect users' personal information. The key contribution of the paper is given below:

(i) A deep learning system based on Modified Deep Belief Neural Network (M-DBNN) is utilized to detect malicious behaviours in cyber environments and classify them based on attack types

(ii) Attributes of the dataset are preprocessed to convert a specified format, making the detection process easy. There are three steps to preprocessing the data: Min-Max normalization, Equal Width Discretization, and Correlation-based Feature Selection

(iii) The preprocessed data are separate for the training and testing process fed in the Modified Deep Belief Neural Network (M-DBNN) classifier, which predicts the input data to either attack or not attack class. The performance of M-DBNN is improved by using the Chimp Optimization Algorithm

(iv) For DDoS is detected, an alert message is forwarded to the user. Otherwise, the data are allowed to be stored in cloud storage via an autoencoder

(v) With the help of a ChOA-based M-DBNN classifier, the CAIDA "DDoS assault 2007" dataset is used to detect DDoS attacks. The proposed technique is very sensitive to detecting DDoS attacks, giving 87% accuracy. In addition, the outcome of the proposed technique is compared with previous techniques of DNN, SVM, ANN, and NN

The rest of the paper contains the following: Section 2 presents the literature review related to detecting DDoS attacks. In Section 3, the methodology and architecture of the proposed part are discussed. Section 4 presents the result and discussion of the proposed outcome and compares the proposed and previous techniques. At last, Section 5 contains the conclusion of the paper.

2. Literature Review

Numerous techniques are introduced to detect DDoS attacks or intrusion in networks. A few techniques are discussed below.

Sahoo et al. [13] suggested that the detection of attack traffic is possible by using the SDN's centralized control feature. Various machine learning (ML) approaches were utilized in the STN sector to prevent anomalous traffic. However, the system had an open question about choosing the right features and precise classifiers to detect the attack. The dimension of feature vectors was reduced using SVM-based KPCA, and different SVM parameters were optimized using GA. An enhanced kernel function (N-RBF) was employed to decrease the noise produced by feature discrepancies. Compared to single SVM, the experimental results demonstrated that the model achieves greater generalization and more accurate classification.

Liu [14] had presented a multiscale convolutional neural network with Grayscale Matrix Feature- (GMF-) based DDoS assault detection. The seven-tuple was distinct from characterizing the flow of network characteristics and translated into a grayscale feature via binary, based on the distinct features of the normal flow and the attack flow in the IP protocol. The convolution kernel of various spatial scales was employed to enhance the precision of feature segmentation and local and global features of the network flow, which were recovered based on the network flow Grayscale Matrix Feature (GMF). A DDoS attack classifier was built using a multiscale convolution neural network.

Tuan et al. [15] had suggested a botnet DDoS attack detection using machine learning approaches. Estimation of the method was based on the UNBS-NB 15 and KDD99 publicity datasets, which were well-known for detecting Botnet DDoS attacks. Various machine learning methods like artificial neural network (ANN), support vector machine (SVM), Naïve Bayes (NB), Unsupervised Learning (USML), and Decision Tree (DT) were used for examining the dataset's False Alarm Rate (FAR), accuracy, specificity, AUC, false positive rate (FPR), Matthews correlation coefficient (MCC), and sensitivity.

Doriguzzi-Corin et al. [16] suggested a LUCID, a lightweight deep learning DDoS detection system that used the features of convolutional neural networks (CNNs) to distinguish between malware and benign traffic flows. LUCID was compared to previous state-of-the-art recognition accuracy while delivering a 40x reduction in processing time when

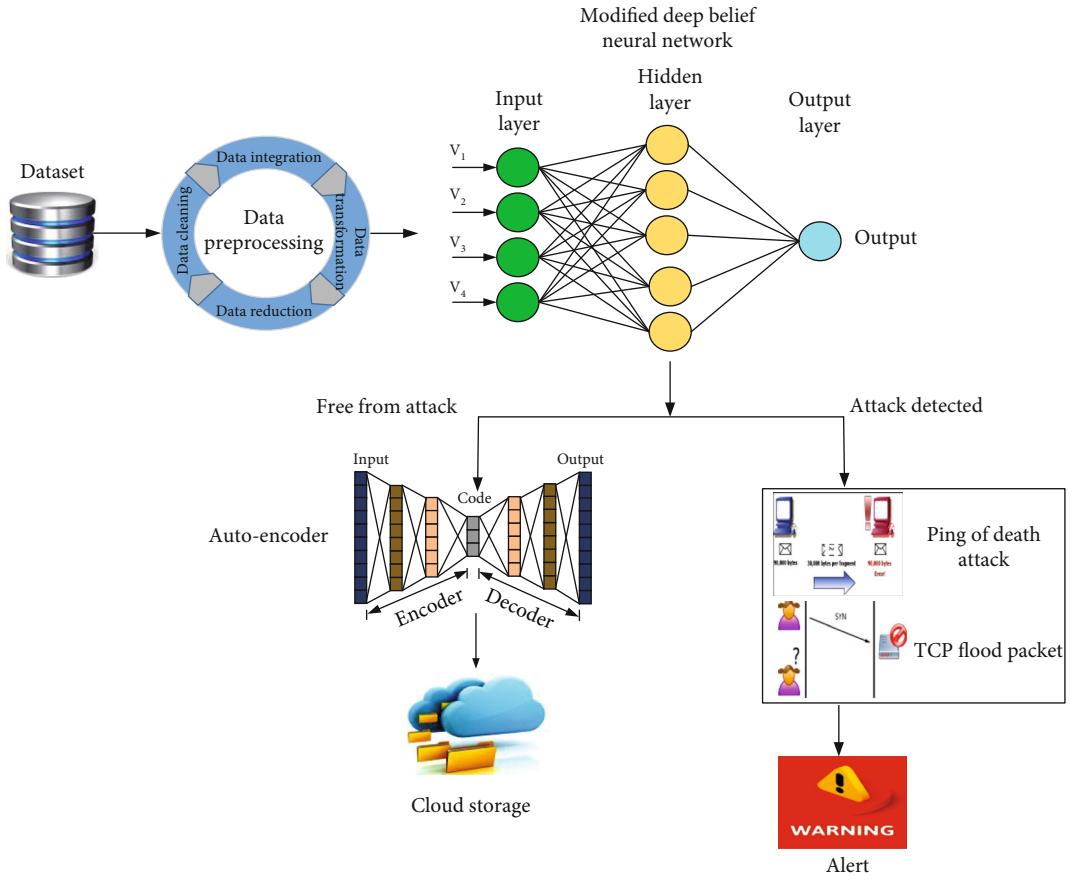


Figure 1: Architecture of planned system.

compared to the state-of-the-art, using the most recent datasets. LUCID's output had the same representation.

Çakmakçı et al. [17] had suggested a multivariate data-friendly online, sequential DDoS detection system. A kernel-based learning method, the Mahalanobis distance, and the chi-square test were all used in the learning algorithm. As detection metrics, extract four entropy-based features and four statistical-based features from network traffic every minute. Then, utilize the entropy features to deploy the kernel-based learning technique to recognize suspicious DDoS input vectors. In every minute, the Mahalanobis distance was calculated among the distribution of dictionary members and suspicious vectors. The Mahalanobis distance was then calculated using the chi-square test. The CICIDS2017 dataset was used to evaluate the DDoS detection strategy.

Dong and Sarem [18] had suggested an SDN technique, and there are two methods for detecting DDoS attacks. To recognize a DDoS assault, one way was to use the severity of the attack, i.e., when a DDoS attack assaulted the SDN controller, four features were examined first. The upgraded *K*-Nearest Neighbor (KNN) algorithm based on machine learning (ML) was used in other techniques to detect the DDoS attack. A novel notion called the degree of attack was introduced and presented for the first time to detect DDoS attacks.

Hussain et al. [19] had suggested an enabled early detection for distributed denial-of-service (DDoS) assaults coordinated via a botnet that controls malevolent devices and used real network data and deep convolutional neural networks (CNNs). To generate a pooled DDoS assault in a cell that can interrupt CPS operations, the puppet device independently performs quiet call, SMS spamming, or a blend of these attacks targeting call, Internet, SMS, signalling, or a combination of these serviced, alternately.

Alzahrani [20] had suggested an attack prevention system of CS-ANN that combines a Cuckoo Search (CS) approach with an artificial neural network (ANN) approach to identify DDoS attacks and allow the server side to be more attentive. As a nature-inspired method, the cloud user features and the attacker features were optimized using CS. These improved features were then sent to the ANN structure. Features that were trained were saved in the database and used at the testing phase to match the test features with the taught features, resulting in results for both attackers and typical cloud users.

In Ref. [13], the authors explained SDN's centralized control feature to detect DDoS attacks. However, the method has some disadvantages by the rapid development of new malware, which would necessitate collecting samples at regular intervals to assess performance. In Ref. [14], the novelist explained a multiscale convolutional neural network

(CNN) to recognize DDoS attacks. But the method has several limitations and assumptions, and the method does not include the packet payload; so the data acquired might be sampled. In Ref. [15], the novelists explained a machine learning approach to detecting botnet DDoS attacks. However, there were scalability limits, and the solution was not ideal. In Ref. [16], the authors describe a method of a LUCID system for DDoS detection. The method is not apposite for online systems due to some flow-level statistics. In Ref. [17], the authors explained a multivariate data-friendly online, sequential DDoS detection system. The method gives high accuracy and detects accurate attacks. Yet the method is not suitable for huge datasets. In Ref. [18], the novelists explained the SDN technique to recognize DDoS attacks. But the solution of the method is inefficient and complex owing to the necessity to discern the packet protocol. In Ref. [19], the novelists explain early detection for distributed denial-of-service (DDoS) via CNN. The outcome is good and also predicts accurate class. Owing to its content-based recognition nature, it is computationally expensive, which is the only drawback of this method. In Ref. [20], the authors explained a CS-ANN-based DDoS attack detection technique. The drawback of the method is that it does not predict an accurate class, so alarm rate and accuracy are less in this approach.

3. Proposed Methodology

DDoS attacks disrupt normal server traffic and overburden the target network or service. Money is lost, fraudulent traffic is generated, reputation is harmed, and clients are harmed when a DDoS attack occurs. Intrusion detection is an important process for system security, as it prevents server attacks and network traffic manipulation. In addition, it improves network data security and determines, detects, and recognizes unapproved program usage, data framework destruction, alteration, and copying. Intrusion or DDoS attacks are detected in the proposed study using a Modified Deep Belief Neural Network (M-DBNN). The dataset is first preprocessed to arrive at a numerical value. The value is then fed into M-DBNN, with numerous invisible, hidden layers, a visible input layer, and a visible output layer. Although the range of the input layer is [0, 1], the values of the buried layer are binary numbers. The modified DBNN classifies the input dataset to determine whether or not it has been subjected to a DDoS attack. The advanced algorithm detects and alerts the user if an attack is detected. Data is supplied to an autoencoder, an encoder, and a decoder after the dataset is free of any attacks or difficulties. The modified DBNN classifies the input dataset to assess whether it has been subjected to a DDoS attack. The advanced algorithm detects and warns the user if an attack is detected. When the dataset is clear of attack challenges, it is sent to the autoencoder, with both an encoder and a decoder. The architecture of the proposed system is given in Figure 1, and pseudocode of the overall system is given in Pseudocode 1.

Pseudocode

```
Input raw dataset = X
{
For all data in dataset
    # Pre-processing
    Re-range the dataset using Eq. (1)
    Convert numerical value using Eq. (2)
    Redundancy using Eq. (3)
    Pre-processing data ⟹ Pre-data
    # M-DBNN
      Data splitting
          {
            Training data
            Testing data
            Actual class
          }
    # Training ChOA optimizer
    For all in input dataset
      {
        Initialization = weight using Eq. (8)
        Fitness function using Eq. (9)
          Update the solution using Eq. (10)
        Best solution
      }
    #Testing the dataset
}
End
Outcome:
Predict the class to detect the server is normal or attack
```

Pseudocode 1: Pseudocode of overall system.

3.1. Preprocessing. The data is not in a specific range, so the preprocessing technique is used to fit the dataset in an exact range. Several approaches are available for the normalization of the dataset. Among them, Min-Max normalization is more efficient. The normalization approach converts a value from σ to σ^* that is suitable in the range of [A, B]. Its mathematical expression is given below.

$$\sigma^* = \frac{\sigma - \sigma_{\min}}{\sigma_{\max} - \sigma_{\min}}. \tag{1}$$

The data is normalized to give an equal weight of all attributes. After the normalizing process, discretization takes place. The discretization process transfers the domain of the continuous feature to an insignificant domain in the form of a finite number. Equal Width Discretization (EWD) is preferred to transfer the continuous field to a finite number that creates several bins. Moreover, this approach improves the performance in large datasets, where K is used for evaluating the bin quantity. Every bin is allied in a distinct, discrete value. EWD split the number lines among $V_{\max}$ and $V_{\min}$ at the interval of k. Generally, the value of k is set as 10. It is known as a user predefined variable. Interval's width is evaluated by

$$W = \frac{B - A}{N}, \tag{2}$$

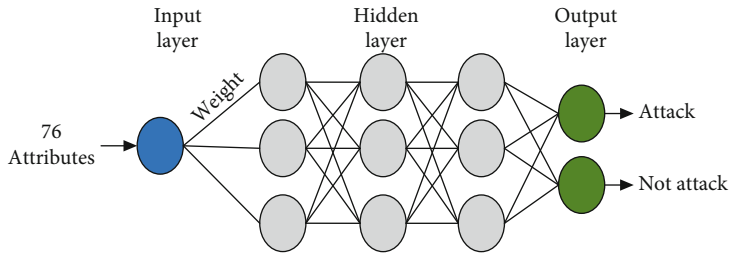


Figure 2: Schematic structure of M-DBNN.

where N denotes the interval and B and A are the highest and lowest values of attributes.

Finally, select the feature. In this process, irrelevant and redundant features present in the dataset are eliminated, thus enhancing the accuracy and decreasing the running cost. Here, Correlation-based Feature Selection (CFS) is used. The evaluation role favours subsets with attributes that are substantially correlated with the class but uncorrelated with one another. The feature of irrelevance in the dataset is needed to eliminate since it links to the class. In addition, the fired feature was also eliminated because it was highly linked with the leftover feature. The evaluation of subset in CFS is given below:

$$\text{Merit}_s = \frac{k\overline{\text{rcf}}}{\sqrt{k + k(k-1)r_{\overline{\text{ff}}}}}. \tag{3}$$

MS is the average feature-feature inter-correlation, $\overline{\text{rcf}}$ is the mean feature-class correlation, and $r_{\overline{\text{ff}}}$ is the average feature-feature intercorrelation of a feature subset containing k features.

3.2. Modified Deep Belief Neural Network (M-DBNN). Deep belief neural network is a growing approach. DBNN contains a layer by layer arrangement of restricted Boltzmann machine (RBM) [21]. RBM trains the data in greedy layer-wise to accomplish the unsupervised filed solid execution. The significant usage of RBMs is likely to be because there is a lack of labelled data, and RBMs and autoencoders can be pre-trained on unlabeled data and fine-tuned on a small portion labelled data. Greedy layer-wise techniques are utilized for training the data of DBN each layer at each time. The schematic structure of M-DBNN is shown in Figure 2.

Moreover, the method optimizes a layer with a greedy time. Next, the completion of unsupervised training supervised training technique is the whole access layer; the name of this process is fine-tune stage. This stage is a combination of two notions: (a) an initial factor presents an effect of major standardizing, and (b) an input distribution study will help the study of input to outcome mapping. For the process of pretraining, the dataset unsupervised is trained by a greedy layer-wise approach. And the supervised layer is trained by the fine-tuning stage from the first layer to improve the labelled sample feature.

Table 1: Parameters of M-DBNN classifier.

Parameters	Ranges
Number of iteration	500
Batch size	14109
Step ratio	0.1
Dropout rate	0.9

For RBM, h and v represent hidden and visible layers, respectively. Here, three variables are needed for system decision, namely,

$$\theta = \{W, A, B\}. \tag{4}$$

Let $A = \{a_i \epsilon R^m\}$ and $A = \{b_j \epsilon R^{n\cdot}\}$.

Here, A denotes an element of visible layer, W denotes weight matrix, and B denotes an element of the hidden layer. a_i represent the i^{th} visible layer threshold, and b_j represent j^{th} hidden layer threshold.

The layer of hidden and visible follows Bernoulli distribution; RBM energy equation is stated as

$$E(v, h \setminus \theta) = -\sum_{i=1}^{n} a_i v_i - \sum_{j=1}^{m} b_j h_j - \sum_{i=1}^{n}\sum_{j=1}^{m} v_i W_{ij} h_j, \tag{5}$$

where $\theta = W_{ij}, a_i, b_i$. The energy function displayed the energy value for separately visible node and hidden layer node valuation. The overall hidden layer node set is expressed as

$$P(v \setminus \theta) = \frac{1}{z(\theta)a}\sum_{h} e^{-E(v,h\setminus\theta)}, \tag{6}$$

where $z(\theta)$ is a standardized factor.

The RBM is linked in a way biphasic. The possibility of the beginning of neurons in the visible layer and hidden layer is expressed as

$$\begin{aligned} P(h_j = 1|v) &= \sigma\left(b_j + \sum_i v_i W_{ij}\right), \\ P(v_i = 1|h) &= \sigma\left(a_i + \sum_j h_j W_{ij}\right). \end{aligned} \tag{7}$$

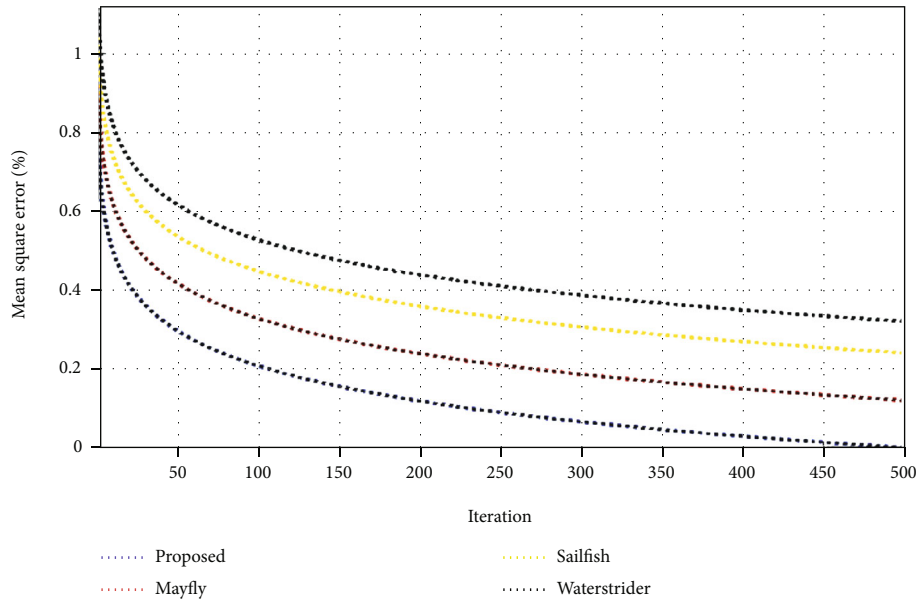


Figure 3: Comparison of mean square error in proposed approach and existing approaches.

3.2.1. Training Data. Chimp Optimization Algorithm (ChOA) is used for weight optimization. Weight is defined as the connection between the hidden and visible layers. A suitable weight is used to enhance the performance of the system. ChOA is a group hunting optimization that is a lifestyle of chimpanzees. Four entitled chimps are a group to attack the prey. They are drivers, barriers, chasers, and attackers. The driver follows the prey, while the barrier blocks the prey to move in a different direction, and the chaser moves suddenly to catch the prey. Finally, the attacker attacks the prey.

Step 1. Initialize the weight as an input,

$$\text{Weight} = \{W_1, W_2, \cdots, W_n\}. \tag{8}$$

Step 2. Find the fitness function. Here, the error between the real and given outcomes is evaluated to find the fitness function.

$$\frac{\partial E}{\partial w_{ij}^{t-p}} = \frac{\partial E}{\partial y_j} \cdot \left[1 - y_j\right] \cdot y_i^{t-p}, \tag{9}$$

where y_i^{t-p} is the output of input unit i, w_{ij}^{t-p} is the weight between input unit i and hidden unit j at time $t-p$, $\partial E/\partial y_j$ can be calculated by standard BP algorithm, and y_j is the output of hidden unit j.

Step 3. Update the outcome to find the best solution. The best solution is not to find, but to repeat the process as Step 2.

$$\begin{aligned} X_{\text{chimp}}(t+1) &= \begin{cases} X_{\text{prey}}(t) - a \cdot d, & \text{if } \mu < 0.5, \\ \text{chaotic}_{\text{value}}, & \text{if } \mu \geq 0.5, \end{cases} \\ a &= 2fr_1 - f, \\ d &= \left|cX_{\text{prey}}(t) - mX_{\text{chimp}}(t)\right|, \\ c &= 2r_2, m = \text{chaotic}_{\text{value}}, \end{aligned} \tag{10}$$

where t indicates the number of the current iteration, X_{chimp} is the position vector of a chimp, X_{prey} is the vector of prey position, and a, m, c represent the coefficient vector. Through the iteration process, f is lowered nonlinearly from 2.5 to 0, m a chaotic vector intended based on several chaotic maps, and r_1 and r_2 are random vectors in the range of [0, 1].

Step 4. The last step is to stop the process. When the best solution is captured, the process goes to an end.

3.2.2. Testing Data. In the proposed work, only 80% dataset is trained; the remaining 20% of the dataset are tested. The test data are correctly predicted for being free from either attack class or attack class. When the attack is recognized, a notification is generated to alert the user.

3.3. Autoencoder. The term autoencoder (AE) is defined as an artificial network system that contains both a decoder and an encoder. The autoencoder compresses the data and encodes the input data and reconstructs the data as near to the original data [22]. The part encoder minimizes the dimensionality of the given data; besides, the decoder decompresses stage to reduce the noise of the given data and create the new data. The key objective of the autoencoder is to reduce the error during reconstruction among input data and reconstructed data. The loss of reconstruction is expressed as L_p distance.

$$\mathscr{L}_{AE}(\theta, \phi) = \min E(x, \hat{x}) = \min \left\|x - \hat{x}\right\|_p, \tag{11}$$

where x denotes data of input and $\hat{x}$ represents the data of reconstructed. In other words, the loss of reconstruction is also defined as entropy.

$$\mathscr{L}_{\text{AE}}(\theta, \phi) = -\sum_{i=1}^{D} \hat{x}_i \log(x_i), \tag{12}$$

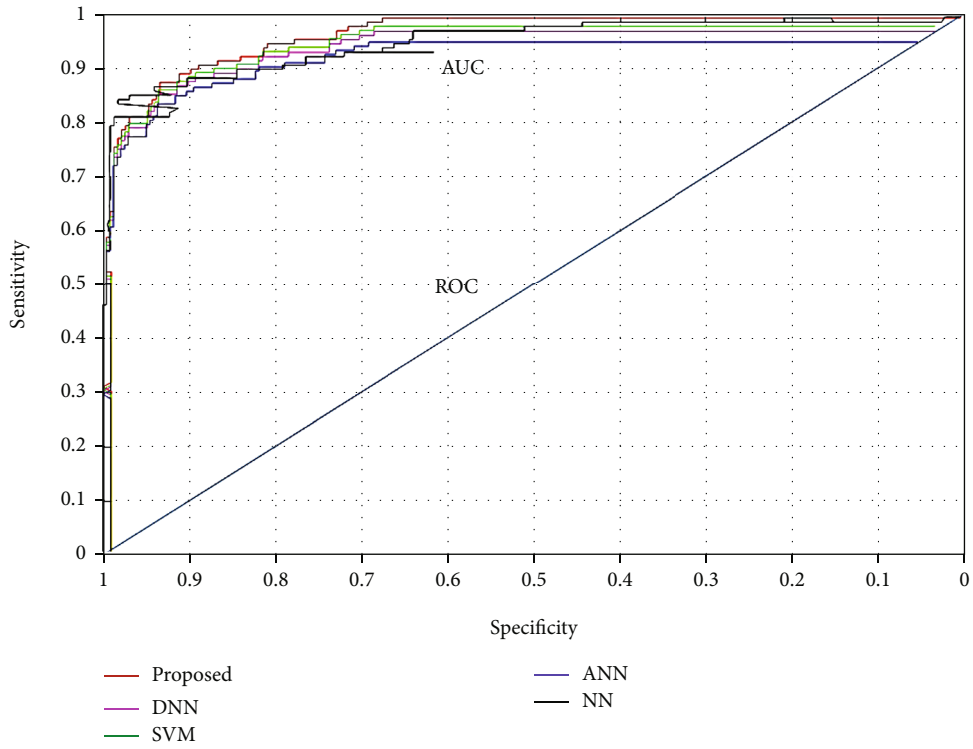


Figure 4: Comparison of AUC and ROC curve.

where D is the raw data x dimensionality. The encoder $q_\phi(z \mid x)$ is the next function of the probability that is utilized to estimate inflexible function $p_\theta(z \mid x)$. Let us assume $p_\theta(z)$ is Gaussian of multivariate in the covariance matrix of diagonal. Sample point z is randomly selected from $p_\theta(z)$. The function of a decoder is $p_\theta(z \mid x)$ which is a mapping from the latent space to the raw sample.

3.4. Cloud Storage. Cloud storage is amorphous, with no clear range of abilities or infrastructure. Many conventional hosted or managed service providers (MSP) provide block or file storage in addition to standard remote management protocols or virtual or physical server hosting, so there are plenty of options. Other options have emerged, such as the Amazon S3 service, which mimics flat databases built to store huge things. Some advantages of cloud storage are cost-effectiveness, disaster preparedness, ease of management, and lower influence outages and upgrades [23].

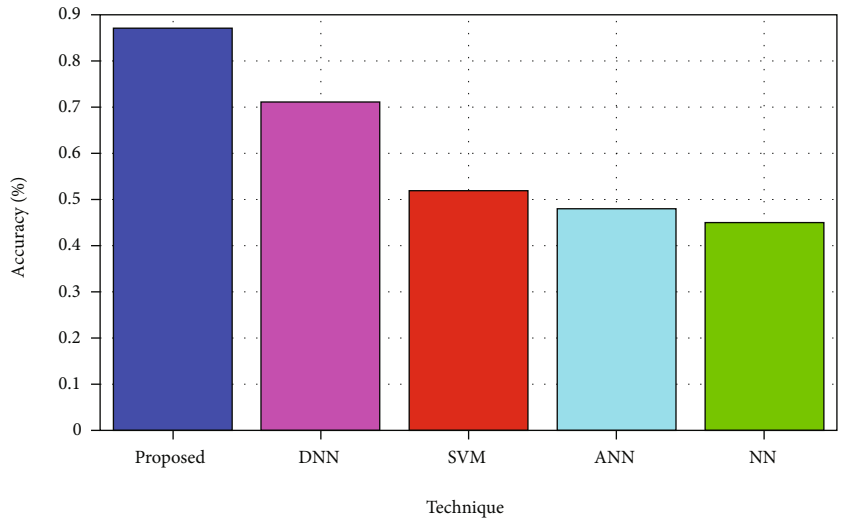
4. Result and Discussion

The ChOA-based M-DBNN is developed to predict the DDoS attack in a network environment. The proposed ChOA-based DBN is implemented and evaluated using MATLAB 2020b in Windows x64 bit platform with Intel Core i5 processor and 8 GB Ram. CAIDA "DDoS Attack 2007" [24] is considered the benchmark dataset to analyze the proposed techniques. The proposed technique includes three phases such as preprocessing, M-DBNN training, and testing.

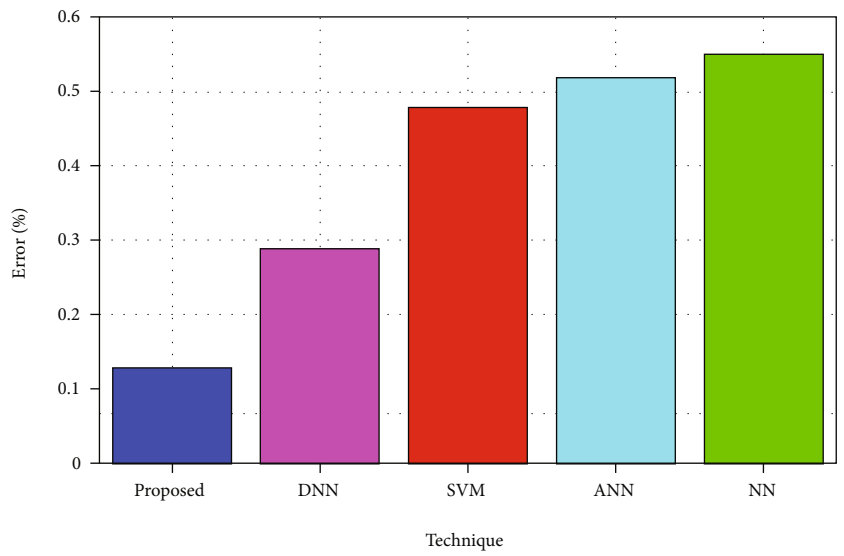
Table 2: Confusion matrix of the proposed and existing techniques.

Techniques	Actual class	Predicted class	
		True	False
M-DBNN	Positive	37008	6228
	Negative	62300	7336
SVM	Positive	50074	4407
	Negative	9352	49039
ANN	Positive	32539	37097
	Negative	17356	25880
NN	Positive	32548	25843
	Negative	17399	37084
DNN	Positive	10864	7587
	Negative	18683	3986

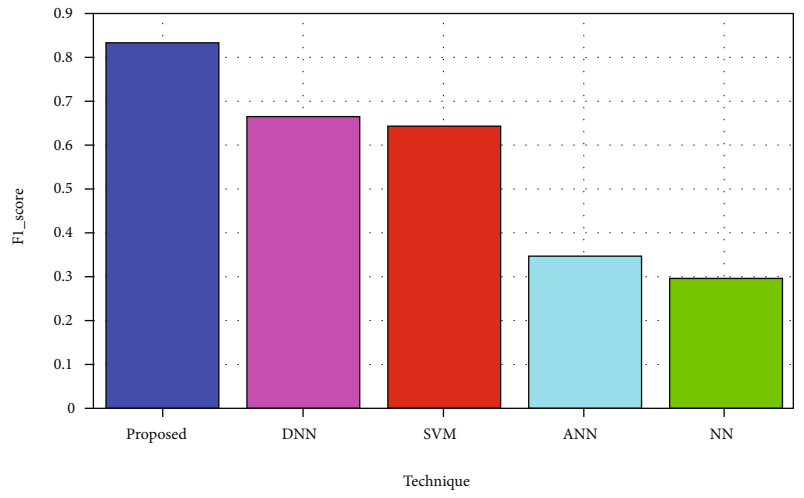
First, the database is taken in *pcap* file format; then extract the *pcap* file to convert to *csv* file format. The database contains 225746 models and 79 attributes, including IP, source IP, source port, protocol type, and destination port. The attributes of the database were preprocessed in which the undesirable variables were removed from the database. There are three steps to preprocessing a database: normalization, discretization, and feature selection. The dataset is first measured to fit within a given range using the Mini-Max normalization. All properties are given equal weight in the data, which is normalized. After that, equal width discrimination occurs, converting successive domain



(a)

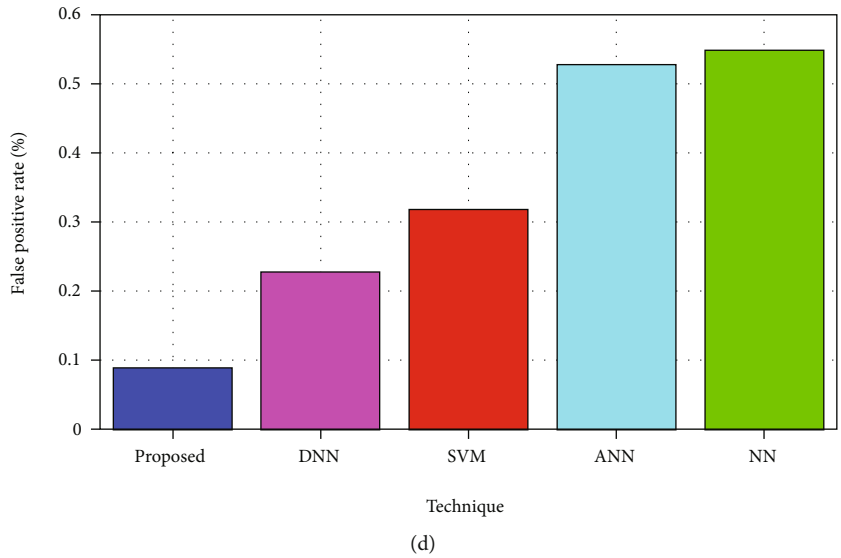


(b)



(c)

Figure 5: Continued.



(d)

Figure 5: Comparison of proposed and existing approaches. (a) Accuracy. (b) Error. (c) F1_score. (d) False positive rate.

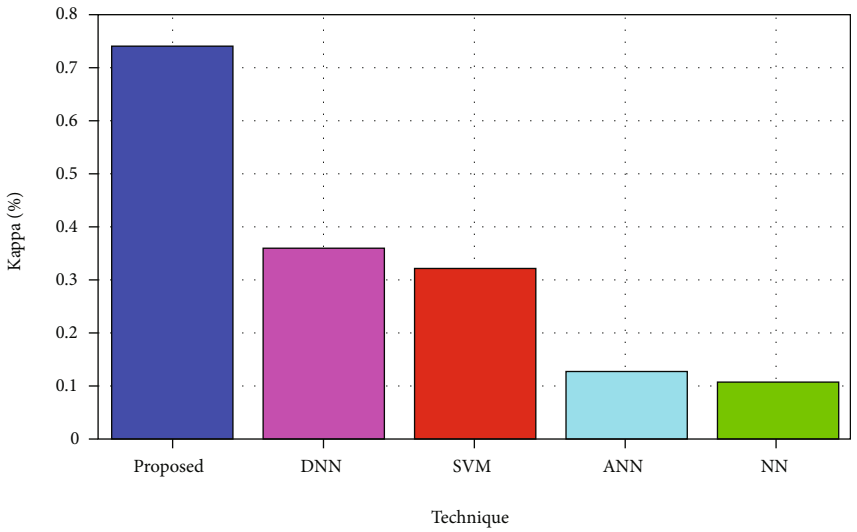
features to normal domain features with a limited number. It leads to improved performance on larger databases. The final stage is feature selection, an important preprocessing stage in network traffic classification. Here, correlation-based feature selection is used to remove unnecessary and irrelevant information from the database to improve accuracy and reduce computation costs. Among these 225745 samples, only 80% dataset is trained; i.e., 180596 samples are trained, and the remaining 20% dataset is tested; i.e. 45149 samples are tested. These samples are predicted for either two classes like DDoS attack or not attack

For M-DBNN, the weight parameters are fine-tuned using a supervised backpropagation algorithm, and the tuning step employs supervised learning to fine tune the network structure's weight. The term weight is defined as the interconnection of the input layer and hidden layers. Here, the weight of the M-DBNN is selected for the Chimp Optimization Algorithm. The Chimp Optimization Algorithm (ChOA) is a new metamorphic that has just been introduced. Surprisingly, this reflects the chimp's social status relationships and hunting activity. Using the Chimp Optimization Algorithm (ChOA) to improve the appropriate weight system to reduce error, the M-DBNN trains the input of 79 attributes to detect an intrusion or DTOS attacks. Table 1 contains the parameters of the M-DBNN classifier. The optimal problem is solved up to 500 iteration levels in the proposed and existing techniques. The batch size refers to the sample used to train a model before modifying its trainable model variables, weights, and biases. Here, 14109 batch sizes are used in the M-DBNN for attack detection.

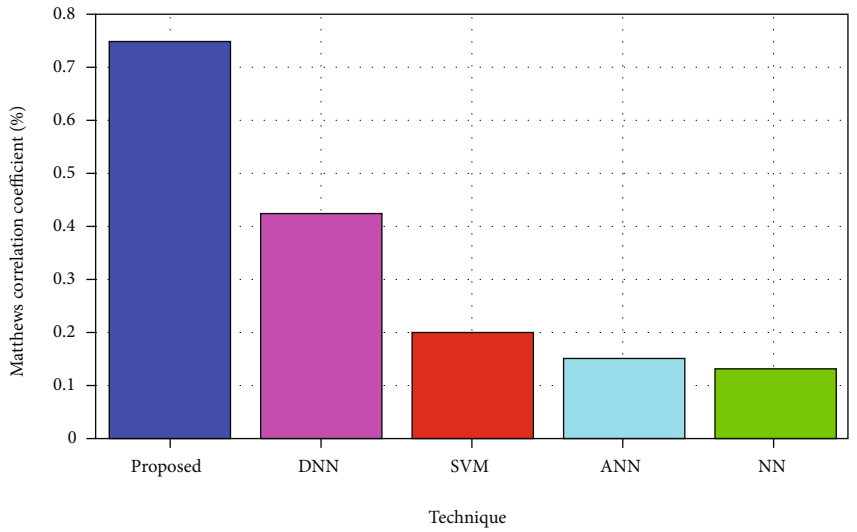
The fitness function of the created chimps is measured using mean square error (MSE) which is dependent on measuring the difference between both the intended and observed values by the generated search agents for all of the training sets. Comparison of mean square error versus iteration graphical model of the proposed and existing approach like mayfly, sailfish, and water strider is sketched in Figure 3. Among these four approaches, the proposed ChOA give the best outcome; that is, in the first iteration, the value of the mean square is 1%; then the error is reduced to reach 0% at the 500^{th} iteration. Mayfly optimization starts from a 1% error on 1^{st} iteration, which reached a 0.13% error on the 500th iteration. Another Sailfish optimization starts with a 1% error on the 1^{st} iteration, and it reduced the error to reach 0.22% on the 500^{th} iteration. The final approach is to reduce the error value to 1% on the 1^{st} iteration of the water striker, reducing the error to 0.33% on the 500^{th} repeat. This graphical model demonstrates that the proposed ChOA gives 0% error, so the computation time of the approach is reduced.

Area Under the Curve- (AUC-) Receiver Operating Characteristic (ROC) curves seem to be a performance statistic for classification issues at variable threshold levels, which is shown in Figure 4. AUC represents the measure or degree of separation, whereas ROC is a probability curve. Generally, the curve of AUC is reached at 1; the performance of the system's measure is good. But the curve is in 0 value, the system's measure is very poor. The curve AUC gradually reaches 1 in 0.68 specifications in the proposed techniques. Calculating and displaying the true positive ratio instead of the false-positive ratio for classification at different thresholds give the ROC curve. Low training capability, sophisticated network attack detection, and local optimization are all M-DBNN problems. The optimal weights and dependencies of the M-DBNN classifier are determined by the ChOA algorithm.

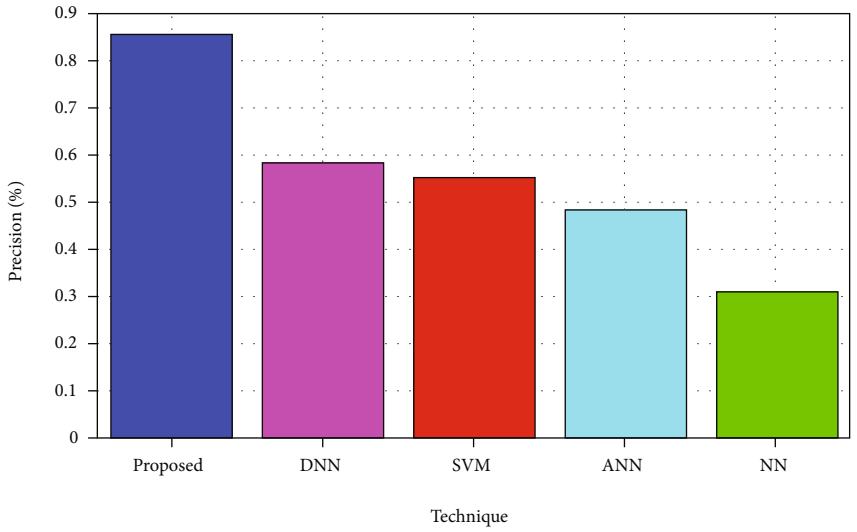
A confusion matrix and performance measures like accuracy, detection rate, and are used to produce analysis assessment. The projected class is represented by each column in the confusion matrix, whereas the actual class is represented by each row. For two or more class matrices, a confusion matrix can be employed. The variables True, False, Negative, and Positive are formulated as follows: True Negative—the number of negative instances is identified



(a)

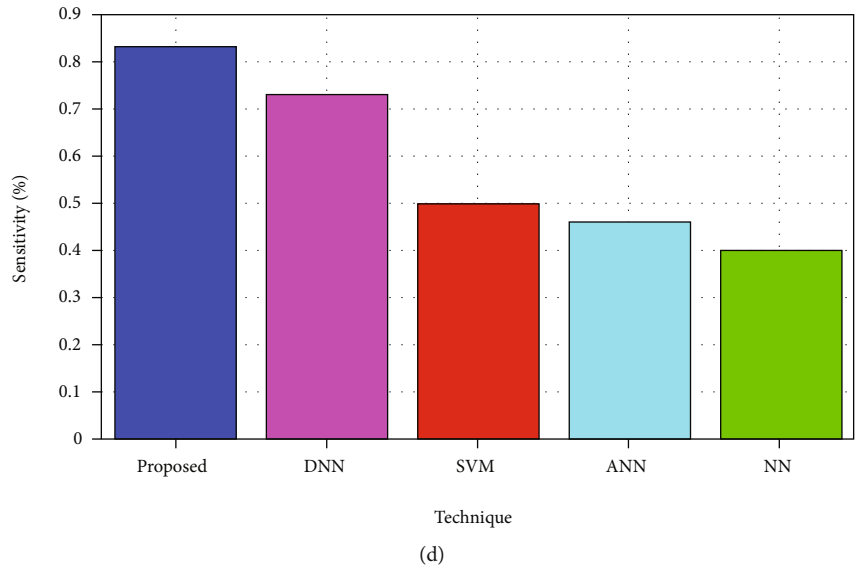


(b)

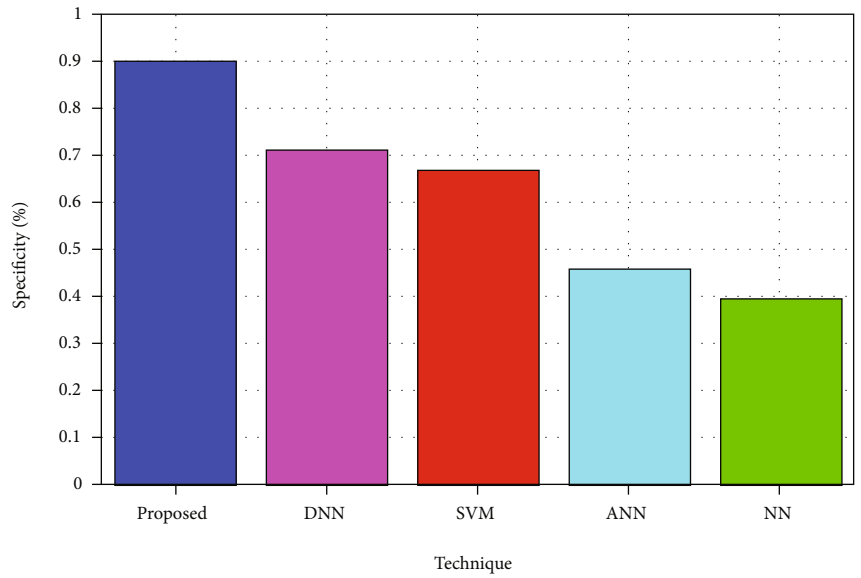


(c)

Figure 6: Continued.



(d)



(e)

Figure 6: Comparison of proposed and existing approaches. (a) Kappa. (b) The Matthews correlation coefficient. (c) Precision. (d) Sensitivity. (e) Specificity.

correctly as ordinary; True Positive—the total number of positive instances is correctly categorized as an attack; False Positive—the number of nonattacking incidents has been incorrectly labelled as attacks; and False Negative—the number of assault instances is labelled as normal when they are not. Table 2 contains the confusion matrix of proposed and existing techniques.

The accuracy of a statistic refers to how close it is to the actual true value. This is significant since inaccuracies in results might be caused by faulty equipment, human error, or inadequate data processing. The accuracy of the proposed and existing approaches is sketched in Figure 5(a). The proposed M-DBNN provides 0.87% accuracy, while the existing DNN, SVM, ANN, and NN techniques give accuracies of 0.71%, 0.52%, 0.48%, and 0.45%, respectively. This is clearly demonstrated, and the proposed approach is more favourable than existing techniques. Similarly, the error is analyzed for the proposed and existing approaches that are shown in Figure 5(b). The discrepancy between the calculated value and the actual value is referred to as "error" in statistical analysis. The error of the proposed techniques is 0.13%, and the current techniques of DNN, SVM, ANN, and NN are 0.29% error, 0.48% error, 0.52% error, and 0.55% error, respectively. Then, the F1_score is analyzed and sketched in Figure 5(c). The F1 score is a machine learning metric that was used to calculate the system's binary categories and quantify the dataset's accuracy. The precision and recall models have a well-defined harmonic mean. The value of F1_score in the proposed approach is 0.84. And the values of F1_score of DNN, SVM, ANN, and NN are 0.67, 0.65, 0.35, and 0.30, respectively. Then, false positive rate is analyzed. That is, the likelihood of rejection of the null

Table 3: Overall performance of proposed and existing approaches.

Performance	Proposed M-DBNN	DNN	SVM	ANN	NN
Accuracy	0.87	0.71	0.52	0.48	0.45
Error	0.13	0.29	0.48	0.52	0.55
F1_score	0.84	0.67	0.65	0.35	0.30
False positive rate	0.09	0.28	0.32	0.53	0.55
Kappa	0.74	0.36	0.32	0.13	0.11
Matthews correlation coefficient	0.74	0.42	0.20	0.15	0.12
Precision	0.85	0.58	0.55	0.48	0.31
Sensitivity	0.83	0.73	0.50	0.45	0.40
Specificity	0.90	0.71	0.67	0.46	0.40

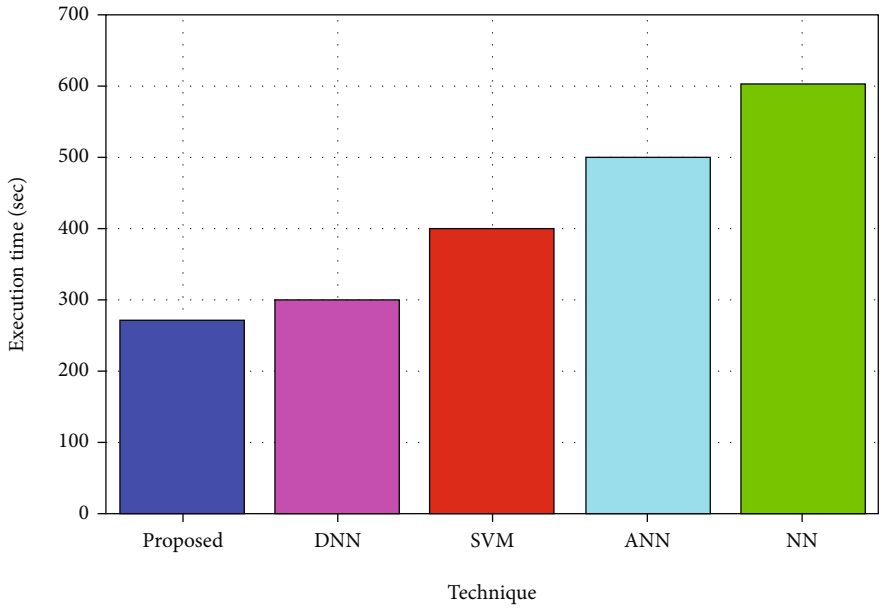


Figure 7: Comparison of proposed and existing techniques execution time.

hypothesis incorrectly for a test item is known as the false-positive ratio. The relationship between the number of negative events incorrectly classified as positive and the total number of true negative events is used to compute the false positive rate. The false positive rates of the proposed approach, DNN, SVM, ANN, and NN are 0.09%, 0.28%, 0.32, 0.53%, and 0.55%, respectively.

The statistical analysis of kappa is described as comparing the observed values of a training dataset to the expected value. In the proposed techniques, the value of kappa is 0.74%, 0.36% in DNN techniques, 0.32% in SVM techniques, 0.13% in ANN techniques, and 0.11% in NN techniques. The Matthews correlation coefficient is then analyzed, and in machine learning, the Matthews correlation coefficient (MCC) is used to estimate the validity of two binary classifications. The Matthews correlation coefficient value of the planned techniques is 0.74%, but those of the existing DNN, SVM, ANN, and NN techniques are 0.42%, 0.20%, 0.15%, and 0.12%, respectively. Next, precision is analyzed, which is presented in Figure 6(c). Precision is a labelling strategy for the positive dataset. The split of true positives into the addition of genuine positives and false positives yields precision. The value of precision in proposed techniques is 0.85%, and those of the existing techniques of DNN, SVM, ANN, and NN contain 0.58%, 0.55%, 0.48%, and 0.31%, respectively. When compared to those of existing techniques, the precision values of the projected techniques is extremely high.

Moreover, the sensitivity of the proposed and existing techniques is analyzed and sketched in Figure 6(d). Sensitivity is critical for sensing network traffic since it precisely measures the positive dataset. The proposed technique is more sensitive, and it is utilized to detect minute variations in the network of a server. In comparison to the proposed techniques, the previous techniques are less sensitive. The sensitivity of the proposed techniques is 0.83%, and the sensitivity values of existing techniques such as DN, SVM, ANN, and NN are 0.73%, 0.50%, 0.45%, and 0.40%, respectively. Specificity is one of the statistical methods for correctly recognizing a negative dataset. Compared to the

current techniques, the planned method identifies the negative energy as 0.90%, which is very positive. Previous techniques of DNN, SVM, ANN, and NN contain 0.71%, 0.67%, 0.46%, and 0.40%, respectively. Table 3 contains the overall performance of the proposed and present approaches.

The suggested and existing approaches' execution times were also examined. The operating time of the approach refers to the time it takes to detect an intrusion or attack on the network. The proposed techniques detect the attack within 270 seconds. The current approaches DNN, SVM, ANN, and NN are 300 seconds, 400 seconds, 500 seconds, and 600 seconds, respectively, as described in Figure 7.

Experimental results have shown that the optimal configuration of M-DBNN techniques can be detected, which increases feature exposure performance, resulting in a much larger detection rate and shorter detection time than the system without reducing the dimension. As a result, the recommended ChOA-based M-DBNN classifier is best suited for DDoS detection and should be used in real-time applications.

5. Conclusion

An enhanced ChOA based M-DBNN is utilized to recognize DDoS attacks in the network. The main aim of the proposed technique is to extract representative features from the CAIDA "DDoS Attack 2007" dataset, reduce classification error, and correctly detect DDoS attacks. The preprocessing of the dataset is to remove redundant and irrelevant features from the dataset and to convert it to a finite number of values. The predata are fed to the M-DBNN to predict the class. An enhancement of M-DBNN is used for ChOA optimization, which reduces the error to select the best weight that is interconnected between the input and hidden layer, so the advanced M-DBNN very accurately predicts the classes. The proposed ChOA based M-DBNN gives high diagnostic accuracy which was demonstrated. Many performance measures of the proposed technique show a significant improvement when compared to the previous technique. The suggested technique had a detection accuracy of 87% and a false positive rate of 0.09% on the CAIDA "DDoS Attack 2007" dataset. In the future, advanced hybrid optimization will be used to predict the class more effectively. In addition, high-speed machine learning is used to detect anomalies in the network.

Data Availability

The dataset used to support the findings of this study is available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea Government (MSIT) (No. 2021R1F1A1063319).

References

[1] S. Haider, A. Akhunzada, I. Mustafa et al., "A deep CNN ensemble framework for efficient DDoS attack detection in software defined networks," *IEEE Access*, vol. 8, pp. 53972–53983, 2020.

[2] R. R. Brooks, L. Yu, J. Oakley, and N. Tusing, "Distributed Denial of Service (DDoS): a history," *IEEE Annals of the History of Computing*, p. 1, 2021.

[3] K. Kumar and S. Behal, "Distributed denial of service attack detection using deep learning approaches," in *2021 8th International Conference on Computing for Sustainable Global Development (INDIACom)*, New Delhi, India, 2021.

[4] H. Zhang, Y. Li, Z. Lv, A. K. Sangaiah, and T. Huang, "A real-time and ubiquitous network attack detection based on deep belief network and support vector machine," *IEEE/CAA Journal of Automatica Sinica*, vol. 7, no. 3, pp. 790–799, 2020.

[5] A. A. Diro and N. Chilamkurti, "Distributed attack detection scheme using deep learning approach for Internet of Things," *Future Generation Computer Systems*, vol. 82, pp. 761–768, 2018.

[6] A. Manimaran, D. Chandramohan, S. G. Shrinivas, and N. Arulkumar, "A comprehensive novel model for network speech anomaly detection system using deep learning approach," *International Journal of Speech Technology*, vol. 23, no. 2, pp. 305–313, 2020.

[7] Q. Tian, D. Han, K. C. Li, X. Liu, L. Duan, and A. Castiglione, "An intrusion detection approach based on improved deep belief network," *Applied Intelligence*, vol. 50, no. 10, pp. 3162–3178, 2020.

[8] A. Makuvaza, D. S. Jat, and A. M. Gamundani, "Deep neural network (DNN) solution for real-time detection of distributed denial of service (DDoS) attacks in software defined networks (SDNs)," *SN Computer Science*, vol. 2, no. 2, pp. 1–10, 2021.

[9] J. Kim, J. Kim, H. Kim, M. Shim, and E. Choi, "CNN-based network intrusion detection against denial-of-service attacks," *Electronics*, vol. 9, no. 6, p. 916, 2020.

[10] I. Sohn, "Deep belief network based intrusion detection techniques: a survey," vol. 167, Article ID 114170, Expert Systems with Applications, 2020.

[11] K. Yang, J. Zhang, Y. Xu, and J. Chao, "Ddos attacks detection with autoencoder," in *NOMS 2020-2020 IEEE/IFIP network operations and management symposium*, pp. 1–9, Budapest, Hungary, 2020.

[12] P. Yang, N. Xiong, and J. Ren, "Data security and privacy protection for cloud storage: a survey," *IEEE Access*, vol. 8, pp. 131723–131740, 2020.

[13] K. S. Sahoo, B. K. Tripathy, K. Naik et al., "An evolutionary SVM model for DDOS attack detection in software defined networks," *IEEE Access*, vol. 8, pp. 132502–132513, 2020.

[14] Y. Liu, "Ddos attack detection via multiscale convolutional neural network," *Computers, Materials & Continua*, vol. 62, no. 3, pp. 1317–1333, 2020.

[15] T. A. Tuan, H. V. Long, R. Kumar, I. Priyadarshini, and N. T. K. Son, "Performance evaluation of botnet DDoS

attack detection using machine learning," *Evolutionary Intelligence*, vol. 13, no. 2, pp. 283–294, 2020.

[16] R. Doriguzzi-Corin, S. Millar, S. Scott-Hayward, J. Martinez-del-Rincon, and D. Siracusa, "LUCID: a practical, lightweight deep learning solution for DDoS attack detection," *IEEE Transactions on Network and Service Management*, vol. 17, no. 2, pp. 876–889, 2020.

[17] S. D. Çakmakçı, T. Kemmerich, T. Ahmed, and N. Baykal, "Online DDoS attack detection using Mahalanobis distance and Kernel-based learning algorithm," *Journal of Network and Computer Applications*, vol. 168, article 102756, 2020.

[18] S. Dong and M. Sarem, "DDoS attack detection method based on improved KNN with the degree of DDoS attack in software-defined networks," *IEEE Access*, vol. 8, pp. 5039–5048, 2020.

[19] B. Hussain, Q. Du, B. Sun, and Z. Han, "Deep learning-based DDoS-attack detection for cyber–physical system over 5G network," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 2, pp. 860–870, 2021.

[20] A. S. Alzahrani, "An optimized approach-based machine learning to mitigate DDoS attack in cloud computing," *International Journal of Engineering Research and Technology*, vol. 13, no. 6, pp. 1441–1447, 2020.

[21] F. Li, J. Zhang, C. Shang, D. Huang, E. Oko, and M. Wang, "Modelling of a post-combustion CO_2 capture process using deep belief network," *Applied Thermal Engineering*, vol. 130, pp. 997–1003, 2018.

[22] S. Wang, H. Chen, L. Wu, and J. Wang, "A novel smart meter data compression method via stacked convolutional sparse auto-encoder," *International Journal of Electrical Power & Energy Systems*, vol. 118, article 105761, 2020.

[23] M. Skafi, M. M. Yunis, and A. Zekri, "Factors influencing SMEs' adoption of cloud computing services in Lebanon: an empirical analysis using toe and contextual theory," *IEEE Access*, vol. 8, pp. 79169–79181, 2020.

[24] http://205.174.165.80/CICDataset/CICDDoS2019/Dataset/PCAPs/01-12/PCAP-01-12_0750-0818.zip.