Research Article

Recurrent and Deep Learning Neural Network Models for DDoS Attack Detection

S. Sumathi,¹ R. Rajesh,² and Sangsoon Lim³

¹Department of Computer Science & Engineering, University V.O.C College of Engineering, Thoothukudi 628008, India
²Department of Engineering Design, Indian Institute of Technology Madras, Chennai 600036, India
³Department of Computer Engineering, Sungkyul University, Anyang 14097, Republic of Korea

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

Received 17 March 2022; Revised 10 August 2022; Accepted 22 August 2022; Published 10 September 2022

Academic Editor: Jaime Lloret

Copyright © 2022 S. Sumathi et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Distributed denial of service (DDoS) attack is a subclass of denial of service attack that performs severe attack in a cloud computing environment. It makes a malicious attempt to disturb the usual services of any network or server by using botnets. Hence, an efficient intrusion detection system (IDS) is essential to detect this attack. Some limitations in the existing IDS models for DDoS attack detection are delayed convergence, local stagnation issues, and local and global optimal trapping issues. These limitations are met by the recurrent neural network (RNN) and deep learning- (DL-) based proposed models that can utilize the previous states of the hidden neuron. The proposed research has used a long short-term memory (LSTM) recurrent neural network and autoencoder- and decoder-based deep learning strategy with gradient descent learning rule. The network parameters like weight vectors and bias coefficient are tuned optimally by employing the proposed a hybrid Harris Hawks optimization (HHO) and particle swarm optimization (PSO) algorithm. The proposed hybrid optimization algorithm selects the essential attributes, and the results obtained confirmed that the proposed LSTM and deep learning model outperformed all other models developed in the literature.

1. Introduction

1.1. Distributed Denial of Service Attack. The DDoS attack is one of the severe and most feared malicious cyber-attacks. It makes the website or a server down by flooding it with fraudulent traffic, ultimately making it inactive. Generally, DDoS attack packets have a high bit rate which performs network layer attacks [1]. A botnet master controls botnet machines present in various remote locations in this attack. Botnets are used in this type of attack as it is very similar to the normal traffic patterns of the Internet, and the owner is not aware of the commands received. Since numerous attack machines are involved, it is complicated to turn off these machines. The four components of a DDoS attack are attackers, master, zombie, and victim. This attack is generally classified into two types: bandwidth and resource depletion. In resource depletion, cloud resources are targeted, preventing legitimate users from accessing these resources. In bandwidth depletion, the victim’s network resources are targeted as shown in Figure 1.

1.2. Recurrent Neural Network. Recurrent neural networks are deep neural networks that can be trained on large volumes of databases and perform well on natural language processing, speech recognition, and other classification problems [2]. The recurrent neural networks differ from the feedforward neural network by their recurrent structure. It has storage units that are programmed to store the previous history of hidden states in hidden layers that are utilized to estimate the output of the current iteration. The basic
The structure of the RNN is shown in Figure 2. The layer units are represented in Figure 3, in which the weight values in the input, hidden, and output units are defined as $W^i_h$, $W^h_h$, and $W^o_h$. The previous hidden states are utilized during the learning process to compute the current iteration output through the delay unit $Z^{-1}$. So, the previous history of output is employed during the learning phase.

The long short-term memory network is an RNN proposed by Hochreiter in 1997 [3]. In practical applications, long short-term memory (LSTM) neural networks, which belong to gated RNNs, can learn long-term dependencies more quickly than the simple recurrent architectures [4]. The LSTM architecture handles the vanishing gradient problem. The data flow during the training process is maintained by switching special gates that decide when to read and write and what data to be stored in the gates coordinately. The LSTM architecture is presented in Figure 4, where the input gate, output gate, and the forget gate maintain the flow of signal between the layers with long-term learning dependencies. The stacked LSTM model is presented in Figure 5, where each layer has individually an LSTM framework.

The recurrent unit is updated by the following expression:

$$ z_t = f(z_{t-1}, x_t) = f(Vz_{t-1}WX_t + b) \quad (1) $$

where the hidden states are represented by $z_t$, the associated weight vectors are presented as $V$ and $W$, $b$ represents the bias coefficient, respectively, $f$ is the nonlinear activation function employed, and $x = [x_1, x_2, x_3 \cdots x_n]$ is the input vector. The sequence of the learning process is initialized with the decision made in forget gate, and the kind of information to be stored or thrown away is decided in this gate. The previous hidden states are estimated for each input $x_t$, and these values are employed over the sigmoid layer. If the obtained result is 1 then the data in $C_{t-1}$ is retained; else, it is removed. The following expression estimates the forget gate value:

$$ f_t = \sigma(Vz_{t-1} + WX_t + b_f) \quad (2) $$

Two factors decide the decision on new information to be stored in cell memory. Initially, the input layer decides with data to be updated, and then the vector of new input data is created in the $\tanh$ layer. Based on these two-step results, the vector to be updated into the memory cells is decided.

$$ i_t = \sigma(Vz_{t-1} + WX_t + b_i), \quad (3) $$

$$ \tilde{C}_t = \tanh (Vz_{t-1} + WX_t + b_c), \quad (4) $$

$$ C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t. \quad (5) $$

---

**Figure 1:** DDoS attack architecture.

**Figure 2:** Basic structure of recurrent neural network.

**Figure 3:** Architecture of RNN.
The output is estimated based on the information stored in cell state by the following expression:

\[ O_t = \sigma(V_O z_{t-1} + W_O X_t + b_O). \]  

(6)

In recent years, deep learning has become the dominant paradigm in machine learning and computer vision due to the availability of extensive public data and computational resources [5]. Deep learning strategies have been introduced in neural network models to perform training by employing numerous hidden layers over the conventional neural network architecture. The proposed research has used a deep learning strategy because the well-trained neural network model can provide better intrusion classification performance than the generic learning algorithms. The architecture of DLNN is shown in Figure 6. The DLNN comprises several invisible
| S. No. | Authors                  | Dataset                                      | Algorithm                  | Performance metrics                          | Application          | Remarks                                                                 |
|-------|--------------------------|----------------------------------------------|----------------------------|----------------------------------------------|----------------------|-------------------------------------------------------------------------|
| 1     | Kona et al. [7]          | CICIDS2017 dataset                           | Time series algorithm      | DR, ERR, TPR, TNR, precision, and FPR       | Cloud security       | Ensemble model with random forests and neural network model achieved the highest accuracy of 95.2% LDoS attack detection done using multi-feature fusion and convolution neural network with high accuracy and good stability |
| 2     | Tang et al. [8]          | NS2 simulation platform and test-bed platform | CNN, BP, and LSTM         | Detection rate, FN, and FP                  | IoT security         | Data traffic in industrial internet is collected using SDN architecture. The proposed LSTM-based model achieved a detection accuracy of 99% for large-scale flooding attack and 98% for stealthy flooding attack. The proposed 1-layer LSTM with a learning rate 0.1 and 2-layer LSTM with a learning rate of 0.2 have achieved better performance measures in DoS detection. LSTM has achieved higher performance metric values when compared to classifier algorithms like SVM and KNN in intrusion detection. The proposed hybrid CNN-LSTM IDS model performs intrusion detection with an accuracy of 96.32% which is much higher than the individual models. The classical machine learning and recurrent neural network algorithms But training time is much longer, and it suffers from high complexity. The proposed metaheuristic OCSA is used for important feature selection and classification is done using RNN which has inexpensive speed as well as memory requirements. The proposed OCSA-RNN model outperforms other existing metaheuristic RNN-based algorithms in terms of precision, recall, $F$-measure, and accuracy. The proposed autoencoder-based LSTM model performs DOS detection with better classification accuracy overcoming the drawbacks in time recurrent neural networks |
| 3     | Shen et al. [9]          | Network flow collected at SDN switch         | LSTM-based deep learning   | Accuracy, precision, recall, $F_1$ score, and FPR | Industrial internet  |                                                                           |
| 4     | Krishnan D et al. [10]   | CICIDS 2017                                  | LSTM, CNN                  | Accuracy, precision, $F_1$ score, recall, ROC score, and kappa score | Network security     |                                                                           |
| 5     | Boukhalfa et al. [11]    | NSL-KDD                                      | LSTM                       | Accuracy, sensitivity, false positive rate, precision, recall, and $F$-measure | Network security     |                                                                           |
| 6     | Abdallah et al. [12]     | In SDN dataset                               | CNN, LSTM                  | Accuracy, precision, recall, and $F_1$ measure | SDN                  |                                                                           |
| 7     | Imrana et al. [13]       | NSL-KDD                                      | Bidirectional long-short-term-memory (BiDLSTM) | Accuracy, precision, recall and $F$ score | Internet security    |                                                                           |
| 8     | Sai Sindhu Theja et al. [14] | KDD CUP 99                               | Oppositional Crow Search Algorithm (OCSA), recurrent neural network (RNN) | Precision, recall, $F$-measure, and accuracy | Cloud security       |                                                                           |
| 9     | Shaikh et al. [15]       | NSL-KDD                                      | Autoencoders and (LSTM)    | Classification accuracy                      | Cyber security       |                                                                           |
| S. No. | Authors | Dataset | Algorithm | Performance metrics | Application | Remarks |
|-------|---------|---------|-----------|--------------------|-------------|---------|
| 10    | Li et al. [16] | 1999 DARPA, 2009 DARPA, and UNB CIC DDoS 2019 | Quintile Deviation Check (QuinDC) | TPR, FPR, temporal false omission rate (TFOR) and false omission rate (FOR) | IoT security | The proposed QuinDC-based DDoS detection models perform real-time volumetric detection in IoT with accurate performance and low latency |
| 11    | Yang et al. [17] | UNB 2017, MAWI | Autoencoder- (AE-) based detection framework (AE-D3F) | Detection rate (DR), FPR, and FNR | Network security | The proposed AE-based DDoS detection framework (AE-D3F) achieved a detection rate of 82% with a 0 FPR value |
| 12    | Saharkhizan et al. [18] | Modbus network traffic | LSTM | Accuracy, precision, recall, and F-measure | IoT cyber-attacks | The novel ensemble LSTM deep models achieved an average accuracy of 98.99% and average detection rate of 92.32% in IoT cyber-attack detection |
| 13    | Parra et al. [19] | Detection of IoT botnet attacks N_BaIoT | LSTM and CNN | Precision, recall accuracy, TPR, TNR, FPR, and FNR | IoT security | The proposed CNN and back-end LSTM model perform phishing attack and botnet detection with an accuracy of 94.3% and 94.80%, respectively |
| 14    | Haider et al. [20] | CICIDS2017 | Deep learning- (DL-) based CNN | Accuracy, error rate, precision, recall, false positive rate, and F1 | Software-defined networking (SDN) | The proposed DL-based CNN model has achieved higher DDoS detection accuracy in minimum time with low computational complexity |
| 15    | Hussain et al. [21] | Open dataset released by Telecom Italia | CNN | Accuracy | Cyber security | The proposed robust CNN-based DDoS detection architecture has achieved higher than 91% normal and under attack cell detection accuracy |
| 16    | Ahmed et al. [22] | CTU-13 dataset | Feed-forward backpropagation ANN, deep learning artificial neural network | Accuracy | Cyber security | The proposed deep learning model accurately and efficiently identify real-time zero-day botnet attacks with highest accuracy when compared to SVM, NB, or backpropagation algorithms |
| 17    | Lam et al. [23] | IDS-2018 dataset | CNN | Precision, recall, and F1 score | IoT security | The proposed CNN-NAS (neural architecture search) has achieved 96.4% detection rate which is higher when compared to other models in literature |
| 18    | Velliangiri et al. [24] | KDD cup, database1, and database2 | Fuzzy and Taylor elephant herd optimization (FT-EHO), deep belief network (DBN) classifier | Detection accuracy, accuracy, precision, and recall | Cloud security | The proposed FT-EHO-DBN classifier outperformed other state-of-the-art algorithms in DDoS attack detection with ideal performance metric values but with higher computational cost |
| 19    | Agarval et al. [25] | KDD99 | KNN, CN, and LSTM | Accuracy | Cyber security | The proposed secured attack-avoidance technique (SAAT) with machine learning algorithms has achieved higher intrusion detection accuracy with less false feedbacks |
| S. No. | Authors          | Dataset                          | Algorithm     | Performance metrics                                      | Application | Remarks                                                                                                                                                                                                 |
|-------|------------------|----------------------------------|---------------|----------------------------------------------------------|-------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 20    | Diro et al. [26] | NSL-KDD, ISCX, and KDD CUP99     | Deep learning | Accuracy, DR, FAR, precision, recall, and F-measure     | IoT security | The proposed deep learning model achieved better evaluation metric values than the shallow model in binary and multiclass classification                                                                        |
| 21    | Yuan et al. [27] | ISCX2012                          | CNN, RNN, LSTM, and gated recurrent unit neural network (GRU) | Error rate, accuracy, precision, recall, F1, and AUC | Internet security | The proposed deep defense detection methodologies perform better DDoS detection than conventional machine learning algorithms with a reduced error rate of 2.103%                                                                 |
| 22    | Asad et al. [28] | CIC IDS 2017                      | DeepDetect neural network                        | F1 score, ROC                                       | Internet security | The proposed novel deep neural network-based detection mechanism has achieved a DDoS detection accuracy of 98% and an F1 score of 0.99                                                                 |
| 23    | Ng et al. [29]   | NSL-KDD UNSWNB15                  | Deep radial intelligence (DeeRai) with cumulative incarnation (Cul) | TN, FP, FN, TP, TPR, FPR, accuracy, and ER            | Cyber security   | The proposed stack AutoEncoder deep learning model is efficient in feature learning and performs attack classification with an average detection rate of 98.99% and an average false positive rate of 1.27%                                                                 |
| 24    | Meidan et al. [30] | IoTPOT dataset                   | Autoencoder, SVM, IsolationForest                | TPR, FPR, and detection time                        | IoT security    | The proposed stacked AutoEncoder deep learning model is efficient in feature learning and performs attack classification with an average detection rate of 98.99% and an average false positive rate of 1.27%                                                                 |
| 25    | Yadav et al. [31] | AL-DDoS attack dataset is created in smart and secure environment (SSE) laboratory | Deep belief network (DBN) and stacked AutoEncoder (SAE) | FPR, DR                                              | Web security    | The proposed BPN-based DDoS detection model achieved higher performance metric results in large-scale data                                                                                                           |
| 26    | Ke et al. [32]   | Simulated datasets                | BPNN, SVM, and decision tree                      | Accuracy rate, testing time, and false alarm rate (FAR) | Internet security | The proposed ANN-based DDoS detection model has achieved a 98.45% accuracy rate                                                                                                                          |
| 27    | Ustebay et al. [33] | CICIDS2017                        | Shallow neural network (SNN), DNN, and autoencoder | Accuracy                                             | Network security | The SVM-based classifier model outperformed all other models in DDoS attack detection with a minimal number of feature subset                                                                 |
| 28    | Sumathi et al. [34] | NSL-KDD                           | C4.5, SVM, and KNN                                | Accuracy, precision, sensitivity, specificity, and F1 score | Cloud security  | The performance of the proposed GRU deep learning method for DDoS detection fared better when compared with seven different approaches namely DNN, CNN, LSTM, SVM, LR, KNN, and GD because of its ability to learn long-term dependencies |
| 29    | Assis et al. [35] | CICDDoS 2019, CICIDS 2018         | Gated recurrent units (GRU)                       | Accuracy, precision, recall, and F-measure           | IoT security    | The proposed deep learning model achieved better evaluation metric values than the shallow model in binary and multiclass classification                                                                        |
layers, and each one carries a nonlinear transformation among the layers. The DLNN has been trained by unsupervised learning techniques and a backpropagation neural network. The unsupervised learning technique has utilized the autoencoder-decoder principle to pretrain the network and adopted a backpropagation neural network to fine-tune the DLNN.

The autoencoder is used in unsupervised learning methods, and its output is taken as the input data [6]. The encoder network transforms the input data into code and from high-dimensional space into low-dimensional space. Then the decoder converts the input into its original form. The encoder vector \( e^v \) in the encoder neural network is given in the following.

\[
e^v = e_f(x^v),
\]

where \( e_f \) represents the encoding function and \( x^v \) denotes the input data. In a decoder neural network, the reconstruction process is performed by its decoding function \( d_f \). This process maps the given dataset from low-dimensional space into high-dimensional space. The decoder process has been done by using the following.

\[
\hat{x}^v = d_f(e^v).
\]

The reconstruction error \( e(x, \hat{x}) \) is minimized by using these encoder and decoder processes for the number of trained samples. The term \( e(x, \hat{x}) \) is denoted as loss function, examining the inconsistency among encoded and decoded samples. The minimization of reconstruction error is the main objective of the unsupervised autoencoder.

\[
\delta_{mn}(\theta, \theta') = \frac{1}{N} \sum_{v=1}^{N} e(x^v, d_{g'}(e_{g}(x^v))).
\]

The function of encoding and decoding, along with the nonlinearity process, has been performed by using the following.

\[
e_{g}(x) = e_{af,x}(b + Wx),
\]

\[
d_{g'}(x) = d_{af,x}(b + W^Tx),
\]

where \( e_{af,x} \) and \( d_{af,x} \) represent the encoder and decoder activation functions. The network bias is indicated by \( b \), and the weight matrices of the network are given by \( W \) and \( W^T \). The reconstruction error process is as follows.

\[
e(x, \hat{x}) = \|x - \hat{x}\|^2.
\]

The pretraining of the DLNN model is carried out by developing the encoder process in the previous module. The input layer of the DLNN network with the first hidden layer is regarded as the encoder neural network of the first autoencoding process for the given input signal \( x^r \). The reconstruction error is minimized by training the first autoencoder process. The first trained parameter of the encoder neural network is used to initialize the first hidden layer of the DLNN process using the following.

\[
e_{g}^1 = e_{g_1}(x^r).
\]

Now, the input data becomes the encoder vector \( e_{g}^1 \). The encoder neural network for the second autoencoder is obtained from the first and second hidden layers of DLNN. Next, the second trained autoencoder is used to initialize the second hidden layer of the DLNN network. The above-
performed process is continued till the final hidden layer of the DLNN model. The generalized form of the final encoder vector is given in the following.

\[ e_N^v = e_{\theta_N}(E_{N-1}^v) \]  

(14)

The \( N \)th trained parameter of the encoder neural network is denoted by \( \theta_N \). The hidden layer of DLNN is pretrained by the \( N \)-stacked encoder process. This pretrained process avoids the local minima and improves the generalization aspect. The output of the DLNN model is calculated by using the following.

\[ y^v = e_{\theta_{N+1}}(e_N^v) \]  

(15)

The trained parameter of the output layer is denoted by \( \theta_{N+1} \). The output error is reduced by using the backpropagation algorithm.

### 2. Related Works on DDoS Attacks Detection

In this section, an overview of existing intrusion detection techniques for DDoS attacks is discussed in detail and tabulated in Table 1. The proposed DDoS detection algorithms and techniques are analyzed based on their performance metrics.

### 3. Proposed Hybrid Optimization Algorithm

The proposed hybrid swarm intelligent optimization algorithm serves a twofold purpose. Initially, the algorithm is employed to select the significant features employed for the attack identification and tune the proposed neural network-based IDS models with optimal parameter settings. These objectives have been achieved by employing the Harris Hawks optimization algorithm, which has limitations during the training process. These limitations are addressed by combining HHO optimizer with particle swarm optimization.

---

**Algorithm 1:** Pseudocode of proposed hybrid HHO-PSO algorithm.

| Input: population size, convergence criteria, random factors, acceleration coefficient, inertia factor, upper and lower bounds |
| Output: the fitness value and the corresponding position of the prey |
| 1: Initialize the population |
| 2: while (stopping criteria) do |
| 3: Fitness (all Hawks in population) |
| 4: if current_pBest > pBest then pBest = current_pBest else pBest = pBest |
| 7: gBest = particle with best pBest among the population |
| 8: Define the position of the rabbit |
| 9: for (all Hawks) do |
| 10: Update the initial energy level of prey and its jumping power |
| 11: Update the current energy level of prey |
| # Exploration Phase |
| 12: if (|E| ≥ 1) then |
| 13: The position of each hawk in the population is adjusted by equation |
| \[ X(t + 1) = \begin{cases} X_{\text{rand}}(t) - r_1 |X_{\text{rand}}(t) - 2r_2 X(t)| + v(t + 1) & q ≥ 0.5 \\ X_{\text{rabbit}}(t) - X_n(t) - r_3 (LB + r_4 (UB - LB)) + v(t + 1) & q < 0.5 \end{cases} \] |
| # Exploitation Phase |
| 14: if (|E| ≤ 1) then |
| 15: if (r < 0.5 and |E| ≥ 0.5) then |
| # Soft Besiege |
| 16: Adjust the Hawks position by equation \[ X(t + 1) = \Delta X(t) - E|X_{\text{rabbit}}(t) - X(t)| \] |
| # Hard Besiege |
| 17: else if (r ≥ 0.5 and |E| < 0.5) then |
| 18: Adjust the position by equation \[ X(t + 1) = X_{\text{rabbit}}(t) - E|\Delta X(t)| \] |
| # Soft besiege with dives |
| 19: else if (r < 0.5 and |E| ≥ 0.5) then |
| 20: \[ Y = X_{\text{rabbit}}(t) - E|X_{\text{rabbit}} - X_n(t) | \] |
| 21: \[ Z = Y + SxLF(D) \] |
| 22: \[ X(t + 1) = Y \text{ if } (F(Y) < F((X(t)))) \text{ and } (F(Z) < F((X(t)))) \] |
| # Hard besiege |
| 23: else if (r < 0.5 and |E| < 0.5) then |
| 24: \[ Y = X_{\text{rabbit}}(t) - E|X_{\text{rabbit}} - X_m(t) | \] |
| 25: \[ Z = Y + SxLF(D) \] |
| 26: \[ X(t + 1) = Y \text{ if } (F(Y) < F((X(t)))) \text{ and } (F(Z) < F((X(t)))) \] |
| 27: return the solution |
The conventional HHO algorithm suffers from poor exploration ability as the Hawks need to wait for prey from several minutes to hours. This limitation has been eliminated by improving the convergence speed of the algorithm, which has been done by integrating particle swarm optimization and HSO. PSO is a population-based optimization technique applied extensively to many engineering problems [40]. The proposed research has chosen the PSO algorithm because of its simplicity and excellent exploration ability. The detailed pseudocode of the PSO algorithm is presented in [41]. The advantages of HHO and PSO have been combined to give a hybrid HHO algorithm to attain a tradeoff between exploration and exploitation mechanisms than

algorithm and also attain better tradeoff between exploration and exploitation ability of the algorithm. The main inspiration of HHO is the cooperative behavior and chasing style of Harris Hawks in nature called surprise pounce. The detailed pseudo code of the HHO algorithm is presented in [39].

The proposed hybrid models.

Table 2: Parameters of the proposed model.

| Parameters                  | LSTM or DLNN model              |
|-----------------------------|---------------------------------|
| Weights and bias            | Optimally fed by HHO-PSO        |
| Number of input neurons     | Number of selected features     |
| Number of hidden layers     | 2                               |
| Number of hidden neurons    | Initialized to (6-8) for LSTM    |
| Number of output neurons    | and (7-12) for DLNN, fixed      |
| Activation function         | during training                 |
| Learning rate               | Sigmoidal activation function    |
| Learning rule               | 0.25 (fixed at end trial)       |
| Hybrid HHO-PSO              | Gradient descent rule           |
| Population size             | 100                             |
| Maximum number of iterations| Until convergence attained       |
| ($u, v$)                    | ($0, 1$)                        |
| $\beta$                     | 1.5                             |
| Initial energy state $E_0$  | ($0, 1$)                        |

Table 3: The selected features by proposed HHO-PSO algorithm.

| No. | Selected features                      | No. | Selected features                      |
|-----|----------------------------------------|-----|----------------------------------------|
| 1   | F3 (service)                           | 2   | F4 (flag)                              |
| 3   | F5 (src_bytes)                         | 4   | F6 (dst_bytes)                         |
| 5   | F12 (logged_in)                        | 6   | F25 (serror_rate)                      |
| 7   | F30 (diff_srv_rate)                    | 8   | F39 (dst_host_srv_serror_rate)         |
HHO and other conventional algorithms. The pseudocode of the proposed hybrid HHO-PSO algorithm is shown in Algorithm 1.

4. Proposed Hybrid HO-PSO-LSTM and Deep Learning Models

The proposed hybrid learning model is shown in Figure 7. The original dataset has 41 features, and the main objective of the proposed models is to attain better accuracy with a reduced number of features. So, the optimal feature selection is performed by employing the proposed HHO-PSO optimization algorithm; the algorithm parameters values are presented in Table 2. The selected features are shown in Tables 3 and 4 and fed into the network, and the corresponding performance is evaluated. 10-fold cross-validation is employed for each fold to select the optimal features, and the selected features are tabulated. The optimal features are identified at the end of the 10-fold cross-validation process based on the frequency of occurrences. The selected features by proposed HHO-PSO algorithm for LSTM is given in Table 5, and the frequency of selected features between the proposed model and existing models is presented in Table 6.

5. Result Comparison and Discussion for Proposed Hybrid HHO-PSO-LSTM

The proposed IDS models are evaluated with NSL benchmark datasets. The model is iterated for ten trial runs to avoid biased output, and the model performance for each trial run is depicted in Figure 8.

The average performance of the proposed LSTM model for the 10-trial run is presented in Table 7 and Figure 9. The classic LSTM model reported an accuracy of 0.9541, but it has poor true negative class identification as compared to the hybrid models. Feeding the model with a PSO-based feature selection strategy enhances performance more than the conventional LSTM strategy. Compared to PSO, on incorporating the HHO strategy, the true negative and true positive cases are improved. So, the model is fed with a hybrid feature selection strategy, and the performance is investigated compared to the other two algorithms. The performance is improved with false negative cases significantly reduced. This shows the significance of the proposed hybrid HHO-PSO optimization strategy in enhancing the performance of the conventional LSTM model.

6. Result Comparison and Discussion for Proposed Hybrid HHO-PSO Deep Learning IDS Model

The size of the original dataset has reduced after the feature selection process, and the selected feature subset is fed into the proposed deep learning model for intrusion classification. Parameter handling is the major challenge of the deep learning models, which has been done by optimally choosing the weight and bias vectors of the proposed model. Initially, numerous trials are made to fix the number of hidden layers for developing the DLNN model. The number of hidden

| Table 4: Selected features of the proposed algorithm during 10-fold cross-validation. |
|-----------------------------------------|-----------------------------------------|
| Proposed hybrid HHO-PSO               | Proposed hybrid HHO-PSO               |
| #1 F3, F4, F5, F6, F8, F10, F12, F25, F26, F29, F30, F35, F36, F37, and F39 | #6 F3, F4, F5, F6, F8, F9, F12, F23, F25, and F39 |
| #2 F4, F5, F6, F8, F12, F25, F26, F30, F35, F36, and F39 | #7 F3, F4, F5, F6, F8, F12, F23, F25, F29, F30, and F38 |
| #3 F3, F4, F5, F6, F8, F10, F12, F23, F25, F26, F30, F36, and F39 | #8 F3, F4, F5, F6, F8, F12, F23, F25, F30, F35, and F39 |
| #4 F4, F5, F6, F8, F10, F12, F23, F25, F26, F36, and F39 | #9 F3, F4, F5, F6, F12, F23, F25, F30, F35, F36, and F39 |
| #5 F3, F4, F5, F12, F23, F25, F26, F30, F35, and F39 | #10 F3, F4, F5, F10, F12, F25, F30, F35, and F39 |

| Table 5: The selected features by proposed HHO-PSO algorithm for LSTM. |
|-----------------------------------------|-----------------------------------------|
| Feature No. | Attribute               |
| F3 | Service |
| F4 | Flag |
| F5 | src_bytes |
| F6 | dst_bytes |
| F12 | logged_in |
| F25 | serror_rate |
| F30 | diff_srv_rate |
| F39 | dst_host_srv_serror_rate |

The random initialization of weight vectors and bias coefficients affects the conventional neural network model. The proposed LSTM model is optimally framed by feeding optimal weight and bias coefficients by the proposed HHO-PSO optimization algorithm that improves the convergence speed presented in Figure 10. The conventional LSTM model converges at the 353rd iteration. In contrast, the proposed optimally constructed LSTM model starts to converge at the 200th iteration, and also the reported lacuna of delayed convergence is handled effectively by the proposed LSTM model. To further demonstrate the effectiveness of the proposed model, the model performances are compared with the existing models in the works of literature and other models, as presented in Table 8. The reported results inferred that the proposed hybrid HHO-PSO-LSTM IDS model outperformed all other models with better intrusion classification performance.
layers and hidden neurons is another factor deciding the complexity of the network. So, the hidden layers and hidden neurons are fixed based on the trial and error method, and the model performances for various hidden layers are presented in Table 9.

The error rate of the proposed model for 10 trial run and their corresponding hidden layers are shown in Figure 11. Handling the overfitting issue is the primary task of the deep learning models, which can be done by analyzing the training efficiency of the proposed model. The training and testing accuracy for various hidden layers are shown in Figure 12, which has confirmed that the network underfits till trial 7 and it overfits from trial 9. So, the overfitting and the underfitting issue has been avoided by fixing 12 numbers of hidden layers. Further, Table 10 confirmed that the proposed model achieved better intrusion classification performance for 12 hidden layers. So, the proposed model has fixed 12 numbers hidden layers. Comparison of the proposed models with existing models is shown in Table 11. Comparison of percentage improvement of the proposed HHO-PSO-DLNN model with other models is shown in Table 10. Comparison of computational time involved in the proposed models and algorithms is tabulated in Table 12. Table 12 shows that the computational time of the proposed hybrid HHO-PSO-DLNN is higher than all the other models under comparison due to its increase in the number of hidden layers. But the important tradeoff is that the performance metric values in DDoS attack detection in proposed optimized DL are much better than other algorithms.

The proposed DLNN model is tuned for its weight and bias vectors by individually employing PSO and HHO algorithms and then using the proposed hybrid HHO-PSO optimization algorithm. The performance comparison of conventional DLNN, PSO-DLNN, HHO-DLNN, and hybrid HHO-PSO-DLNN has been done and is shown in Figure 13. ROC curve and rate of convergence of the proposed model are shown in Figures 14 and 15. The performance of the proposed hybrid HHO-PSO-DLNN model is observed to be better than the conventional DLNN, PSO-DLNN, and HHO-DLNN models.

7. Statistical Analysis of the Proposed Models

Dietterich [58] recommended the statistical models that can be executed 10 times. The $5 \times 2$ cv test is a powerful strategy for measuring the classifier algorithms’ statistical variation. In the proposed scheme, twofold cross-validation is performed five times, where 50% of the data sample is employed for training, and the remaining 50% is employed for testing. Then the dataset is shared such that the testing sample is employed for training and the training sample is adopted for testing, respectively. The performance difference between

| Feature | C4.5 | KNN | SVM | PSO | HHO | HHO-PSO |
|---------|------|-----|-----|-----|-----|---------|
| F2      | 0    | 0   | 0   | 0   | 1   | 0       |
| F3      | 3    | 0   | 0   | 5   | 7   | 8       |
| F4      | 9    | 10  | 10  | 8   | 9   | 10      |
| F5      | 10   | 10  | 10  | 9   | 10  | 10      |
| F6      | 1    | 4   | 0   | 4   | 4   | 8       |
| F7      | 1    | 0   | 0   | 0   | 0   | 0       |
| F8      | 10   | 10  | 10  | 8   | 10  | 7       |
| F10     | 10   | 10  | 10  | 7   | 8   | 4       |
| F11     | 1    | 0   | 0   | 3   | 0   | 0       |
| F12     | 10   | 10  | 10  | 9   | 10  | 10      |
| F23     | 10   | 9   | 10  | 9   | 10  | 7       |
| F25     | 10   | 9   | 3   | 7   | 9   | 10      |
| F26     | 0    | 8   | 7   | 0   | 0   | 0       |
| F29     | 10   | 10  | 10  | 8   | 4   | 2       |
| F30     | 10   | 10  | 10  | 8   | 10  | 9       |
| F33     | 0    | 7   | 0   | 0   | 0   | 0       |
| F34     | 0    | 0   | 1   | 0   | 0   | 0       |
| F35     | 9    | 10  | 10  | 8   | 8   | 6       |
| F36     | 10   | 10  | 9   | 10  | 8   | 5       |
| F37     | 8    | 9   | 4   | 8   | 7   | 1       |
| F38     | 1    | 3   | 5   | 2   | 4   | 1       |
| F39     | 0    | 10  | 2   | 3   | 5   | 8       |
| F40     | 0    | 0   | 0   | 0   | 0   | 0       |
| F41     | 1    | 0   | 0   | 0   | 0   | 0       |
| Total number of selected features | 12 | 14 | 10 | 10 | 10 | 8 |
Input: initialize the necessary parameters such as weight vector, α, input, output, and hidden layer neuron numbers, stopping criterion, population size, random factors, acceleration coefficient, inertia factor, upper and lower bounds

Output: fitness value and MSE

1: all input – target pair present the input
2: while (stopping condition)
3: for i = 1 to number of autoencoders
4: compute the net input of the layer
5: employ tangential activation function
6: find the output of the autoencoder layer
7: The position of each hawk in the population is adjusted by the equations
8: update the current energy level of prey
9: update the initial energy level of prey and its jumping power
10: fine-tuning of the network is performed by employing backpropagation strategy
11: compute the reconstruction error
12: if (convergence criteria)
13: stop the process return the output
14: else
15: invoke the proposed Hybrid HHO-PSO optimizer algorithm
16: while (stopping criteria) do
17: Fitness (all Hawks in population)
18: if current. pBest > pBest
19: then pBset = current. pBest
20: else
21: pBest = pBest
22: gBest = particle with best pBest among the population
23: define the position of the rabbit
24: for (all Hawks)
25: update the initial energy level of prey and its jumping power
26: update the current energy level of prey
27: if (|E| ≥ 1)
28: the position of each hawk in the population is adjusted by the equations
29: if (|E| ≤ 1)
30: if (r ≥ 0.5 and |E| ≥ 0.5)
31: adjust the Hawks position by ΔX(t) = Xrabbit(t) – X(t)
32: else if (r ≥ 0.5 and |E| < 0.5)
33: adjust the Hawks position by X(t + 1) = Xrabbit(t) – E|ΔX(t)|
34: else if (r < 0.5 and |E| ≥ 0.5)
35: Y = Xrabbit(t) – E |Xrabbit – X(t)|
36: Z = Y + SxLF(D)
37: X(t + 1) = Y if (F(Y) < F(X(t)))
38: Z if (F(Z) < F(X(t)))
39: else if
40: Y = Xrabbit(t) – E |Xrabbit – X(t)|
41: Z = Y + SxLF(D)
42: X(t + 1) = Y if (F(Y) < F(X(t)))
43: Z if (F(Z) < F(X(t)))
44: return weight values
45: end if
46: end while
47: stop

Algorithm 2: Pseudocode of the proposed hybrid deep learning IDS model
The mean and variance of difference are evaluated by using the following equations.

\[
\text{Acc}_{\text{avg}} = \frac{(\text{Acc}_A + \text{Acc}_B)}{2},
\]

\[
\sigma^2 = \left(\frac{\text{Acc}_A - \text{Acc}_{\text{avg}}}{\sigma_1^2} + \frac{\text{Acc}_B - \text{Acc}_{\text{avg}}}{\sigma_2^2}\right).
\]

The \(t\) statistics is evaluated after 5 iterations as follows:

\[
t = \frac{\text{Acc}_{A,1}}{\sqrt{(1/5)\sum_{i=1}^{5}\sigma_i^2}},
\]

where \(\text{Acc}_{A,1}\) is the accuracy of the first iteration sample.

The \(t\) distribution with 5 degrees of freedom is followed by \(t\) statistics. The \(p\) value is estimated and compared with the level of significance \(\alpha = 0.05\). When the \(p\) value is less than the level of significance, then the null hypothesis is rejected, and both the models reported similar performance. This signifies the estimated difference to be real. Otherwise, if the \(p\) value is greater than the significance level, then the null hypothesis is not rejected. The difference obtained in performance is probably due to stochastic factors or a statistical coincidence. The performance of the proposed HHO-PSO-tuned DLNN model is compared with the performance of other models and confirmed that the proposed HHO-PSO possesses better performance, which is reported in Table 13. The results obtained confirmed that the proposed HHO-PSO-DLNN model is statistically better than all other models developed in this proposed research work.
Figure 10: Convergence graph.

Table 8: Comparative analysis made with other models in literatures.

| Model under study         | Accuracy | Precision | Sensitivity | Specificity | F1 score |
|---------------------------|----------|-----------|-------------|-------------|----------|
| Naïve Bayes [38]          | 0.9216   | 0.9649    | 0.9039      | 0.9490      | 0.9334   |
| CART [39]                 | 0.8993   | 0.9491    | 0.8827      | 0.9253      | 0.9147   |
| ABC-BPN [40]              | 0.9116   | 0.9585    | 0.8939      | 0.9393      | 0.9250   |
| BR-BPN [41]               | 0.8989   | 0.9556    | 0.8777      | 0.9335      | 0.9150   |
| GA-BPN [42]               | 0.9212   | 0.9618    | 0.9056      | 0.9450      | 0.9328   |
| PSO-BPN [43]              | 0.9096   | 0.9618    | 0.8886      | 0.9434      | 0.9237   |
| GA-MLP [44]               | 0.9140   | 0.9588    | 0.8972      | 0.9401      | 0.9270   |
| MLP [45]                  | 0.9229   | 0.9694    | 0.9025      | 0.9551      | 0.9347   |
| LSTM [46]                 | 0.9545   | 0.9607    | 0.9594      | 0.9480      | 0.9600   |
| LSTM [47]                 | 0.9542   | 0.9676    | 0.9527      | 0.9563      | 0.9601   |
| C4.5 [34]                 | 0.9370   | 0.9680    | 0.9248      | 0.9549      | 0.9459   |
| KNN [34]                  | 0.9170   | 0.9368    | 0.9189      | 0.9143      | 0.9278   |
| SVM [34]                  | 0.9494   | 0.9585    | 0.9528      | 0.9448      | 0.9557   |
| BPN [48]                  | 0.9272   | 0.9663    | 0.9111      | 0.9515      | 0.9379   |
| MLP [49]                  | 0.9380   | 0.9726    | 0.9227      | 0.9611      | 0.9470   |
| PSO-BPN [50]              | 0.9440   | 0.9732    | 0.9315      | 0.9623      | 0.9519   |
| PSO-MLP [51]              | 0.9473   | 0.9722    | 0.9376      | 0.9613      | 0.9546   |
| HHO-BPN [52]              | 0.9501   | 0.9686    | 0.9451      | 0.9571      | 0.9567   |
| HHO-MLP [53]              | 0.9584   | 0.9699    | 0.9576      | 0.9596      | 0.9637   |
| LSTM                     | 0.9541   | 0.9592    | 0.9601      | 0.9461      | 0.9597   |
| PSO-LSTM                 | 0.9631   | 0.9666    | 0.9684      | 0.9560      | 0.9675   |
| HHO-LSTM                 | 0.9682   | 0.9693    | 0.9747      | 0.9597      | 0.9720   |
| Hybrid HHO-PSO-BPN       | 0.9708   | 0.9725    | 0.9761      | 0.9638      | 0.9743   |
| HHO-PSO-MLP              | 0.9774   | 0.9763    | 0.9838      | 0.9690      | 0.9800   |
| Proposed hybrid HHO-PSO-LSTM | **0.9853** | **0.9832** | **0.9909** | **0.9780** | **0.9870** |
Table 9: Performance of the proposed DLNN for various hidden layers.

| Trial runs | No. of hidden layer | Accuracy | Precision | Sensitivity | Specificity | F1 score | Error rate | Computational time in sec |
|------------|---------------------|----------|-----------|-------------|-------------|----------|------------|--------------------------|
| 1          | 5                   | 0.9881   | 0.9908    | 0.9883      | 0.9878      | 0.9895   | 0.0119     | 200.22                   |
| 2          | 6                   | 0.9868   | 0.9902    | 0.9897      | 0.9871      | 0.9900   | 0.0114     | 220.98                   |
| 3          | 7                   | 0.9859   | 0.9900    | 0.9853      | 0.9867      | 0.9876   | 0.0141     | 259.98                   |
| 4          | 8                   | 0.9862   | 0.9898    | 0.9859      | 0.9864      | 0.9879   | 0.0138     | 330.07                   |
| 5          | 9                   | 0.9870   | 0.9895    | 0.9877      | 0.9861      | 0.9886   | 0.0130     | 365.23                   |
| 6          | 10                  | 0.9870   | 0.9889    | 0.9883      | 0.9853      | 0.9886   | 0.0130     | 379.93                   |
| 7          | 11                  | 0.9880   | 0.9875    | 0.9914      | 0.9836      | 0.9895   | 0.0120     | 380.23                   |
| 8          | 12                  | 0.9953   | 0.9969    | 0.9926      | 0.9959      | 0.9947   | 0.0047     | 393.12                   |
| 9          | 13                  | 0.9986   | 0.9990    | 0.9986      | 0.9986      | 0.9988   | 0.0014     | 439.98                   |
| 10         | 14                  | 0.9993   | 0.9993    | 0.9995      | 0.9991      | 0.9994   | 0.0007     | 459.32                   |

Figure 11: Error rate for 10-trial runs.

Figure 12: Training efficiency of the proposed model for 10-trial runs.
### Table 10: Comparison of percentage improvement of the proposed HHO-PSO-DLNN model with other models.

| Model under study | Accuracy | Precision | Sensitivity | Specificity | F1 score |
|-------------------|----------|-----------|-------------|-------------|----------|
| C4.5 [34]         | 5.83     | 2.89      | 6.78        | 4.1         | 4.88     |
| KNN [34]          | 7.83     | 6.01      | 7.37        | 8.16        | 6.69     |
| SVM [34]          | 4.59     | 3.84      | 3.98        | 5.11        | 3.9      |
| BPN [52]          | 6.81     | 3.06      | 8.15        | 4.44        | 5.68     |
| MLP [53]          | 5.73     | 2.43      | 6.99        | 3.48        | 4.77     |
| PSO-BPN [54]      | 5.13     | 2.37      | 6.11        | 3.36        | 4.28     |
| PSO-MLP [55]      | 4.80     | 2.47      | 5.50        | 3.46        | 4.01     |
| HHO-BPN [56]      | 4.52     | 2.83      | 4.75        | 3.88        | 3.8      |
| HHO-MLP [57]      | 3.69     | 2.70      | 3.50        | 3.63        | 3.1      |
| LSTM              | 4.12     | 3.77      | 3.25        | 4.98        | 3.5      |
| PSO-LSTM          | 3.22     | 3.03      | 2.42        | 3.99        | 2.72     |
| HHO-LSTM          | 2.71     | 2.76      | 1.79        | 3.62        | 2.27     |
| DLNN              | 2.67     | 1.75      | 2.67        | 2.36        | 2.21     |
| PSO-DLNN          | 1.78     | 0.77      | 2.09        | 1.05        | 1.43     |
| HHO-DLNN          | 1.42     | 1.12      | 1.15        | 1.49        | 1.13     |
| Proposed hybrid HHO-PSO-LSTM | 1   | 1.37      | 0.17        | 1.79        | 0.77     |

### Table 11: Comparison of proposed models with existing models.

| Model under study     | Accuracy | Precision | Sensitivity | Specificity | F1 score |
|-----------------------|----------|-----------|-------------|-------------|----------|
| Naïve Bayes [42]      | 0.9216   | 0.9649    | 0.9039      | 0.9490      | 0.9334   |
| CART [43]             | 0.8993   | 0.9491    | 0.8827      | 0.9253      | 0.9147   |
| ABC-BPN [44]          | 0.9116   | 0.9585    | 0.8939      | 0.9393      | 0.9250   |
| BR-BPN [45]           | 0.8989   | 0.9556    | 0.8777      | 0.9335      | 0.9150   |
| GA-BPN [46]           | 0.9212   | 0.9618    | 0.9056      | 0.9450      | 0.9328   |
| PSO-BPN [47]          | 0.9096   | 0.9618    | 0.8866      | 0.9434      | 0.9237   |
| GA-MLP [48]           | 0.9140   | 0.9588    | 0.8972      | 0.9401      | 0.9270   |
| MLP [49]              | 0.9229   | 0.9694    | 0.9025      | 0.9551      | 0.9347   |
| LSTM [50]             | 0.9545   | 0.9607    | 0.9594      | 0.9480      | 0.9600   |
| LSTM [51]             | 0.9542   | 0.9676    | 0.9527      | 0.9563      | 0.9601   |
| C4.5 [34]             | 0.9370   | 0.9680    | 0.9248      | 0.9549      | 0.9459   |
| KNN [34]              | 0.9170   | 0.9368    | 0.9189      | 0.9143      | 0.9278   |
| SVM [34]              | 0.9494   | 0.9585    | 0.9528      | 0.9448      | 0.9557   |
| BPN [52]              | 0.9272   | 0.9663    | 0.9111      | 0.9515      | 0.9379   |
| MLP [53]              | 0.9380   | 0.9726    | 0.9227      | 0.9611      | 0.9470   |
| PSO-BPN [54]          | 0.9440   | 0.9732    | 0.9315      | 0.9623      | 0.9519   |
| PSO-MLP [55]          | 0.9473   | 0.9722    | 0.9376      | 0.9613      | 0.9546   |
| HHO-BPN [56]          | 0.9501   | 0.9686    | 0.9451      | 0.9571      | 0.9567   |
| HHO-MLP [57]          | 0.9584   | 0.9699    | 0.9576      | 0.9596      | 0.9637   |
| LSTM                  | 0.9541   | 0.9592    | 0.9601      | 0.9461      | 0.9597   |
| PSO-LSTM              | 0.9631   | 0.9666    | 0.9684      | 0.9560      | 0.9675   |
| HHO-LSTM              | 0.9682   | 0.9693    | 0.9747      | 0.9597      | 0.9720   |
| DLNN                  | 0.9686   | 0.9794    | 0.9659      | 0.9723      | 0.9726   |
| PSO-DLNN              | 0.9775   | 0.9892    | 0.9717      | 0.9854      | 0.9804   |
| HHO-DLNN              | 0.9811   | 0.9857    | 0.9811      | 0.9810      | 0.9834   |
| Proposed hybrid HHO-PSO-LSTM | 0.9853 | 0.9832    | 0.9909      | 0.9780      | 0.9870   |
| Proposed HHO-PSO-DLNN | 0.9953   | 0.9969    | 0.9926      | 0.9959      | 0.9947   |
Table 12: Comparison of computational time involved in the proposed model and algorithms.

| Model under study                        | Computational time in seconds |
|------------------------------------------|-------------------------------|
| PSO-LSTM                                 | 280.4                         |
| HHO-LSTM                                 | 290.6                         |
| Proposed hybrid HHO-PSO-LSTM             | 300.7                         |
| Hybrid HHO-PSO-BPN                       | 350.2                         |
| Hybrid HHO-PSO-MLP                       | 360.8                         |
| Hybrid HHO-PSO-DLNN                      | 393.12                        |

Figure 13: Performance of the proposed DLNN model with other models.

Figure 14: ROC curve for the proposed DLNN model.
8. Conclusion and Future Directions

The proposed model in this paper employed a recurrent neural network and deep learning neural network model for the intrusion classification problem. The LSTM network is adopted, and its intrusion detection performances are investigated. The hybrid HHO-PSO algorithm is employed to improve the model’s performance, and the model response is analyzed by the metric values obtained. It is investigated that the proposed hybrid HHO-PSO optimization algorithm has improved the performance of the neural network models by providing a minimal number of optimal feature subsets with better classification accuracy. The convergence speed of the proposed model is improved than the other models in this study and demonstrated better performance than other models in the literature. The deep learning architecture for DDoS attack detection based on autoencoder-decoder strategy is initially framed based on numerous trial results. Then, the weight vectors of the proposed DLNN model are optimally tuned by the proposed hybrid HHO-PSO optimization algorithm. The model performances are analyzed for each trial run. Further, the effectiveness of the proposed model has been analyzed by comparing the model’s performance with other existing works. Finally, a statistical analysis is made based on 5×2 cv test to justify the superiority of the proposed hybrid HHO-PSO-DLNN model with all other existing models. It is confirmed that the proposed DLNN model outperformed all other models in intrusion detection under a cloud computing environment.

The following are the future research directions for DDoS attack detection using DL methods:

(i) DDoS DL-based detection methods can be combined with eXplainable Artificial Intelligence (XAI) techniques, which leads to global interpretation [59]

(ii) The proposed DDoS detection DL-based methods can be implemented in a larger system with which it is easy to detect the compromised end points. It also helps us to improve the performance of the proposed algorithms making them skillful in handling the various abnormalities that occurs in the performance of the network

(iii) Efficient and lightweight DL models can be developed for attack prone networks which has limited resources for computing

(iv) Pattern of attacks changes at a much faster rate, and hence, automatic updated DL models can be developed to detect new DDoS attack instances

(v) The increased computational time of the proposed hybrid HHO-PSO-DLNN can further be reduced in future, retaining its ideal performance metric values in DDoS attack detection
Acknowledgments

The authors declare that they have no conflicts of interest.

Data Availability

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

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] S. Sumathi and R. Rajesh, "Comparative study on TCP SYN flood DDoS attack detection: a machine learning algorithm based approach," WSEAS Transactions on Systems and Control, vol. 16, pp. 584–591, 2021.

[2] P. Dixit and S. Silakari, "Analysis of state-of-art attack detection methods using recurrent neural network," Proceedings of the international conference on paradigms of communication, computing and data sciences, , pp. 795–804, Springer, Singapore, 2022.

[3] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural Computation, vol. 9, no. 8, pp. 1735–1780, 1997.

[4] Z. Y. Peng and P. C. Guo, "A data organization method for LSTM and transformer when predicting Chinese banking stock prices," Discrete Dynamics in Nature and Society, vol. 2022, Article ID 7119678, 8 pages, 2022.

[5] K. Chumachenko, A. Iossifidis, and M. Gabbouj, "Feedforward neural networks initialization based on discriminant learning," Neural Networks, vol. 146, pp. 220–229, 2022.

[6] S. S. Ilango, S. Vimal, M. Kaliappan, and P. Subbulakshmi, "Optimization using artificial bee colony based clustering approach for big data," Cluster Computing, vol. 22, Supplement 5, pp. 12169–12177, 2019.

[7] S. S. Kona, Detection of DDoS attacks using RNN-LSTM and Hybrid model ensemble, National College of Ireland, Doctoral dissertation, Dublin, 2020.

[8] D. Tang, L. Tang, W. Shi, S. Zhan, and Q. Yang, "MF-CNN: a new approach for LDoS attack detection based on multi-feature fusion and CNN," Mobile Networks and Applications, vol. 26, no. 4, pp. 1705–1722, 2021.

[9] C. Shen, G. Xiao, S. Yao, B. Zhou, Z. Pan, and H. Zhang, "An LSTM based malicious traffic attack detection in industrial internet," in 2021 International Conference on Security, Pattern Analysis, and Cybernetics(SPAC), pp. 60–65, Chengdu, China, June 2021.

[10] D. Krishnan, "Detection of denial-of-service attacks using stacked LSTM networks," Proceedings of data analytics and management, , pp. 229–239, Springer, Singapore, 2022.

[11] A. Boukhalfa, A. Abdellauoi, N. Hmina, and H. Chaoui, "LSTM deep learning method for network intrusion detection system," International Journal of Electrical and Computer Engineering, vol. 10, no. 3, p. 3315, 2020.

[12] M. Abdallah, N. An Le Khac, H. Jahromi, and A. Delia Jurcut, "A hybrid CNN-LSTM based approach for anomaly detection systems in SDNs," in The 16th International Conference on Availability, Reliability and Security, pp. 1–7, Vienna, Austria, August 2021.

[13] Y. Imrana, Y. Xiang, L. Ali, and Z. Abdul-Rauf, "A bidirectional LSTM deep learning approach for intrusion detection," Expert Systems with Applications, vol. 185, article 115524, 2021.

[14] 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, article 106997, 2021.

[15] R. A. Shaikh and S. V. Shashikala, "An autoencoder and LSTM based intrusion detection approach against denial of service attacks," in 2019 1st international conference on advances in information technology (ICAIT), pp. 406–410, Chikmagalur, India, July 2019.
[16] J. Li, M. Liu, Z. Xue, X. Fan, and X. He, “RTVD: a real-time volumetric detection scheme for DDoS in the internet of things,” IEEE Access, vol. 8, pp. 36191–36201, 2020.

[17] K. Yang, J. Zhang, Y. Xu, and J. Chao, “Ddos attacks detection with autoencoder,” in NOMS 2020-2020 IEEE/IIFIP network operations and management symposium, pp. 1–9, Budapest, Hungary, April 2020.

[18] M. Saharkhizan, A. Azmoodeh, A. Dehghantanha, K. K. Choo, and R. M. Parizi, “An ensemble of deep recurrent neural networks for detecting IoT cyber attacks using network traffic,” IEEE Internet of Things Journal, vol. 7, no. 9, pp. 8852–8859, 2020.

[19] G. D. L. T. Parra, P. Rad, K.-K. R. Choo, and N. Beebe, “Detecting internet of things attacks using distributed deep learning,” Journal of Network and Computer Applications, vol. 163, article 102662, 2020.

[20] S. Haider, A. Akhunzada, G. Ahmed, and M. Raza, “Deep learning based ensemble convolutional neural network solution for distributed denial of service detection in SDNs,” in 2019 UK/China emerging technologies (UCET), pp. 1–4, Glasgow, UK, August 2019.

[21] 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, 2020.

[22] A. A. Ahmed, W. A. Jabbar, A. S. Sadiq, and H. Patel, “Deep learning-based classification model for botnet attack detection,” Journal of Ambient Intelligence and Humanized Computing, vol. 13, no. 7, pp. 3457–3466, 2022.

[23] J. Lam and R. Abbas, “Machine learning based anomaly detection for 5g networks,” 2020, https://arxiv.org/abs/2003.03474.

[24] S. Vellangiri and H. M. Pandey, “Fuzzy-Taylor-elephant herd optimization inspired deep belief network for DDoS attack detection and comparison with state-of-the-arts algorithms,” Future Generation Computer Systems, vol. 110, pp. 80–90, 2020.

[25] S. Agarwal, A. Tyagi, and G. Usha, “A deep neural network strategy to distinguish and avoid cyber-attacks,” in Artificial Intelligence and Evolutionary Computations in Engineering Systems, pp. 673–681, Springer, Singapore, 2020.

[26] 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.

[27] X. Yuan, C. Li, and X. Li, “DeepDefense: identifying DDoS attack via deep learning,” in 2017 IEEE international conference on smart computing (SMARTCOMP), pp. 1–8, Hong Kong, China, May 2017.

[28] M. Asad, M. Asim, T. Jawed, M. O. Beg, H. Mujtaba, and S. Abbas, “Deepdetect: detection of distributed denial of service attacks using deep learning,” The Computer Journal, vol. 63, no. 7, pp. 983–994, 2020.

[29] N. G. Bhuvaneswari Amma and S. Selvakumar, “Deep radial intelligence with cumulative incarnation approach for detecting denial of service attacks,” Neurocomputing, vol. 340, pp. 294–308, 2019.

[30] Y. Meidan, M. Bohadana, Y. Mathov et al., “N-bait—network-based detection of iot botnet attacks using deep autoencoders,” IEEE Pervasive Computing, vol. 17, no. 3, pp. 12–22, 2018.

[31] S. Yadav and S. Subramanian, “Detection of application layer DDoS attack by feature learning using stacked autoencoder,” in 2016 international conference on computational techniques in information and communication technologies (ictict), pp. 361–366, New Delhi, India, March 2016.

[32] G. Ke, R. S. Chen, Y. C. Chen, N. Xiong, Y. Tian, and N. Al-Nabhani, DDoS Attack Detection Method under Deep Learning in the Context of SDN-OpenFlow.

[33] S. Usbeck, Z. Turgut, and M. A. Aydin, “Cyber attack detection by using neural network approaches: shallow neural network, deep neural network and autoencoder,” in International conference on computer networks, pp. 144–155, Springer, Cham, 2019.

[34] S. Sumathi, R. Rajesh, and N. Karthikeyan, “DDoS attack detection using hybrid machine learning based IDS models,” Journal of Scientific & Industrial Research, vol. 81, pp. 276–286, 2022.

[35] M. V. Assis, L. F. Carvalho, J. Lloret, and M. L. Proença Jr., “A GRU deep learning system against attacks in software defined networks,” Journal of Network and Computer Applications, vol. 177, article 102942, 2021.

[36] M. P. Novaes, L. F. Carvalho, J. Lloret, and M. L. Proença Jr., “Adversarial deep learning approach detection and defense against DDoS attacks in SDN environments,” Future Generation Computer Systems, vol. 125, no. 125, pp. 156–167, 2021.

[37] Y. Mirsky, T. Doitshman, Y. Elovici, and A. Shabtai, “Kitsune: an ensemble of autoencoders for online network intrusion detection,” 2018, https://arxiv.org/abs/1802.09089.

[38] G. Bovenzi, G. Aceto, D. Ciouzno, V. Persico, and A. Pescapé, “A hierarchical intrusion detection approach in IoT scenarios,” in GLOBECOM 2020 - 2020 IEEE global communications conference, pp. 1–7, Taipei, Taiwan, December 2020.

[39] A. A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja, and H. Chen, “Harris hawks optimization: algorithm and applications,” Future Generation Computer Systems, vol. 97, pp. 849–872, 2019.

[40] S. K. Fan and C. H. Jen, "An enhanced partial search to particle swarm optimization for unconstrained optimization," Mathematics, vol. 7, no. 4, p. 357, 2019.

[41] D. Wang, D. Tan, and L. Lui, “Particle swarm optimization algorithm: an overview,” Soft Computing, vol. 22, no. 2, pp. 387–408, 2018.

[42] S. Mukherjee and N. Sharma, “Intrusion detection using naive Bayes classifier with feature reduction,” Procedia Technology, vol. 4, pp. 119–128, 2012.

[43] P. I. Radoglou-Grammatikis and P. G. Sarigiannis, “An anomaly-based intrusion detection system for the smart grid based on cart decision tree,” in 2018 global information infrastructure and networking symposium (GIIS), pp. 1–5, Thessaloniki, Greece, October 2018.

[44] U. Ali, K. K. Dewangan, and D. K. Dewangan, “Distributed denial of service attack detection using ant bee colony and artificial neural network in cloud computing,” in Nature inspired computing, pp. 165–175, Springer, Singapore, 2018.

[45] O. Ali and P. Cotae, “Towards DoS/DDoS attack detection using artificial neural networks,” in 2018 9th IEEE annual ubiquitous computing, Electronics & Mobile Communication Conference (UEMCON), pp. 229–234, New York, NY, USA, November 2018.

[46] Z. Chiba, N. Abghour, K. Moussaid, A. El Omri, and M. Rida, “New anomaly network intrusion detection system in cloud environment based on optimized back propagation neural network using improved genetic algorithm,” International
[47] Z. Liu, Y. He, W. Wang, and B. Zhang, “DDoS attack detection scheme based on entropy and PSO-BP neural network in SDN,” China Communications, vol. 16, no. 7, pp. 144–155, 2019.

[48] K. J. Singh and T. De, “MLP-GA based algorithm to detect application layer DDoS attack,” Journal of Information Security and Applications, vol. 36, no. 36, pp. 145–153, 2017.

[49] T. T. Teoh, Y. Y. Nguwi, Y. Elovici, W. L. Ng, and S. Y. Thiang, “Analyst intuition inspired neural network based cyber security anomaly detection,” International Journal of Innovative Computing, Information and Control, vol. 14, no. 1, pp. 379–386, 2018.

[50] X. Tang, M. Chen, J. Cheng, J. Xu, and H. Li, “A security situation assessment method based on neural network,” in International Symposium on Cyberspace Safety and Security, pp. 579–587, Springer, Cham, 2019.

[51] K. Sinha, A. Viswanathan, and J. Bunn, “Tracking temporal evolution of network activity for botnet detection,” 2019, https://arxiv.org/abs/1908.03443.

[52] M. Anjum and K. S. Shreedhara, “Performance analysis of semi-supervised machine learning approach for DDoS detection,” International Journal of Innovative Research in Technology, vol. 6, no. 2, pp. 144–147, 2019.

[53] R. V. de Miranda, P. R. Inácio, D. Magoni, and M. M. Freire, “Detection of reduction-of-quality DDoS attacks using fuzzy logic and machine learning algorithms,” Computer Networks, vol. 186, article 107792, 2021.

[54] S. M. Hosseini, “Bi-objective web service composition problem in multi-cloud environment: a bi-objective time-varying particle swarm optimisation algorithm,” Journal of Experimental & Theoretical Artificial Intelligence, vol. 33, no. 2, pp. 179–202, 2021.

[55] R. V. Kulkarni and G. K. Venayagamoorthy, “Neural network based secure media access control protocol for wireless sensor networks,” in 2009 International Joint Conference on Neural Networks, pp. 1680–1687, Atlanta, GA, USA, June 2009.

[56] M. Bukhsh, S. Abdullah, A. Rahman, M. N. Asghar, H. Arshad, and A. Alabdulatif, “An energy-aware, highly available, and fault-tolerant method for reliable IoT systems,” IEEE Access, vol. 9, pp. 145363–145381, 2021.

[57] W. F. Josephen, H. L. Warnars, E. Abdurrahman, P. Assiroj, A. I. Kistijantoro, and A. Doucet, “Dragonfly algorithm in 2020,” Communications in Mathematical Biology and Neuroscience, vol. 2021, 2021.

[58] T. G. Dietterich, “Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms,” Neural Computation, vol. 10, no. 7, pp. 1895–1923, 1998.

[59] A. Nascita, A. Montieri, G. Aceto, D. Ciouonzo, V. Persico, and A. Pescapé, “XAI meets mobile traffic classification: understanding and improving multimodal deep learning architectures,” IEEE Transactions on Network and Service Management, vol. 18, no. 4, pp. 4225–4246, 2021.