mathematics

Article

Design of Network Intrusion Detection System Using Lion Optimization-Based Feature Selection with Deep Learning Model

Rayed AlGhamdi

Department of Information Technology, Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah 21589, Saudi Arabia; raalghamdi8@kau.edu.sa

Abstract: In the domain of network security, intrusion detection systems (IDSs) play a vital role in data security. While the utilization of the internet amongst consumers is increasing on a daily basis, the significance of security and privacy preservation of system alerts, due to malicious actions, is also increasing. IDS is a widely executed system that protects computer networks from attacks. For the identification of unknown attacks and anomalies, several Machine Learning (ML) approaches such as Neural Networks (NNs) are explored. However, in real-world applications, the classification performances of these approaches are fluctuant with distinct databases. The major reason for this drawback is the presence of some ineffective or redundant features. So, the current study proposes the Network Intrusion Detection System using a Lion Optimization Feature Selection with a Deep Learning (NIDS-LOFSDL) approach to remedy the aforementioned issue. The NIDS-LOFSDL technique follows the concept of FS with a hyperparameter-tuned DL model for the recognition of intrusions. For the purpose of FS, the NIDS-LOFSDL method uses the LOFS technique, which helps in improving the classification results. Furthermore, the attention-based bi-directional long short-term memory (ABiLSTM) system is applied for intrusion detection. In order to enhance the intrusion detection performance of the ABiLSTM algorithm, the gorilla troops optimizer (GTO) is deployed so as to perform hyperparameter tuning. Since trial-and-error manual hyperparameter tuning is a tedious process, the GTO-based hyperparameter tuning process is performed, which demonstrates the novelty of the work. In order to validate the enhanced solution of the NIDS-LOFSDL system in terms of intrusion detection, a comprehensive range of experiments was performed. The simulation values confirm the promising results of the NIDS-LOFSDL system compared to existing DL methodologies, with a maximum accuracy of 96.88% and 96.92% on UNSW-NB15 and AWID datasets, respectively.

Keywords: network intrusion detection system; network security; lion optimization algorithm; feature selection; deep learning

MSC: 68-11

check for updates

Citation: AlGhamdi, R. Design of Network Intrusion Detection System Using Lion Optimization-Based Feature Selection with Deep Learning Model. *Mathematics* **2023**, *11*, 4607. https://doi.org/10.3390/math11224607

Academic Editors: Cheng-Chi Lee and Antanas Cenys

Received: 23 September 2023
Revised: 22 October 2023
Accepted: 3 November 2023
Published: 10 November 2023

1. Introduction

Network security is the most interesting aspect that is responsible for the emergence of internet applications. However, the number of cyberattacks has also increased on the internet in the past decade. Therefore, it is essential to develop new approaches that can effectively detect and prevent such attacks. This can be achieved by developing novel techniques for intrusion detection [1]. In order to avoid these cyberattacks, key management, access control, and intrusion detection systems (IDS) are necessary [2]. Amongst these, IDS is the most commonly used system for ensuring network security. Presently, cyberattacks pose a major issue for system and network security in the form of Denial of Service (DoS) attacks, computer viruses, and data breaches [3]. To mitigate this problem, the IDSs are frequently employed in different organizations. According to the identification techniques, these detection methods are categorized as signature-based or misuse-based NIDS and anomaly-based NIDS. The objective of the current research is to determine the anomalies by recognizing a clear abnormality between the existing actions and predetermined normal actions, utilized for representing a normal activity or normal connection [4,5]. Automatically, the anomaly-based detection techniques exhibit the ability to identify new (or 0-day) attacks, whereas the misuse-based detection systems identify only the known attacks [6].

The researchers established IDSs for various platforms depending on the security issues of diverse networks. The operations of the IDS involve data collection to analyze every potential security attack from various fields within a network or computer [7]. In recent years, intrusion detection and other security technologies, namely, firewalls, cryptography, and authentication, have significantly improved. Machine Learning (ML) is the main assistant of Artificial Intelligence (AI) technology [8]. It enables the creation of computers that can perform without precise programming. In these computers, the ML techniques can perform the execution of tasks depending on generalized data or samples. This characteristic helps these computers in improving themselves by learning from the available information [9]. The ML technique is capable of identifying unknown attacks in network traffic, thus sharing its ability to identify other types of attacks trained on rare and general types of traffic. However, the effectiveness of the ML approaches does not remain consistent when using various types of datasets, because of the presence of higher-dimensional data [10]. For instance, redundant or inefficient features can increase the computational period and reduce the identification outcome. In this context, Feature Selection (FS) is a better approach to mitigate this problem.

For intrusion detection, FS and Deep Learning (DL) techniques are applied. FS is highly needed nowadays, owing to the presence of numerous attributes of the network data, which are repetitive and unrelated. With the application of the FS technique, the detection model can focus primarily on highly useful features, reduce the dimensionality, enhance the model's interpretability, and increase the detection accuracy. On the other hand, the DL technique can proficiently learn complex patterns and temporal dependencies from the network traffic data. The DL models can learn intricate intrusion patterns that may be challenging for traditional rule-based or statistical approaches to discern. By combining the FS's data pre-processing capabilities with DL's pattern recognition prowess, the network intrusion detection process can be significantly fortified. This outcome enables the timely and accurate identification of both known and novel cyber threats in the ever-evolving landscape of network security.

The current study proposes a Network Intrusion Detection System using a Lion Optimization Feature Selection with Deep Learning (NIDS-LOFSDL) model. The NIDS-LOFSDL technique uses the LOFS technique, which aids in improving the classification performance. Furthermore, the study also used the attention-based bi-directional long short-term memory (ABiLSTM) system for intrusion detection. In order to enhance the intrusion detection performances of the ABiLSTM methodology, the gorilla troops optimizer (GTO) is deployed for hyperparameter tuning. For validating the enhanced solution of the NIDS-LOFSDL technique for intrusion detection, a comprehensive range of experiments was conducted. The key contributions of the study are summarized herewith:

- A new NIDS-LOFSDL technique has been developed in this study, comprising LOFS, ABiLSTM classifier, and GTO-based parameter tuning approaches for network intrusion detection. The rationale behind combining the FS and hyperparameter-tuned DL model is to enhance the accuracy and efficiency of the intrusion detection systems;
- LOFS has been incorporated as an FS method that selects the most relevant and informative features from a dataset. This characteristic helps in improving the accuracy and interpretability of the intrusion detection models;
- The ABiLSTM network has been deployed for intrusion detection, and the model is known for its ability to capture temporal dependencies in sequential data, thus making it an appropriate choice for the detection of complex intrusion patterns in network traffic;

- To further enhance the performance of the ABiLSTM algorithm, the study used GTO for hyperparameter tuning. It intends to determine the optimal hyperparameter configuration for the ABiLSTM model to accomplish an enhanced detection performance.

2. Related Works

In the literature [11], a new anomaly-based IDS technique has been deployed for IoT networks, utilizing the DL method. In particular, a filter-based FS-DNN technique was introduced in the study, in which extremely correlated features were dropped. Additionally, this technique was tuned with several parameters and hyperparameters. Mohy-eddine et al. [12] developed an NIDS for IoT platforms by employing FS and KNN methods. The authors created the NIDS with the help of the K-NN method to enhance the IDS Detection Rate (DR) and accuracy (ACC). Also, the GA, univariate statistical test, and PCA were utilized for the FS technique individually to increase the quality of the data and select ten better effective features. In the study conducted earlier [13], a hybrid DL technique and shallow learning method were presented for identifying the intrusions in the IoT devices. In the developed method, the spider monkey optimization FS method was primarily employed to select a greater number of relevant features. Secondarily, a Siamese NN-based approach was presented for making the data highly classifiable. Syed et al. [14] suggested a new fog-cloud-based IoT-IDS that integrates a distributed process by separating the database based on the type of attacks and an FS stage on time-series IoT data. Then, a DL-Recurrent-NN (Simple RNN and BiLSTM) was used to identify the attacks.

In the literature [15], a network intrusion detection classification (NIDS-CNNLSTM) technique was presented based on DL. This technique was designed for the wireless sensing environment of the Industrial IoT (IIoT) for the efficient differentiation and detection of network traffic data, and to ensure the safety of the equipment and the functioning of the IIoT. The NIDS-CNNLSTM technique integrated the robust learning capability of LSTM-NNs in time series data, and classified and learnt the FS utilizing CNN. Further, the efficiency was also confirmed based on multi-classification and binary classification methods. Ravi et al. [16] recommended an endwise system for network attack classification and identification by DL-based recurrent approaches. This method extracts the features of v-layers present in the recurrent algorithms and uses a kernel-based PCA (KPCA)-FS technique for the detection of the optimum features. Lastly, the optimum features of the recurrent methods were incorporated, and classification was executed using an ensemble meta-classifier.

Atefinia and Ahmadi [17] introduced a multi-architectural integrated DNN technique to reduce the false positive rate of anomaly-based IDSs. This approach contains a feed-forward method, a stack of limited Boltzmann machine methods and two recurrent methods. The output weights of these methods were input into an aggregator method to generate the solution for these models. In the literature [18], the authors introduced an efficient network IDS based on Random Forest (RF) and Sparse-AE (SAE) to alleviate the issue. The extraction feature ability of the SAE and identification and classification potential of the RF were integrated to enhance the identification accuracy and performance. The SAE-RF identification technique was developed.

With an increase in network-based threats and sophisticated intrusion techniques, the requirement for highly robust and adaptive intrusion detection systems has grown exponentially. A major research gap in the field of network intrusion detection lies in the requirement for efficient models to perform FS and hyperparameter selection. Though considerable developments have been made in ML and DL models to detect intrusions, the intricate and high-dimensional nature of the network data continue to pose challenges. The existing approaches find it challenging to deal with feature redundancy, irrelevant attributes, and suboptimal hyperparameter configurations, thus resulting in low detection performance. So, it is now necessary to design new models for the effective selection of relevant features and the fine-tuning of the model hyperparameters to adapt to the dynamic and evolving nature of network threats.

3. The Proposed Model

In the current study, a novel NIDS-LOFSDL approach has been established for intrusion recognition so as to accomplish network security. The NIDS-LOFSDL technique follows the concept of FS with a hyperparameter-tuned DL algorithm for the recognition of the intrusions. The proposed model encompasses LOFS, ABiLSTM-based detection, and GTO-based hyperparameter tuning. Figure 1 exhibits the entire procedure of the NIDS-LOFSDL methodology.

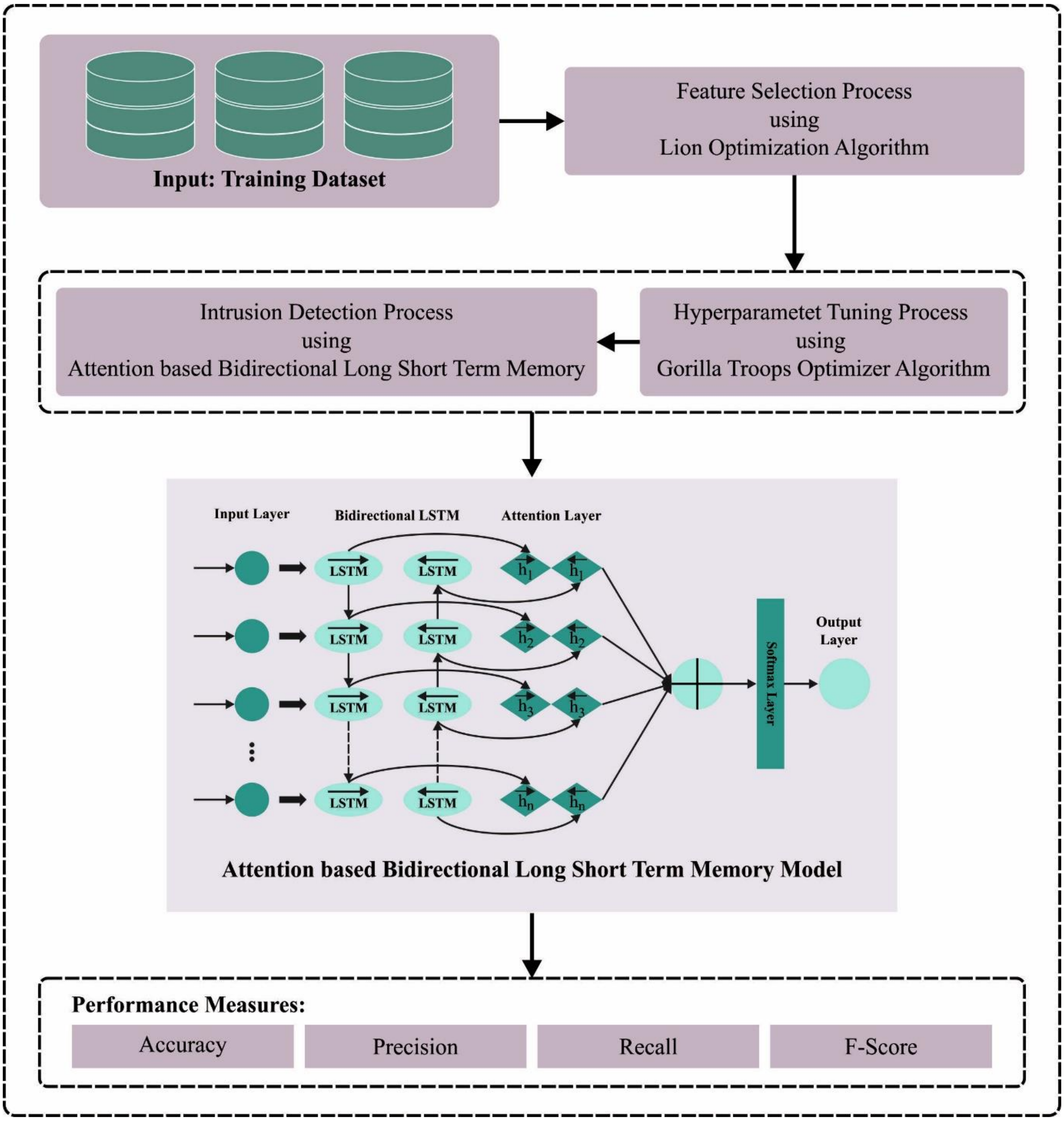
Input: Training Dataset
Feature Selection Process using Lion Optimization Algorithm
Intrusion Detection Process using Attention based Bidirectional Long Short Term Memory
Hyperparametet Tuning Process using Gorilla Troops Optimizer Algorithm
Input Layer
Bidirectional LSTM
Attention Layer
LSTM
h1
h2
h3
hn
Softmax Layer
Output Layer
Attention based Bidirectional Long Short Term Memory Model
Performance Measures:
Accuracy
Precision
Recall
F-Score

Figure 1. Overall process of the NIDS-LOFSDL algorithm.

3.1. Feature Selection Using the LOFS Approach

For the feature selection process, the LOFS approach is used. The LO algorithm is a population-based algorithm in which the lemurs set is mathematically modeled as follows [19].

$$X = \begin{bmatrix} l_1^1 & l_1^2 & \cdot & l_1^d \\ l_2^1 & l_2^2 & \cdot & l_2^d \\ \vdots & \vdots & \vdots & \vdots \\ l_n^1 & l_n^2 & \cdot & l_n^d \end{bmatrix}, \tag{1}$$

where n stands for the solution candidate and d indicates the decision variable. X shows the matrix in $n \times d$ size. Figure 2 illustrates the flowchart of the LO algorithm. The steps contained in the LOFS approach are given below.

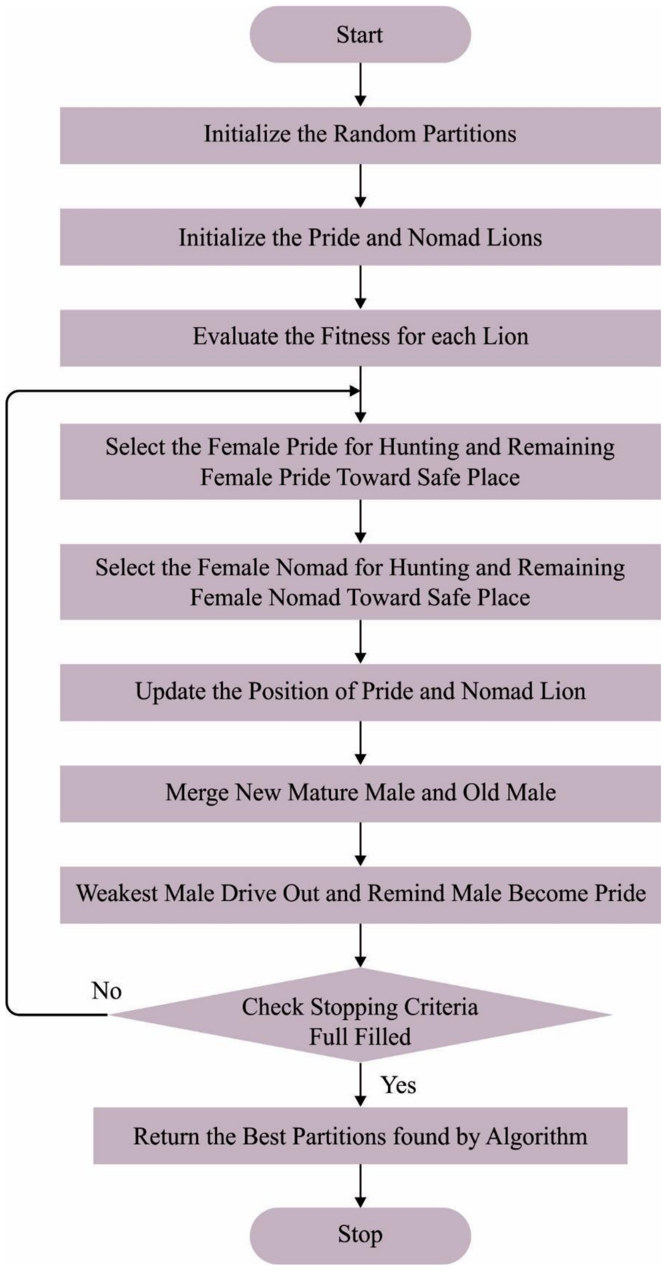
Start
Initialize the Random Partitions
Initialize the Pride and Nomad Lions
Evaluate the Fitness for each Lion
Select the Female Pride for Hunting and Remaining Female Pride Toward Safe Place
Select the Female Nomad for Hunting and Remaining Female Nomad Toward Safe Place
Update the Position of Pride and Nomad Lion
Merge New Mature Male and Old Male
Weakest Male Drive Out and Remind Male Become Pride
No
Check Stopping Criteria Full Filled
Yes
Return the Best Partitions found by Algorithm
Stop

Figure 2. Flowchart of the LO algorithm.

Step 1: Define the parameter N Population when Max_{iter} represents the maximum iteration count. d corresponds to the dimensionality of the searching region over the dataset size. In addition, UB and LB indicate the upper and lower boundaries of the problem, respectively.

Step 2: Produce X decision parameters in the ith solution, according to Equation (2)

$$X_i^j = \left(LB + \left(UB_j - LB_j\right)\right) \times r, \tag{2}$$

where r implies the uniformly distributed random integer $\in [0,1]$.

Step 3: Inside the loop for all the iterations, evaluate the Free Risk Rate (FRR), a co-efficient of LO,

$$FRR = HRR - t \times \frac{(HRR - LRR)}{\text{Max}_{iter}}, \tag{3}$$

In Equation (3), t indicates the existing iteration counter. Max_{iter} shows the size of the iteration. Low-Risk Rate (LRR) and High-Risk Rate (HRR) are two constant and predefined values.

Step 4: Compute the fitness values for x_i^j, as given below.

$$Fit\ (x_i^j) = \alpha \times (1 - Acc) + \beta \times \left(\frac{s}{S}\right), \tag{4}$$

In Equation (4), Acc represents the accuracy of the subset that can be extracted by the ABiLSTM classification function in order to assess the selected subset in all the iterations. $Fit\ (x_i^j)$ denotes the fitness values, s implies the number of features selected and S suggests the maximal number of features selected.

Step 5: Lemurs are categorized into two dissimilar processes to increase their fitness values. Initially, the best near lemurs (bnl) are recognized, which implies the selection of the solution with a low fitness values. According to the FS objective, bnl provides a better feature for the existing iteration. Then, the global best lemur (gbl) is selected in the whole population, which represents the total optimum solution.

Step 6: Set the value of r_1, a randomly generated value, to $\in [0,1]$, and compare it with FRR. Later, the location is updated for the lemur, far from the risk-based position, according to Equation (5).

$$X_i^j = \begin{cases} x(i,\ j) + |(x(i,\ j) - x(bnl,\ j)| \times (r_3 - 0.5) \times 2; & r_1 < FRR \\ x(i,\ j) + |(x(i,\ j) - x(gbl,\ j)| \times (r_3 - 0.5) \times 2; & r_1 > FRR \end{cases}, \tag{5}$$

where r_1 refers to the random integer $\in [0,1]$. The present ith lemur of the Nth population is (i, j), i.e., the solution candidate at the jth dimension.

The LO process begins by arbitrarily generating a swarm of lemurs. Next, it tries to move towards the lemurs with low fitness value by dance hup. The optimization process randomly generates a group of lemurs. The FRR value begins towards the LRR, thus representing that the lemur starts with the move and moves near to the bnl through "dance hup". The purpose of LO, implementing these dance hup actions, is to decrease the value of FRR near to the HRR. Next, it exploits the leap-up action to move the lemur towards the global optimal performance. This process is repeated until the ending condition is met.

3.2. Intrusion Detection Using ABiLSTM Model

To detect the presence of the intrusions, the ABiLSTM model is applied. LSTM is a revised edition of the classical RNN that exploits the specially adapted memory units to effectively express the long-term dependency of the MTS dataset [20]. The LSTM model's design provides an effective solution to the gradient disappearing problem on the contrary to the traditional RNN methods. According to the present input and the previous state of the hidden units, the LSTM cell learns about the existing state of the hidden unit. Nevertheless, it replaces the structure of the hidden unit with a memory cell that corresponds to the long-term dependency of the MTS signal. The LSTM model includes four controlled gates, such as one self-loop memory cell, one input, one output, and one forget, for manipulating the interaction of the data stream among different memory neurons. In the hidden layer of the LSTM model, the forget gate is used to determine the data that need to be ignored or preserved from the prior moment. Simultaneously, the entrance of the input neuron decides whether the input signal needs to be injected with the information of the memory unit. The output neuron gate decides whether the state of the memory unit should be changed or not. Consider the input x_t of MTS and the dynamic output state h_t; the neuron state, output of HL, and gate states are calculated using the subsequent formula.

$$i_t = \sigma(U_i x_t + W_i h_{t-1} + b_i), \tag{6}$$

$$f_t = \sigma(U_f x_t + W_f h_{t-1} + b_f), \tag{7}$$

$$o_t = \sigma(U_o x_t + W_o h_{t-1} + b_o), \tag{8}$$

$$\tilde{C}_t = tanh(U_c x_t + W_c h_{t-1} + b_c), \tag{9}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{C}_t, \tag{10}$$

$$h_t = o_t \odot tanh(c_t). \tag{11}$$

The recurrent weight matrices are represented as W_i, W_f, W_o, and W_c, while the weighted matrix for the input, forget, output and memory cell gates are denoted by U_i, U_f, U_o, and U_c, correspondingly. The gates bias is formulated by b_i, b_f, b_o, and b_c. The cell state of the candidate $\tilde{C}_t$ is used to update the original memory cell state, c_t. At any time step, h_t represents the state of HL and o_t denotes the output. The symbol $\odot$ denotes the element-wise multiplication operation. $tanh$ denotes the hyperbolic tangent function and σ shows the logistic sigmoid activation function.

The classical LSTM model may inadvertently discard the sequential information at the time of training as it processes only the input signals in one direction. Therefore, the time series data cannot be completely reviewed. In order to over this limitation, the BLSTM was developed with a bidirectional structure to capture the representation of MTS information via forward and backward directions. The BLSTM comprises two LSTM layers that are carried out in parallel but opposite directions. In the case of the reverse propagation direction, $h_{b(t)}$ represents the hidden layer, which comprises data from the future MTS values. In forward propagation, $h_{f(t)}$ denotes the data of the hidden LSTM neuron, and it retains the data from the prior sequence value. Both $h_{f(t)}$ and $h_{b(t)}$ are connected to each other for creating the final output of the BiLSTM model. The tth hidden layer of BLSTM for forward and backward states is computed using Equations (12) and (13):

$$h_{f(t)} = \psi(W_{fh} x_t + W_{bhh} h_{f(t-1)} + b_{fb}), \tag{12}$$

$$h_{b(t)=}\psi(W_{bh} x_t + W_{bhh} h_{b(t+1)} + b_b). \tag{13}$$

In addition to these, b_{fb} and b_b correspond to biased data in two directions. The weight matrices W_{fh} and W_{bh} represent the forward and backward synapsis weight from the input to the internal unit weight. Likewise, W_{fhh} and W_{bhh} represent the forward and backward feedback recurrent weights.

tanh indicates the activation function of the HLs ψ. Using this component, the output of BiLSTM y_t is defined herewith.

$$y_t = \sigma(W_{fhy} h_{f(t)} + W_{bhy} h_{b(t)} + b_y), \tag{14}$$

In Equation (16), the forward and backward weights of the resultant layers are denoted by W_{jhy} and W_{bhy}, correspondingly. The activation function of the resultant layer σ is either given as a linear function or sigmoidal function. Further, b_y represents the output bias.

In ABiLSTM, when the attention mechanism is utilized, it supports the model in learning by assigning various weights. For an HL h_i, its attention a_i is expressed as in Equation (15).

$$\begin{aligned} u_i &= \tanh(W \cdot h_i + b), \\ a_i &= \frac{e^{u_i^T \cdot u_w}}{\Sigma_i e^{u_i^T \cdot u_w}}, \end{aligned} \tag{15}$$

whereas W signifies the weighted matrix, b implies the bias, and u_w represents the global context vector, and all three are learned in the training method.

3.3. Hyperparameter Tuning Using GTO Algorithm

Eventually, the hyperparameter values of the ABiLSTM methodology are chosen using the GTO algorithm. The GTO approach is one of the main metaheuristic optimization approaches, inspired by the intelligent behaviors of gorillas [21]. These behaviors are explained using five major operators, as follows. Two of the operators represent the exploitation stage, whereas the other three operators define the exploration stage. The three operators are sometimes described as strategies or the exploration stage, and they can be inferred from the movement to another gorilla, migration towards an unknown place, and migration towards a known place. As mentioned before, the exploitation stage uses two operators reflected by the competition for adult females, and follows the behavior of the silverback. The competition is initiated between the adult females in such a way that they follow the silverback.

Using the following equations, the three prior approaches of the exploration stage are defined.

$$GX(t+1) = \begin{cases} (UB - LB) \times r_1 + LB \\ (r_2 - C) \times X_r(t) + L \times H \\ X(i) - L \times (L \times (X(t) - GX_r(t))0, \\ +r_3 \times (X(t) - GX_r(t))) \end{cases} \tag{16}$$

$$C = F \times \left(1 - \frac{it}{\mathrm{Maxs}It}\right), \tag{17}$$

$$F = \cos(2 \times r_4) + 1, \tag{18}$$

$$L = C \times l\,, \tag{19}$$

$$H = Z \times X(t), \tag{20}$$

$$Z = [-C,\, C]\,, \tag{21}$$

In this equation, the upper as well as lower boundaries are denoted using UB and LB, respectively. Using $X(it+1)$, the position selected is defined in the iteration (it), whereas the existing location is represented as X (it). Max$_it$ is known by the maximum number of iterations. The parameter p defines the probability of the migration that lies in the range of 0 to 1. Lastly, the exploration stage ends by enabling the outcome $GX(it)$ to exchange $X(it)$, and these solutions are known if the silverback arises, when $X(it)$ is greater than $GX(it)$.

Using Equations (18)–(24), the following competition strategies are defined.

$$X(t+1) = L \times M \times (X(t) - X_{silverback}) + X(t), \tag{22}$$

$$M = \left(\left| \frac{1}{N} \sum_{i=1}^{N} GX_i(t) \right|^{g} \right)^{\frac{1}{g}}, \tag{23}$$

$$g = 2^L, \tag{24}$$

$$GX(i) = X_{silverback} - (X_{silverback} \times Q - X(t) \times Q) \times A, \tag{25}$$

$$Q = 2 \times r_5 - 1, \tag{26}$$

$$A = \beta \times E\,, \tag{27}$$

$$E = \begin{cases} N_1 \; rand \geq 0.5 \\ N_2 \; rand < 0.5' \end{cases} \tag{28}$$

The GTO approach develops the following FF to make the best classification solutions. It defines a positive integer to denote the good solution of the candidate's performance. In this case, the reduction in the classification errors is supposed to be the FF.

$$\begin{aligned} fitness(x_i) &= ClassifierErrorRate(x_i) \\ &= \frac{No.\ of\ misclassified\ instances}{Total\ no.\ of\ instances} * 100, \end{aligned} \tag{29}$$

4. Results and Discussion

The proposed model was simulated in the Python 3.8.5 tool configured on a PC with specifications of i5-8600k, GeForce 1050Ti 4 GB, 16 GB RAM, 250 GB SSD, and 1 TB HDD.

The ID detection outcomes of the NIDS-LOFSDL methodology were validated using two benchmark datasets, the UNSW-NB15 [22] and AWID [23]. Table 1 shows the details of both datasets.

Table 1. Details of two datasets.

UNSW-NB15 Dataset	
Class	No. of Instances
Normal	15,000
Attack	15,000
Total Instances	30,000
AWID Dataset	
Class	No. of Instances
Normal	15,000
Attack	15,000
Total Instances	30,000

Figure 3 establishes the classification performances of the NIDS-LOFSDL system on the UNSW-NB15 database. Figure 3a,b demonstrate the confusion matrices produced by the NIDS-LOFSDL methodology on the 60:40 TR set/TS set. The outcome values show that the NIDS-LOFSDL system detected and classified both the classes accurately. Afterwards, Figure 3c reveals the PR outcomes of the NIDS-LOFSDL method. The simulation value infers that the NIDS-LOFSDL methodology attained the maximum PR values on both the classes. However, Figure 3d demonstrates the ROC outcomes of the NIDS-LOFSDL methodology. The outcomes show that the NIDS-LOFSDL approach led to a proficient solution with better ROC values on both the classes.

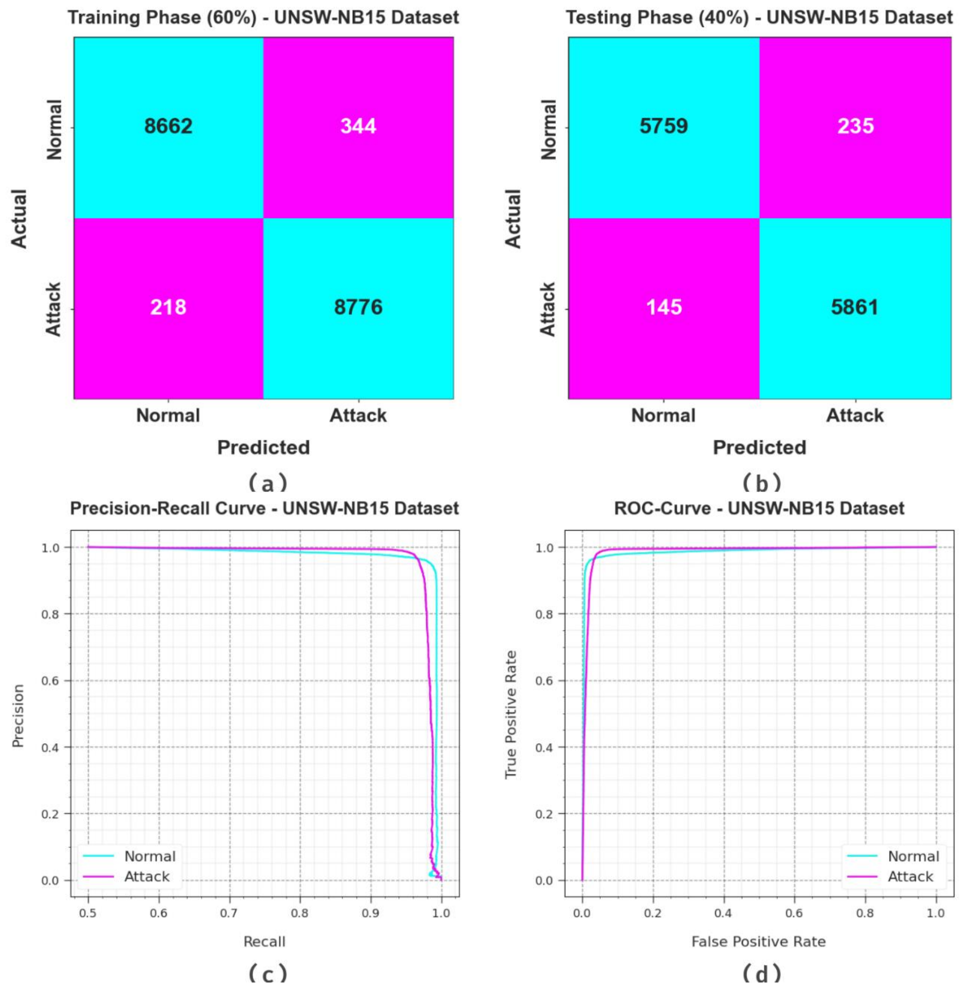

Figure 3. UNSW-NB15 dataset. (**a**,**b**) Confusion matrices, (**c**) PR_curve, and (**d**) ROC.

Table 2 and Figure 4 highlight the recognition outcomes of the NIDS-LOFSDL system upon the UNSW-NB15 database. The outcomes indicate the proficient recognition of normal and attack instances. With the 60% TR set, the NIDS-LOFSDL technique achieved an average $accu_y$ of 96.88%, a $prec_n$ of 96.89%, a $reca_l$ of 96.88%, and an F_{score} of 96.88%. Additionally, with the 40% TS set, the NIDS-LOFSDL algorithm attained an average $accu_y$ of 96.83%, a $prec_n$ of 96.84%, a $reca_l$ of 96.83% and an F_{score} of 96.83%.

Table 2. Recognition outcomes of the NIDS-LOFSDL technique applied to the UNSW-NB15 database.

Class	$Accu_y$	$Prec_n$	$Reca_l$	$F1_{Score}$
		TR set (60%)		
Normal	96.18	97.55	96.18	96.86
Attack	97.58	96.23	97.58	96.90
Average	96.88	96.89	96.88	96.88
		TS set (40%)		
Normal	96.08	97.54	96.08	96.81
Attack	97.59	96.15	97.59	96.86
Average	96.83	96.84	96.83	96.83

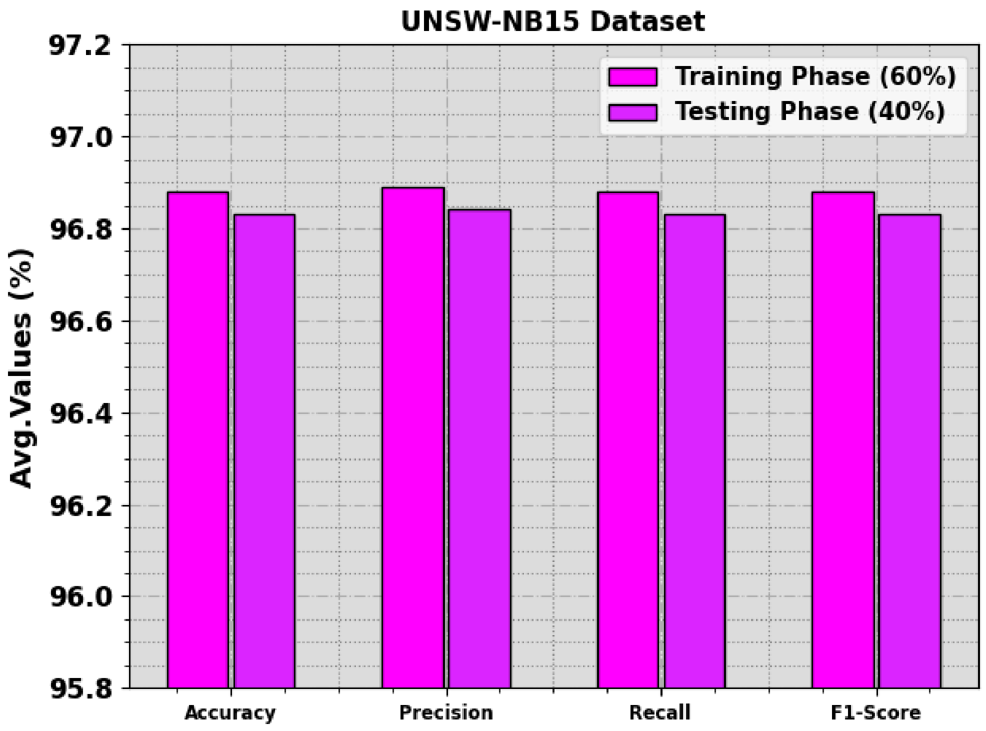

Figure 4. Average values of the NIDS-LOFSDL technique applied to the UNSW-NB15 database.

Figure 5 illustrates the training accuracy values, i.e., TR_accu_y and VL_accu_y, attained by the NIDS-LOFSDL technique on the UNSW-NB15 dataset. TL_accu_y is determined by evaluating the NIDS-LOFSDL method on the TR dataset, whereas the VL_accu_y value is computed by evaluating the outcomes on a separate testing dataset. The results imply that both the TR_accu_y and VL_accu_y values increased with an upsurge in the number of epochs. Accordingly, the performance of the NIDS-LOFSDL system is confirmed to achieve the maximum performance on both TR and TS datasets, with an increase in the number of epochs.

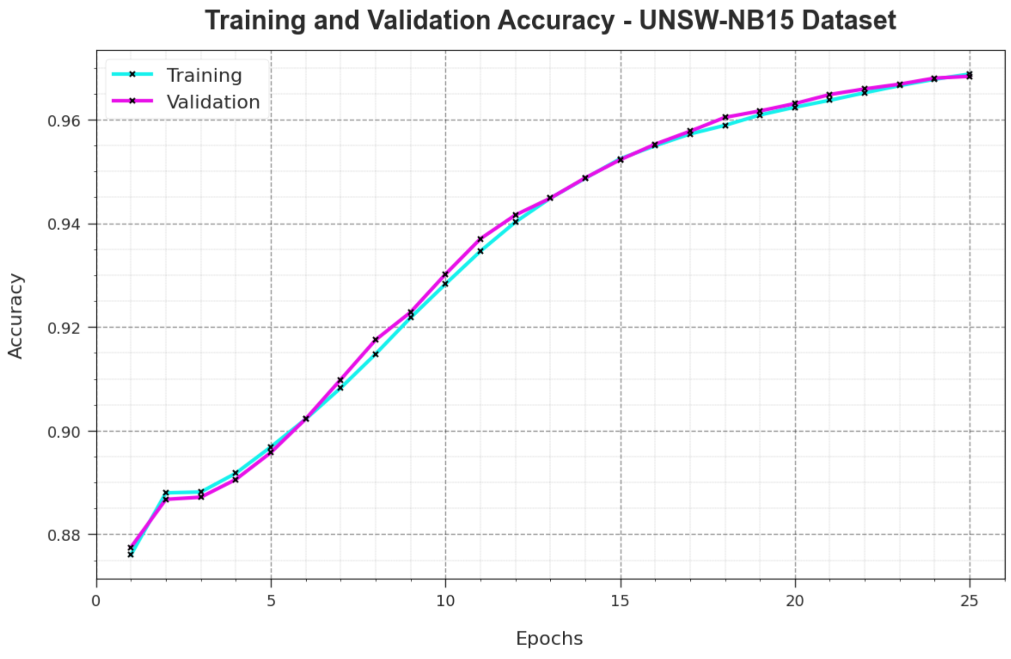

Figure 5. $Accu_y$ curve of the NIDS-LOFSDL technique on the UNSW-NB15 database.

In Figure 6, the TR_loss and VR_loss results of the NIDS-LOFSDL algorithm on the UNSW-NB15 dataset are revealed. The TR_loss corresponds to the error between the predictive performance and original values on the TR data. The VR_loss represents the performance evaluation of the NIDS-LOFSDL technique on individual validation data. The outcomes imply that both TR_loss and VR_loss values were reduced with an increase in the number of epochs. This scenario portrays the enhanced performance of the NIDS-LOFSDL approach and its ability to produce an accurate classification. The minimal TR_loss and VR_loss values demonstrate the enhanced performance of the NIDS-LOFSDL method in capturing the patterns and relationships.

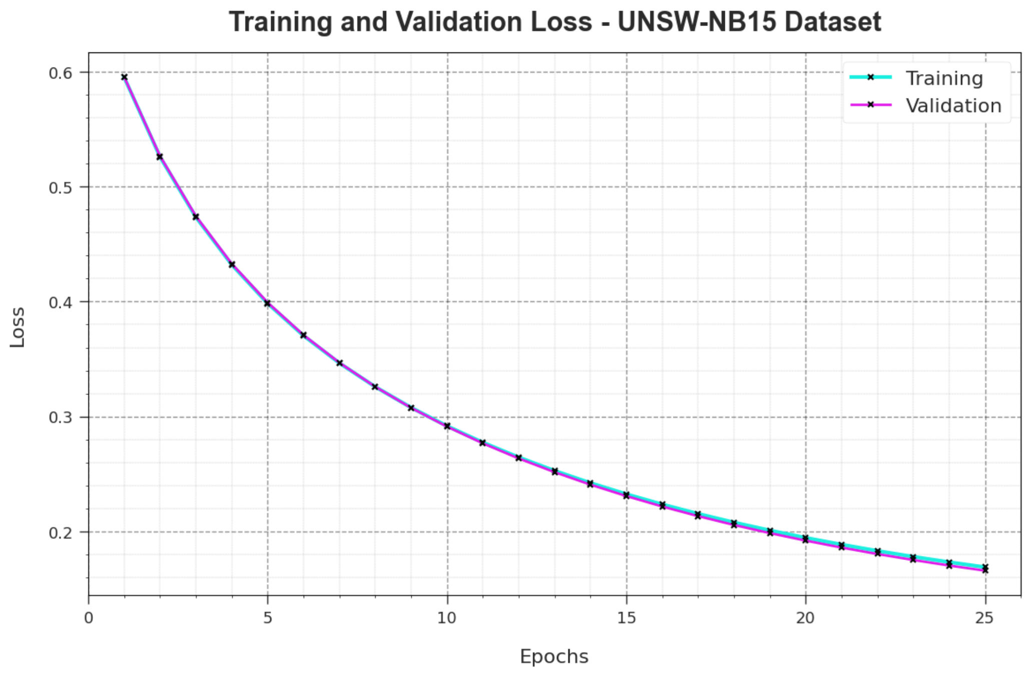

Figure 6. Loss curve of the NIDS-LOFSDL technique on the UNSW-NB15 database.

Figure 7 illustrates the classification outcomes of the NIDS-LOFSDL algorithm on the AWID database. Figure 7a,b exhibit the confusion matrices generated by the NIDS-LOFSDL methodology upon 60:40 of the TR set/TS set. The outcomes show that the NIDS-LOFSDL system outperformed all other techniques and detected and classified both the classes accurately. Then, Figure 7c depicts the PR outcomes of the NIDS-LOFSDL approach. The simulation value shows that the NIDS-LOFSDL system reached increased PR values on both the classes. Moreover, Figure 7d shows the ROC curve of the NIDS-LOFSDL methodology. The outcome values demonstrate the superior capability of the NIDS-LOFSDL algorithm with higher ROC values on both the classes.

Table 3 and Figure 8 demonstrate the recognition outcomes of the NIDS-LOFSDL methodology on the AWID database. The simulation value refers to the proficient recognition of both normal and attack samples. With the 60% TR set, the NIDS-LOFSDL system attained an average $accu_y$ of 96.92%, $prec_n$ of 96.92%, $reca_l$ of 96.92%, and an F_{score} of 96.92%. Then, with the 40% TS set, the NIDS-LOFSDL methodology accomplished an average $accu_y$ of 96.88%, $prec_n$ of 96.89%, $reca_l$ of 96.88%, and an F_{score} of 96.88%.

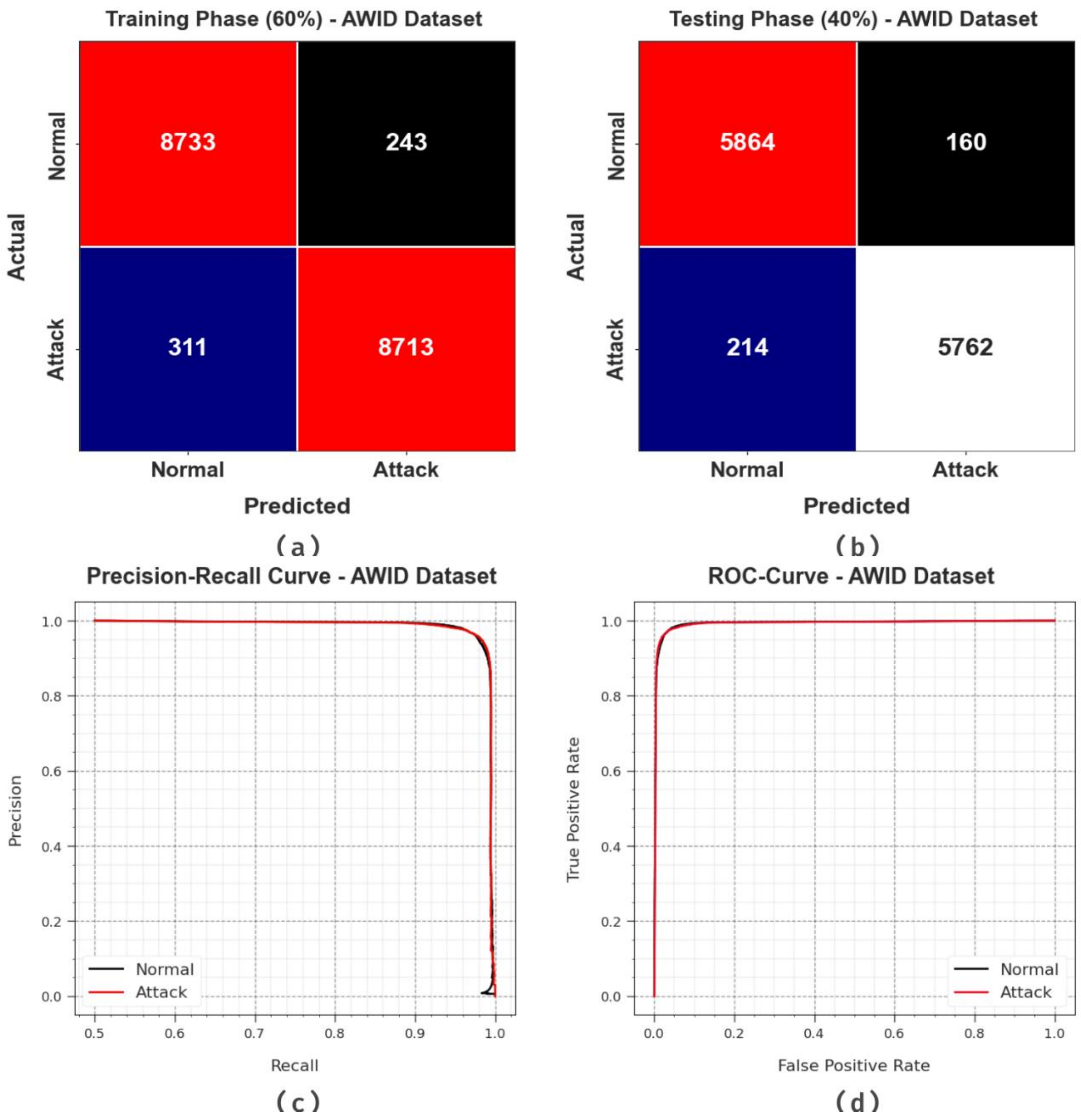

Figure 7. AWID dataset: (**a**,**b**) Confusion matrices, (**c**) PR_curve, and (**d**) ROC.

Table 3. Recognition outcomes of the NIDS-LOFSDL technique on the AWID dataset.

Class	$Accu_y$	$Prec_n$	$Reca_l$	$F1_{Score}$
		TR set (60%)		
Normal	97.29	96.56	97.29	96.93
Attack	96.55	97.29	96.55	96.92
Average	96.92	96.92	96.92	96.92
		TS set (40%)		
Normal	97.34	96.48	97.34	96.91
Attack	96.42	97.30	96.42	96.86
Average	96.88	96.89	96.88	96.88

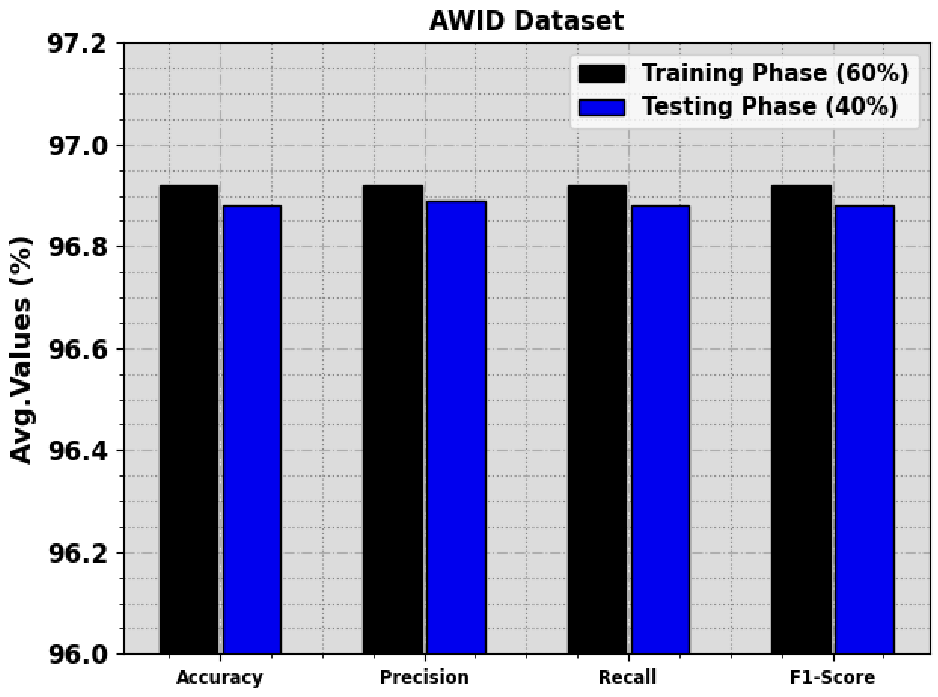

Figure 8. Average values of the NIDS-LOFSDL technique on the AWID dataset.

Figure 9 illustrates the training accuracy TR_accu_y and VL_accu_y values accomplished by the NIDS-LOFSDL algorithm on the AWID dataset. TL_accu_y is determined by evaluating the NIDS-LOFSDL methodology on the TR dataset, whereas the VL_accu_y value is computed by calculating the outcome on a separate testing dataset. The outcomes show that both the TR_accu_y and VL_accu_y values increased with an upsurge in the number of epochs. Therefore, the performance of the NIDS-LOFSDL methodology enhances the TR and TS datasets, with an increase in the number of epochs.

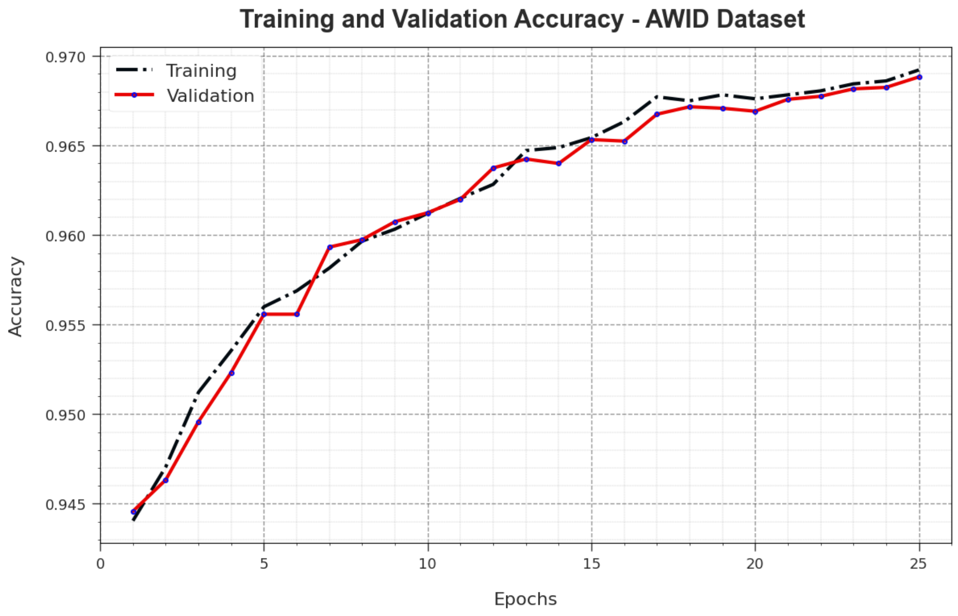

Figure 9. $Accu_y$ curve of the NIDS-LOFSDL technique on the AWID dataset.

In Figure 10, the TR_loss and VR_loss curves of the NIDS-LOFSDL approach on the AWID dataset are shown. TR_loss corresponds to the error between the predictive solution and the original values of the TR data. VR_loss signifies the performance outcomes of the

NIDS-LOFSDL technique on individual validation data. The outcomes imply that both TR_loss and VR_loss values tend to decrease with increasing numbers of epochs. The outcomes represent the enhanced performance of the NIDS-LOFSDL technique and its capability to produce accurate classification. The decreased TR_loss and VR_loss values demonstrate the better solution of the NIDS-LOFSDL technique in terms of capturing the patterns and relationships.

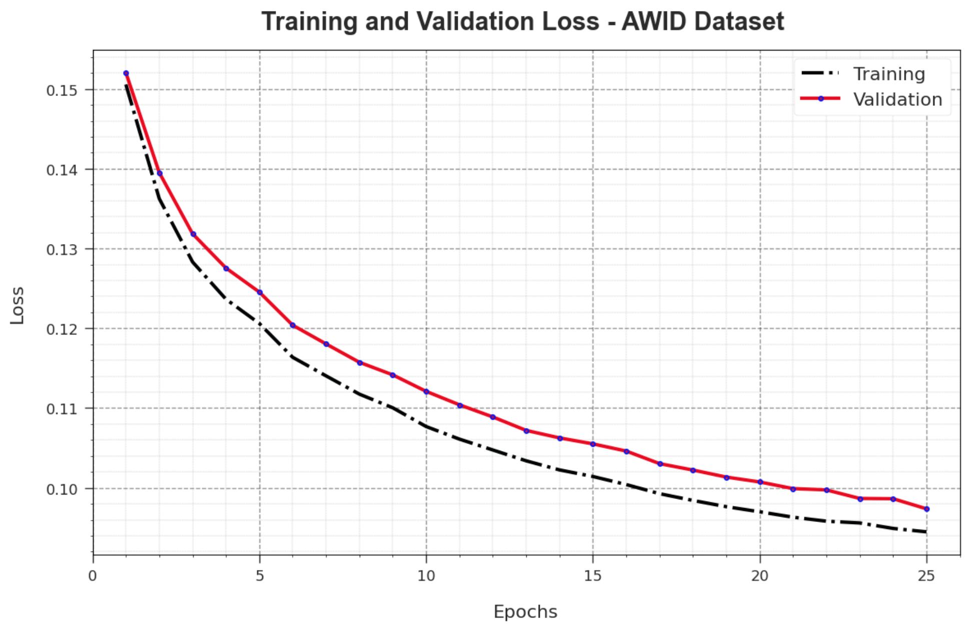

Figure 10. Loss curve of the NIDS-LOFSDL technique on the AWID dataset.

To ensure better results of the NIDS-LOFSDL technique, an extensive comparative analysis was conducted and the results are shown in Table 4 and Figure 11 [24,25]. The simulation values state that the SVM, NB-Bagging, NB-Adaboost, GCHSE, and CNN-Adaboost approaches achieved the worst performance. However, the BBAFS-DRL approach demonstrated a considerable performance with an $accu_y$ of 95.04%, $prec_n$ of 95.22%, $reca_l$ of 95.06%, and an F_{score} of 95.04%. Nevertheless, the NIDS-LOFSDL technique outperformed all other models with a maximum $accu_y$ of 96.92%, $prec_n$ of 96.92%, $reca_l$ of 96.92%, and an F_{score} of 96.92%. These outcomes confirm the effective performance of the NIDS-LOFSDL methodology on IDS.

Table 4. Comparative analysis outcomes of the NIDS-LOFSDL algorithm and other methods [24,25].

Methods	$Accu_y$	$Prec_n$	$Reca_l$	$F1_{Score}$
NIDS-LOFSDL	96.92	96.92	96.92	96.92
BBAFS-DRL	95.04	95.22	95.06	95.04
SVM	75.91	78.72	76.24	77.76
NB-Bagging	70.01	69.53	72.81	70.93
NB-Adaboost	71.34	74.44	73.13	74.07
GCNSE	80.17	80.01	81.28	80.82
CNN-Adaboost	74.16	69.28	71.53	68.16

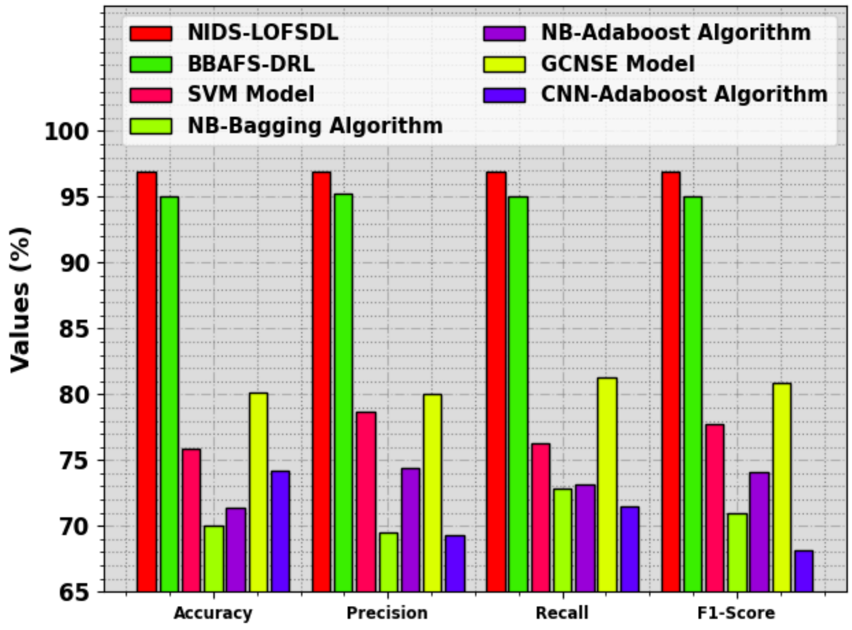

Figure 11. Comparative analysis outcomes of the NIDS-LOFSDL algorithm and other methodologies.

5. Conclusions

In the current study, a novel NIDS-LOFSDL technique has been developed for the detection of intrusions so as to accomplish network security. The NIDS-LOFSDL technique follows the concept of FS with a hyperparameter-tuned DL model for intrusion recognition. For the purpose of FS, the NIDS-LOFSDL technique uses the LOFS technique, which helps in improving the classification outcomes. Besides this, the ABiLSTM model is also executed for intrusion detection. In order to enhance the intrusion detection results of the ABiLSTM methodology, GTO is deployed for hyperparameter tuning. For validating the enhanced solution of the NIDS-LOFSDL system upon intrusion detection, a comprehensive range of experiments was conducted. The simulation values establish the promising results of the NIDS-LOFSDL system compared to the recent state-of-the-art DL approaches, with an improved accuracy of 96.88% and 96.92% on UNSW-NB15 and AWID datasets, respectively. Future research works can extend the proposed model to accommodate the dynamic and evolving nature of network threats. Besides this, continuous adaptation and learning mechanisms within the model, such as online or semi-supervised learning, can also be incorporated to enhance the capability of intrusion detection patterns proficiently. Finally, the scalability issue of the NIDS-LOFSDL technique should be resolved in order to enable it to be deployed in large-scale environments with high-speed data streams.

Funding: This research work was funded by Institutional Fund Projects under grant no. IFPIP: 194-611-1443. Therefore, the authors gratefully acknowledge the technical and financial support provided by the Ministry of Education and Deanship of Scientific Research (DSR), King Abdulaziz University (KAU), Jeddah, Saudi Arabia.

Data Availability Statement: Data sharing does not apply to this article as no datasets were generated during the current study.

Conflicts of Interest: The author declares no conflict of interest.

References

1. Thakkar, A.; Lohiya, R. Fusion of statistical importance for feature selection in Deep Neural Network-based Intrusion Detection System. *Inf. Fusion* **2023**, *90*, 353–363. [CrossRef]
2. Pranto, B.; Alam Ratul, H.; Rahman, M.; Diya, I.J.; Zahir, Z.-B. Performance of machine learning techniques in anomaly detection with basic feature selection strategy—A network intrusion detection system. *J. Adv. Inf. Technol.* **2022**, *13*, 36–44. [CrossRef]

3. Katib, I.; Ragab, M. Blockchain-Assisted Hybrid Harris Hawks Optimization Based Deep DDoS Attack Detection in the IoT Environment. *Mathematics* **2023**, *11*, 1887. [CrossRef]
4. Moizuddin, M.D.; Jose, M.V. A bio-inspired hybrid deep learning model for network intrusion detection. *Knowl.-Based Syst.* **2022**, *238*, 107894. [CrossRef]
5. Talukder, M.A.; Hasan, K.F.; Islam, M.M.; Uddin, M.A.; Akhter, A.; Yousuf, M.A.; Alharbi, F.; Moni, M.A. A de-pendable hybrid machine learning model for network intrusion detection. *J. Inf. Secur. Appl.* **2023**, *72*, 103405.
6. Sah, G.; Banerjee, S.; Singh, S. Intrusion detection system over real-time data traffic using machine learning methods with feature selection approaches. *Int. J. Inf. Secur.* **2023**, *22*, 1–27. [CrossRef]
7. Maabreh, M.; Obeidat, I.; Abu Elsoud, E.; Alnajjar, A.; Alzyoud, R.; Darwish, O. Towards Data-Driven Network Intrusion Detection Systems: Features Dimensionality Reduction and Machine Learning. *Int. J. Interact. Mob. Technol.* **2022**, *17*, 123–135. [CrossRef]
8. Ragab, M.; Alshammari, S.M.; Al-Ghamdi, A.S. Modified Metaheuristics with Weighted Majority Voting Ensemble Deep Learning Model for Intrusion Detection System. *Comput. Syst. Sci. Eng.* **2023**, *47*, 2497–2512. [CrossRef]
9. Ragab, M.; Sabir, M.F.S. Outlier detection with optimal hybrid deep learning enabled intrusion detection system for ubiquitous and smart environment. *Sustain. Energy Technol. Assess.* **2022**, *52*, 102311. [CrossRef]
10. Kocher, G.; Kumar, G. Analysis of machine learning algorithms with feature selection for intrusion detection using unsw-nb15 dataset. *Int. J. Netw. Secur. Its Appl.* **2021**, *13*, 21–31. [CrossRef]
11. Sharma, B.; Sharma, L.; Lal, C.; Roy, S. Anomaly based network intrusion detection for IoT attacks using deep learning technique. *Comput. Electr. Eng.* **2023**, *107*, 108626. [CrossRef]
12. Mohy-Eddine, M.; Guezzaz, A.; Benkirane, S.; Azrour, M. An efficient network intrusion detection model for IoT security using K-NN classifier and feature selection. *Multimed. Tools Appl.* **2023**, *82*, 23615–23633. [CrossRef]
13. Hosseini, S.; Sardo, S.R. Network intrusion detection based on deep learning method in the internet of thing. *J. Reliab. Intell. Environ.* **2023**, *9*, 147–159. [CrossRef]
14. Syed, N.F.; Ge, M.; Baig, Z. Fog-cloud based intrusion detection system using Recurrent Neural Networks and feature selection for IoT networks. *Comput. Netw.* **2023**, *225*, 109662. [CrossRef]
15. Du, J.; Yang, K.; Hu, Y.; Jiang, L. NIDS-CNNLSTM: Network intrusion detection classification model based on deep learning. *IEEE Access* **2023**, *11*, 24808–24821. [CrossRef]
16. Ravi, V.; Chaganti, R.; Alazab, M. Recurrent deep learning-based feature fusion ensemble meta-classifier approach for intelligent network intrusion detection system. *Comput. Electr. Eng.* **2022**, *102*, 108156. [CrossRef]
17. Atefinia, R.; Ahmadi, M. Network intrusion detection using multi-architectural modular deep neural network. *J. Supercomput.* **2021**, *77*, 3571–3593. [CrossRef]
18. Wang, Z.; Jiang, D.; Huo, L.; Yang, W. An efficient network intrusion detection approach based on deep learning. *Wirel. Netw.* **2021**, 1–14. [CrossRef]
19. Ra'ed, M.; Al-qudah, N.E.A.; Jawarneh, M.S.; Al-Khateeb, A. A Novel Improved Lemurs Optimization Algorithm for Feature Selection Problems. *J. King Saud Univ. -Comput. Inf. Sci.* **2023**, *35*, 101704.
20. Jiang, K.; Huang, Z.; Zhou, X.; Tong, C.; Zhu, M.; Wang, H. Deep belief improved bidirectional LSTM for multivariate time series forecasting. *Math. Biosci. Eng.* **2023**, *20*, 16596–16627. [CrossRef]
21. Ghith, E.S.; Tolba, F.A.A. Tuning PID Controllers Based on Hybrid Arithmetic Optimization Algorithm and Artificial Gorilla Troop Optimization for Micro-Robotics Systems. *IEEE Access* **2023**, *11*, 27138–27154. [CrossRef]
22. UNSW_NB15. 2023. Available online: https://www.kaggle.com/datasets/mrwellsdavid/unsw-nb15 (accessed on 26 August 2023).
23. Bee-Mar. AWID Intrusion Detection. 2023. Available online: https://github.com/Bee-Mar/AWID-Intrusion-Detection/blob/master/final_documents/resources/dataset-headers-reduced-removed-null.zip (accessed on 26 August 2023).
24. Wang, A.; Wang, W.; Zhou, H.; Zhang, J. Network intrusion detection algorithm combined with group convolution network and snapshot ensemble. *Symmetry* **2021**, *13*, 1814. [CrossRef]
25. Priya, S.; Kumar, K.P.M. Binary bat algorithm based feature selection with deep reinforcement learning technique for intrusion detection system. *Soft Comput.* **2023**, *27*, 10777–10788. [CrossRef]