SSRG International Journal of Electronics and Communication Engineering
ISSN: 2348-8549/ https://doi.org/10.14445/23488549/IJECE-V11I6P111

Volume 11 Issue 6, 122-136, June 2024

Original Article

SDN-IDS: A Deep Learning Model for Detecting DDoS Attacks

M. Ahsan Shariff[1], C. Nelson Kennedy Babu[2]

[1,2]*Department of Computer Science and Engineering, Saveetha School of Engineering, Saveetha Institute of Medical and Technical Sciences, Chennai, Tamil Nadu, India.*

[1]*Corresponding Author : ahsan.shariff@gmail.com*

Received: 18 April 2024 Revised: 27 May 2024 Accepted: 12 June 2024 Published: 29 June 2024

Abstract *- The centralization of control and programmability in Software-Defined networking (SDN) have enhanced network functionality, but they have also made it vulnerable to security threats like Distributed Denial of Service (DDoS) attacks, which may target both the data and control planes. To detect and mitigate the DDoS attacks in SDN's control plane, a novel attack detection model is proposed in this research. The proposed model is developed utilizing Deep Learning (DL) and metaheuristic optimization algorithms. The key objective of this research is to classify and detect the attacks in SDN's control plane layer. The proposed model, SDN-Intrusion Detection System (SDN-IDS), includes four main phases: data collection, data preprocessing, feature selection and classification. Initially, the InSDN dataset is collected to train and evaluate the research model. The data preprocessing phase includes data cleaning, data transformation, and normalization processes. After preprocessing, a Binary variant of the Ant Lion Optimizer (BALO) algorithm is used for selecting optimal features from the input dataset. Based on the selected features, the Attention-Based Bidirectional Long Short-Term Memory (ABiLSTM) model is implemented for classification. To improve the classification accuracy of the ABiLSTM model, the Bayesian Optimization (BO) technique is applied for hyperparameter tuning. The SDN-IDS model is assessed in terms of detection rate, accuracy, f1-score, FAR, and precision. Based on this analysis, the model attained 99.61% accuracy, 99.53% detection rate, 99.70% precision, 99.58% f1-score, and 0.46% FAR. Overall, these results indicate that the proposed SDN-IDS model effectively detects and classifies DDoS attacks within the SDN control layer with higher accuracy while maintaining a low FAR compared to the existing models.*

Keywords *- SDN, IDS, DDoS, Attack detection, InSDN dataset, Binary ALO, ABiLSTM, Bayesian optimization.*

1. Introduction

SDN offers enhanced programmability, management, efficiency, and flexibility in comparison to conventional networks. These are due to the network's inherent mutual segregation or independence of the data and control planes. The separation of two planes and the centralized behaviour of SDN improve security against DDoS attacks by simplifying the application of network regulations. The controller's capacity to screen network traffic and identify malevolent streams was assigned to its comprehensive network perspective. The separation of data and control planes has offered many advantages, but it has additionally posed a new difficulty regarding its vulnerability to DDoS attacks. DDoS pose a significant risk to SDN as it involves intentionally disrupting the services provided to normal users [1].

A DDoS attack is a method of flooding a server with a large amount of internet traffic, causing it to become unreachable to normal users. It imposes limitations on users' network access, perhaps causing a complete halt to the entire network. Over the past decade, this assault has transformed into a significant threat because of the heightened severity and intensity of its impact on the networks [2]. The intensity of DDoS attacks is steadily escalating each year, resulting in significant disruption of network services. Several variables contributing to the increase in the severity of these attacks include the widespread use of IoT devices, the abundance of upload bandwidth, and the easy accessibility of source codes of the attacks. The menacing nature of DDoS attacks has prompted businesses and scholars to develop innovative measures to protect the internet infrastructure [3].

The emergence of the SDN architecture was a response to the complexity of conventional networks. It allows for the establishment of a network that is both scalable and flexible and can be programmed according to specific needs. This is the most notable technical innovation in the domain of networking during the last few decades [4]. The

main concept of the architectural model is to separate the data forwarding layer from the network device's logical control. The SDN architecture consists of three linked levels (data, control, and application layers) and a pair of interfaces (northbound and southbound). This categorization leads to a straightforward network that simplifies its controllability [5].

Figure 1 depicts three levels of the SDN architecture, each serving a specific role and having a clear purpose. SDN implementation requires some mandatory elements, such as Networks Operating System (NOS), application network, and northbound and southbound APIs. However, optional features like language-based hypervisor or virtualization are not necessary. The subsequent sections delineate each layer and API interfaces in a hierarchical order, starting from the highest level and descending to the lowest level [6].

The forwarding devices based on software or hardware-based in the data plane (also called the infrastructure plane) could forward, change, and drop packets based on control plane regulations. The control planes, the SDN architecture's "brain," processes and forwards data plane data (southbound communication). It receives northbound application requests and creates forwarding rules to satisfy them. For network traffic monitoring, it can collect data layer device performance statistics. Applications in the application layer could interact with the control layers to provide networking needs like bandwidth and latency so the control plane can configure the data layers to meet application requirements [7].

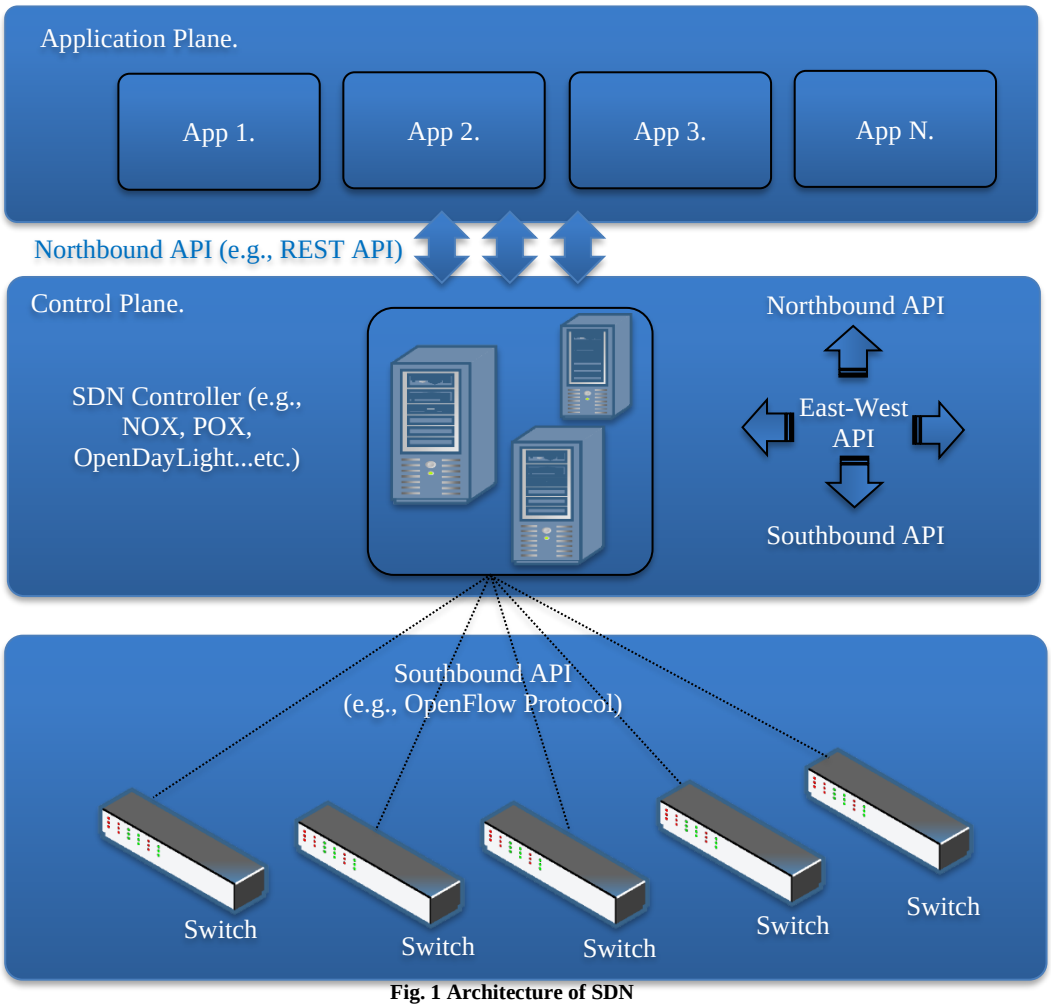

Fig. 1 Architecture of SDN

In the SDN architecture, the controllers are an especially appealing target for DDoS attacks since they may be viewed as a network's single point of failure. The ways that can be used to initiate DDoS attacks in the control layer include targeting the controller, the eastbound API, the southbound API, the northbound API, or the westbound API [8]. DDoS assaults aim to inundate the resources of the network by creating a substantial number of unauthorized

requests, therefore obstructing the delivery of network services to actual users. Within the context of the control layer, the objective of DDoS/DoS attacks was to hinder the normal functioning of the control layer by inundating it with fraudulent requests. As a result, valid requests from data or application layers were either postponed or entirely disregarded owing to insufficient resources [9].

Researchers have dedicated significant effort to developing efficient solutions for identifying and mitigating DDoS/DoS attacks. Similarly, most of the research in the field of deep learning for SDN security is mostly concentrated on the detection and prevention of DoS/DDoS attacks. In this context, the objective of a DDoS attack closely resembles that of a DoS attack. The only variation between this attack and a DDoS attack is in the severity [10].

Although SDN offers a multitude of advantages by detaching the control plane from the data plane, there is a link that is incongruous between DDoS attack and SDN. SDN's capabilities make it simple to identify DDoS attacks and respond to them [11]. The separation of the control plane and the data plane in SDN results in the introduction of new types of attacks. Because of this, SDN might become a target of DDoS attacks. It is essential to have an efficient attack detection model in SDN to identify and mitigate these attacks quickly [12]. Through the preservation of network availability, the protection of resources, the maintenance of service quality, the prevention of financial losses, the protection of data and resources, and the enhancement of organizational reputation and trust, the detection model plays a significant role in ensuring that network operations are uninterrupted and improving overall cybersecurity defences [13]. Figure 2 depicts a common architecture of the attack detection model.

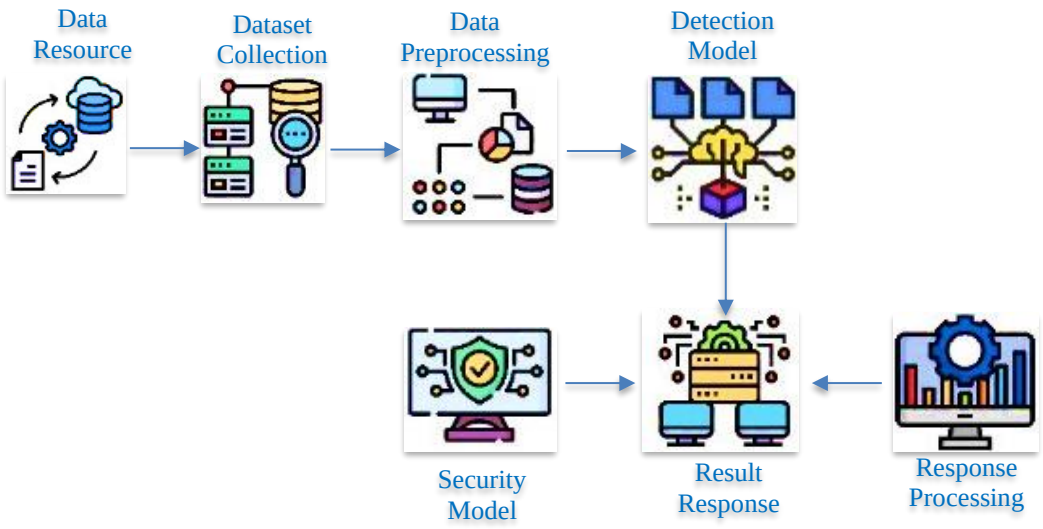

Fig. 2 Common architecture of attack detection model

1.1. Research Contribution

In this research, an attack detection model called SDN-IDS is proposed, which uses the DL and optimization techniques to identify attacks in SDN's control layer. The research model includes data collection, data preprocessing, feature selection, and classification processes. The InSDN dataset is used to train and evaluate the proposed model. In data preprocessing, data cleaning, data transformation, and normalization processes are performed. After preprocessing, the feature selection is carried out using the BALO algorithm. Based on the selected attributes, the Attention-BiLSTM DL model was employed to detect and classify the attacks. Further to improve the classification performance, the Bayesian Optimization technique is applied for hyperparameter tuning. The proposed SDN-IDS model's performance is evaluated based on accuracy, detection rate (recall), precision and f1-score. Based on the research contribution, the following presents the research objectives:

- To develop a novel attack detection model called SDN-IDS designed for identifying attacks in the control layer of SDNs.
- To utilize the InSDN dataset for training and evaluating the performance of the proposed SDN-IDS model in identifying attacks.
- To apply the BALO algorithm for feature selection to choose the most significant features for identifying attacks within the SDN environment.
- To employ the Attention-BiLSTM deep learning architecture to perform identification and classification of attacks based on the selected features.
- To utilize the Bayesian Optimization technique for hyperparameter tuning to optimize the performance of

the Attention-BiLSTM model within the SDN-IDS framework.

- To evaluate the performance of the proposed SDN-IDS model to assess its effectiveness in detecting and classifying DDoS attacks accurately and efficiently within the SDN control layer.

The subsequent sections of the research are structured as follows: Section 2 is dedicated to investigating the existing attack detection models in SDN. Section 3 provides the presentation of the materials and methods employed in this research. The implementation of the proposed SDN-IDS model is presented in Section 4. Section 5 includes the discussion of experimental analysis and the comparison of findings. In Section 6, the research concludes the work and suggests the possible areas for further research.

2. Related Works

SDNs are vulnerable to DDoS attacks. The research [14] presented a Machine-Learning (ML) and feature-engineering-based SDN attack detection method. The best subset of features was identified utilizing an updated binary grey wolf optimizer approach after cleaning and normalizing the data set. After training and testing the ideal feature subset in Support Vector Machines (SVM), XGBoost, Decision Trees, Random Forests (RF), and K-Nearest Neighbors (k-NN), the optimal classifier for detecting attacks was chosen and implemented in the controller. Multiple performance measurements demonstrated that RF performed better. The study [15] presented an effective SDN data and control planes DDoS identification method. The control layer used a DL method to identify DDoS attacks utilizing new traffic data characteristics. DL-based detection employed Bidirectional Gate Recurrent Units (BGRU) with Autoencoders (AE). The destination address of the unknown IP, TLP header, packet inter-arrival time, and ToS header were proposed control layer features. The approach measured the average arrival bit rate of the switch with the destination address of an unknown data plane. Experimental findings showed that the model was more accurate and had fewer false alarms. DDoS attacks were widespread in SDN network security.

The study [16] detected DDoS attacks using SDN live traffic feature extraction and categorization. A good feature extraction process will filter task-relevant data and enhance ML algorithms. Some popular classifiers, including XGBoost, KNN, RF, SVM, and NB, were trained and evaluated using these extracted features to find the best classifier. SVM outperformed other classifiers. The Comprehensive Coordinated (CC)-Guard defensive framework was developed in [17]. The framework included the attack detection trigger module, which speeds up the DDoS reaction. The switch migration module reduced controller congestions and simplified network flow transfers. The anomaly detection module enhanced detection accuracy with coarse-grained two-stage detections. The mitigation module cleared anomalous blacklist traffics via the controller's cross-domain collaborations. Results demonstrated that the framework could defend against DDoS attacks in real time, with higher detection accuracy and efficient network resource use. An SD-Internet of Things (IoT) DDoS detection model using feature extraction was developed in [18]. Normal and DDoS attack traffic was produced utilizing the hping3, and Distributed Internet Traffics Generator programs, and six characteristics were extracted. Popular classifiers, including RF, LGBM, SVM, and KNN were modelled and evaluated with the six vector pairs of attack and normal data to evaluate the derived features. The best LGBM model was used in the SDN controller to identify real-time traffic attacks.

The research [19] proposed a safe and intelligent smart city DDoS defense system. This study reduced smart city DDoS attacks utilizing SDN security controllers and optimized ML models. This study used SDN-based security controllers and an optimized XGBoost detection system to prevent common DDoS attacks in smart cities. The use of non-realistic datasets and non-qualified features causes significant false positives and low accuracy in existing techniques. Thus, SDN controller DDoS attacks may be detected using DL. Data preprocessing, cross-feature selection, and Recurrent Neural Networks (RNNs) detection were proposed in [20] to detect DDoS. The model was evaluated using false positive rate and detection accuracy on a benchmark dataset. Results show that the model identified DDoS attacks. The study [21] thoroughly investigated the connections between data layer switches and control layer applications and introduced two novel attacks, the Control layer reflection attack, to utilize hardware switches based on SDN's low processing power. Reflection attacks used indirect and direct data layer events to induce the control layer to send costly downlink messages to SDN switches. A probing-triggering attack with a two-phase approach improved reflection attacks' efficiency and power. Attacks on a testbed with three physical OpenFlow switches showed that they damaged new flows and interrupted SDN controller-switch connections.

A Spider Monkey-based Elman Spike Neural Networks (SM-ESNN) model was proposed in [22] to detect SDN intrusion risks. Detecting central controller intrusions and floods via multidimensional IP traffic analysis. Additionally, the dataset was used to update the SDN's secure defense system. A software defense system with detection and mitigation modules was developed. The model improved SDN security by rapidly and accurately detecting attacks. FMDADM, an SDN-based attack mitigation and detection system for IoT networks, was developed in [23]. The first module used a 32-packet window size for average drop rate early detection. The

second module utilized a unique double-check mapping function to identify data layer attacks. The next module was a detection application based on ML, including preprocessing, feature extractions, training and testing, and classifications. This model identified DDoS assaults, and the last module introduced attack mitigation. The framework detected DDoS assaults at higher and lower rates, distinguished attack traffic from flash crowds, and protected remote and local IoT nodes by avoiding ISP infections.

DL has been used to detect and prevent DDoS attacks in SDN systems. A hybrid DL model in [24], which used a 1D-Convolutional Neural Network (CNN), Dense Neural Network (DNN), and gated recurrent unit (GRU), outperformed traditional ML algorithms in detecting DDoS attacks and optimizing SDN networks.

DDoS and its version, Low-Rate DDoS was one of the hardest to identify and defend since the fraudulent user created destructive traffic slowly. ML, specifically Federated Learning (FL), has been successful in identifying and protecting against these threats. A Weighted-FL (WFL) was proposed in [25] to identify LR-DDoS assaults. The model has proven its worth for IDS, notably IoT networks based on SDN.

Table 1. Comparative analysis of reviewed research works

Ref	Approach	Application Scenario	Advantages	Disadvantages
[14]	Feature engineering and ML-based approach	SDN DDoS detection	Identified best feature subset using optimization; RF performed better	Manual feature engineering was not generalized well
[15]	DL-based approach	SDN data plane and control plane DDoS detection	Utilized DL for accurate detection with fewer false alarms	Required significant computational resources for DL training
[16]	Feature extraction and ML-based approach	SDN live traffic feature extraction and categorization	SVM outperformed other classifiers	Manual feature extraction does not capture all relevant information
[17]	CC-Guard framework	Real-time defense against DDoS attacks in SDN networks	Higher detection accuracy and efficient resource use	Complexity in implementation and deployment
[18]	Feature extraction and ML-based approach	SD-IoT DDoS detection	Utilized LGBM for real-time detection; extracted relevant features	Dependency on manually extracted features limit adaptability
[19]	SDN-based security controllers and optimized ML models	Smart City DDoS defense	Utilized SDN and ML for effective defense against DDoS attacks	Require specialized hardware for ML optimization
[20]	DL-based approach	DL-based detection with RNNs	DL approach for DDoS detection improved accuracy	Computational complexity of DL training
[21]	Attack investigation and prevention	Control Plane Reflection Attacks in SDN-enabled switches	Identified novel attack types; highlighted vulnerabilities	Requires advanced understanding of SDN infrastructure
[22]	Intrusion detection and defense system	SDN intrusion risk detection and mitigation	Improved SDN security with rapid and accurate attack detection	Implementation complexity and resource overhead
[23]	DDoS attack detection and mitigation	SDN-based DDoS detection and mitigation for IoT networks	Modular technique for mitigation and detection of attacks.	Complexity in integrating with existing IoT infrastructure
[24]	Hybrid DL-based approach	DL-based detection and optimization for SDN networks	Outperformed traditional ML algorithms in detection and optimization	Increased computational complexity compared to traditional ML
[25]	FL-based approach	LR-DDoS assault identification in SDN-based IoT networks	WFL effectively identified LR-DDoS assaults	Dependency on FL infrastructure

2.1. Research Gap Analysis

It has been observed that traditional ML methods, while demonstrating effectiveness in DDoS attack detection, often necessitate manual feature engineering. This approach cannot capture all pertinent information and could struggle with generalizability across various network environments. In contrast, DL-based techniques have emerged as promising solutions for enhancing detection accuracy and reducing false positives. However, these methods require substantial computational resources for training and lack interpretability, posing challenges for real-time deployment within SDN environments.

Additionally, a comprehensive evaluation regarding their efficacy against evolving and intricate DDoS attack methods. A pressing need exists for the evaluation of more standardized and realistic datasets. This would facilitate the accurate benchmarking and comparison of different detection methodologies. While some studies have directed their focus towards specific application scenarios, such as smart city DDoS defense or IoT networks, a comprehensive framework encompassing the diverse range of attack vectors and network topologies encountered in real-world SDN deployments remains elusive. Addressing these research gaps is paramount for achieving advancements in state-of-the-art SDN-based DDoS detection and defense mechanisms.

3. Materials and Methods

This section presents the subject of materials and methods focused on this research to develop an attack detection model. The primary aim of this attack detection model is to identify and minimize the attacks in the SDN's control layer. The SDN-IDS model includes data collection, data preprocessing, feature selection and classification processes. For these processes, the research model utilizes different techniques and algorithms, as discussed in the following sections. Figure 3 represents the architecture of the proposed SDN-IDS model.

3.1. Dataset Collection

Generally, in IDS, the training dataset quality determines the performance. A fundamental challenge to implement detection models is the lack of current real-world datasets. The latest real-time dataset, InSDN, was utilized in this work to train and evaluate the proposed SDN-IDS model. The InSDN dataset is publicly available. Brute Force, Probe, Web, DoS application, U2R, password guessing, and DDos attacks are included in this dataset. In addition, InSDN normal traffic has some properties. Attacks from internal and external networks are used in the dataset to simulate attacks. In CSV format, it provides 83 statistics features, including protocol, duration, byte number, packet number, etc. Table 2 shows that the dataset includes 343,939 instances for normal and attack traffic, 68,424 for normal and 275,515 for attack classes [26].

Table 2. InSDN dataset distribution

Attack Classes	Number of Instances
Botnet	164
Brute force	1405
DoS-Network	3772
DoS-Application	31628
DDoS	9943
Normal	68424
Probe	15225
U2R	17
Web-Application	192
Total	343939

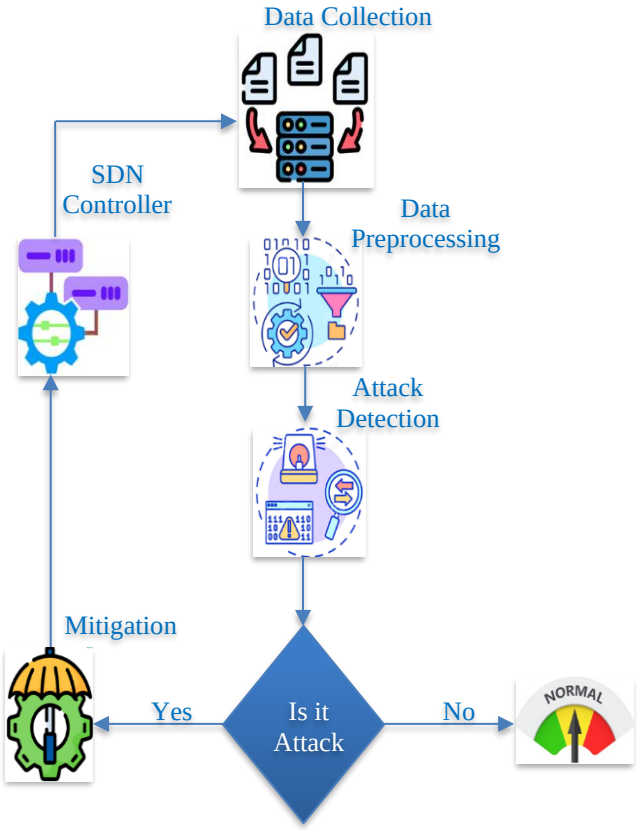

Fig. 3 Proposed SDN-IDS architecture

Since the SDN controller decides on regular and malicious traffic, attack traffic behaves normally. In contrast to the conventional networks of multiple attacks, the SDN network's centralized perspective and separation of the data layer from the control layer provide the hacker with additional options. Since an attacker may access the victims' server, these intrusions are hard to detect. This data collection could serve as a useful dataset for real-world model evaluation. This dataset has no duplicate information to keep the training model focused on the most prevalent records. The data set was split into 70:30 for training and evaluating the research model.

3.2. Data Preprocessing

Data preparation refers to the process of preparing raw data to be suitable for a DL/ML model. During the preprocessing step, any missing values must be changed before the training phase. The research model includes preprocessing steps such as data cleaning, data transformation and normalization. Data cleaning involves fixing or removing data that is replicated, damaged, formatted incorrectly, or incomplete. Transformation of data is the process of translating data from one type or format to another. An example of this is transforming text into integers, which is necessary for it to be used in DL algorithms.

This work employed min-max scaling to standardize all characteristics in the dataset, considering the different scales available. Min-max scaling is a linear adjustment that is done to the original data to normalize it. The scaling formula is expressed as follows, where d represents the original data and d' represents the transformed data:

$$d' = \frac{(d-min)}{(max-min)} \quad (1)$$

In this context, "max" and "min" represent the highest and lowest values found in the column where the variable "d" is located. This standardization procedure has wide-ranging applicability. By employing this technique, data is transformed to a scale of [0, 1] while preserving the real data structures, which sets it apart from the Z-Score standardization approach. Consequently, this method allows for expedited and uncomplicated data standardization within a certain range [14].

3.3. Binary Ant Lion Optimizer

Meta-heuristic optimization approaches are divided into population-based and local search-based algorithms. Local search-based optimization approaches to progress with a single solution and use the neighborhood mechanism to enhance it. The population-based methods proceed by using multi-solution to improve candidate solutions at each generation to develop one or more superior solutions. The Ant Lion Optimizer (ALO) algorithm uses these two methods to create an intelligent algorithm that can search successfully using local exploitation and global exploration. In contrast to various meta-heuristic approaches, ALO is simple, customizable, and implementable, making it suitable for a range of optimization tasks [27].

In this research, the ALO metaheuristic algorithm is planned to be used for the feature selection process. Since the binary variant algorithms are superior for feature selection, this research model used the binary type of ALO called Binary Ant Lion Optimizer (BALO). Binary vector representations are used to get the pertinent features. In this BALO technique, the solution vectors are denoted as (10101100....). In this representation, a value of 1 indicates that a certain feature is chosen, while a value of 0 indicates that the feature is not included in the subset [28].

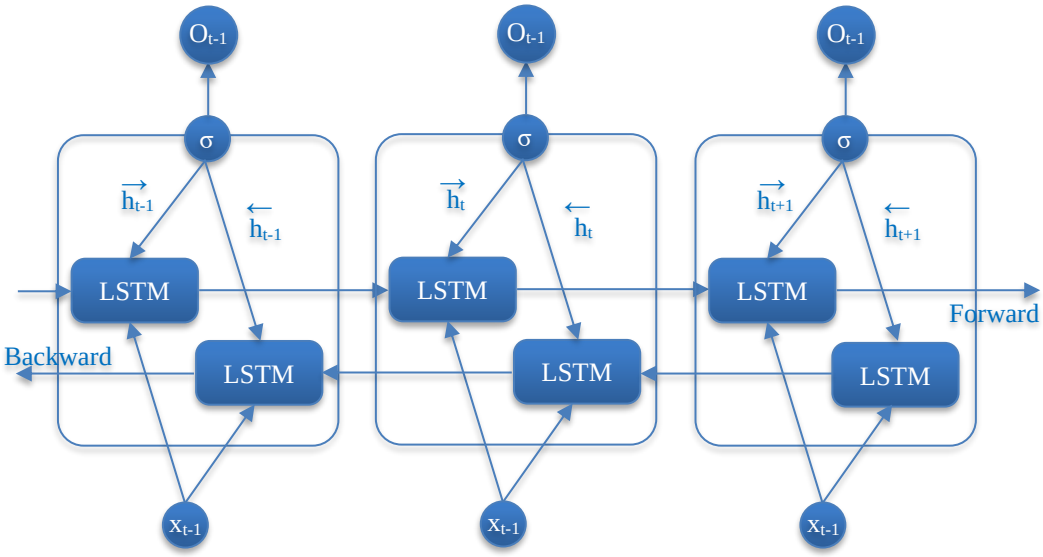

Fig. 4 Bi-LSTM architecture

3.4. Bi-LSTM

Due to the LSTM's limitation of incorporating only forward sequence information into Neural Networks (NN) prediction and the challenge of capturing the content of backward data during model training, it was susceptible to problems such as gradient disappearance or gradient inflation when dealing with connections between distant nodes. Conversely, the BiLSTM could effectively preserve information from nodes that are further away. The BiLSTM layer is constructed by combining a forward LSTM with a

backward LSTM. Obtaining two different sets of hidden layer representation for each phrase is accomplished by the BiLSTM model through the utilization of sequential and inverse order computations. The final layer representation is then created by the process of vector stitching, which brings about an improvement in speed on time-series data that is more thorough. The forgetting gate, the input gate, and the output gate are the three gating structures that are included in each LSTM cell, including the BiLSTM structure, as seen in Figure 4 [29].

Unlike LSTM, which can only incorporate the data from a forward sequence into the NN for prediction, BiLSTM is composed of both a backward and a forward LSTM unit. All the LSTM units conform to the structure of LSTM, including separate forward and backward units. Therefore, based on the studies that have been conducted, BiLSTM is superior to LSTM when it comes to the prediction of time series data.

4. Proposed SDN-IDS Model

4.1. BALO-Based Feature Selection

This research implements a BALO technique for feature selection. Each ant in this method adjusts its location based on Equation (1). The crossover procedure is performed by combining the dual binary solutions acquired from a random walk over the selected and elite antlion.

$$A_i^{t+1} = CO(RX_1, RX_2) \tag{2}$$

In this equation, $CO(a, b)$ represents the appropriate crossover among solutions a and b. RX_1 and RX_2 were binary vectors that indicate the impact of the elite antlion and a randomly picked antlion.

The average operators employed in the conventional ALO serve to draw the ants towards the antlion trap cones, making them the primary operators for global search or exploration. In this case, the Crossover Operators (CO) substitute the average operators that are utilized in the ALO algorithm with two operators aiming to attain exploration and global searching. The CO was a straightforward method for generating the intermediate solutions by combining two existing solutions. The employed CO was a basic stochastic CO that alternates among the two vector inputs with equal probabilities, as specified in the equation as follows:

$$a^d = \begin{cases} a_1^d & if(rand) \geq 0.5 \\ a_2^d & otherwise \end{cases} \tag{3}$$

In this equation, a^d was the output obtained by performing crossover at dimension d among vectors a_1^d and a_2^d. RX_1 denotes the ant's attraction towards the elite antlion, which was depicted as a random movement around the elite antlion with a well-suited step size. The process involves stochastic mutations occurring over a chosen ant lion, with the mutation value that is appropriate for the binary-valued elite antlion. This is expressed in the binary form as described in Equation (3). RX_2 is a representation of the attraction exerted by the other antlions. It is achieved by applying random mutations to an antlion in binary mode, using the roulette wheel selection technique for choosing the mutations.

$$a_{out}^d = \begin{cases} a_{in}^d & if\ rand1 \geq r \\ rand2 & otherwise \end{cases} \tag{4}$$

The variable a_{out}^d represents a value of d-dimensional for the vector output resulting from mutations. The variable a_{in}^d refers to the input vector that will undergo mutation. 'rand1' and 'rand2' were dual random integers generated from the uniform distributions within the scale of [0, 1]. Here, 'r' represents the rate of mutation. It is evident that the value of r decreases directly with each repetition, spanning from 0.9 to 0, as indicated by Equation (5):

$$r = 0.9 + \frac{-0.9*(i-1)}{Itermax-1} \tag{5}$$

Here, 'r' represents the rate of mutation at the iteration' i', whereas $Itermax$ denotes the total iterations for executing the optimization. The ALO is an algorithm that is utilized to dynamically explore the feature space to discover the optimal subsets of features. The optimal feature subset is the one that has the lowest classification error rate and the fewest chosen attributes. The fitness function utilized in this algorithm to assess separate search agents was represented in the equation as follows:

$$FF = \alpha\gamma_R(D) + \beta\frac{|C|}{|TF|} \tag{6}$$

The classification error rate, $\gamma_R(D)$ was the rate at which classifier C incorrectly classifies instances based on the features selection decision D. The variable C represents the selected subset of the feature's length, while TF represents the total features in the dataset. The value of α is determined by calculating the minimal distance between the actual instance and the training instances [30].

The InSDN dataset includes 83 features. Out of 83 features, only 15 optimal features, such as Protocol, Tot Fwd Pkts, Init Bwd Win Byts, Subflow Bwd Pkts, Bwd IAT Tot, Bwd IAT Max, Bwd Pkt Len Max, Bwd IAT Mean, Bwd Pkt Len Mean, Bwd Header Len, Bwd Pkts/s, Flow Pkts/s, Flow duration, Fwd Header Len, and Flow IAT Mean features are selected using the BALO algorithm.

4.2. Attack Detection using A-Bi-LSTM

The A-BiLSTM model has been utilized in the SDN-IDS model for intrusion categorization. LSTM was an enhanced NN model that builds upon the RNN architecture

to address time-series problems. The unshaded sections indicate the current instant, while the shaded part indicates the preceding and subsequent actions, as shown in architecture. The network addresses the issues of the RNN's susceptibility to gradients vanishing and exploding gradients by incorporating three regulating units: memory, forget, and output gates. It achieves this by implementing a distinct method of defining a threshold for the gradient range and storing "memory". The fundamental purpose of LSTM is outlined as follows:

The forget gate combines with the previous moment's output h_{t-1} to filter the cell data C_{t-1}, removing the influence of these moments. The mathematical expression can be represented using Equation (7):

$$fg_t = \sigma\left(w_f \cdot [h_{t-1}, x_t] + b_f\right) \tag{7}$$

The sigmoid function is denoted as σ. The final output at the last layer was represented as h_{t-1}. The weighting coefficient and offsets of linear correlations were w_f and b_f, respectively. The result of the forget gate was fg_t and the input at the current time is x_t. The memory gate stores the data of the cell c_{t-1} together with an output h_{t-1} from the previous time step. It is expressed as:

$$mg_t = \sigma(w_i \cdot [h_{t-1}, x_t] + b_i) \tag{8}$$

$$c_t = \tanh(w_c \cdot [h_{t-1}, x_t] + b_c) \tag{9}$$

$$C_t = fg_t \times C_{t-1} \times i_t \times c_t \tag{10}$$

The equations include the input variables. i_t and c_t for the initial and subsequent sections, respectively. The updated cell state is denoted as C_t. The related weighted coefficients and offset values are represented by w_i, b_i, and w_c. The output gate is connected to the previous instant h_{t-1} and the current cell data C_t. The input provided to the neural network for processing is as follows:

$$O_t = (w_o \cdot [h_{t-1}, x_t] + b_o) \tag{11}$$

$$h_t = O_t \times \tanh(C_t) \tag{12}$$

Currently, O_t represents the top layer, and the values of h_{t-1} and x_t are determined by the utilization of the sigmoid function. The A-BiLSTM technique utilizes an attention mechanism to give individual weight to the parts of the feature sequences. The mechanism of attention highlights the most salient information, mitigates the impact of poorly correlated data, and enables the classifier to provide a highly accurate estimation.

$$Aten = \sum_{i=1}^{m} softmax\left(f(x_i)\right) \times x_i \tag{13}$$

The variable x_i represents the sequence of features that are inputted into the state of the attention scheme. The attention scheme, on the other hand, refers to the total of weights that are weighted [31].

4.3. Hyperparameter Tuning using BO

Optimizing the hyperparameters of the model is necessary to considerably enhance the model's prediction accuracy using the ideal combination of hyperparameters. Bayesian optimization is a method for optimizing parameters, as represented by the function expression indicated in equation (14).

$$X^* \in argmax_{x \in X} f(x) \tag{14}$$

In this equation, the variable *x* represents the hyperparameter's value that is being optimized, while $f(x)$ represents the performance function. The Bayesian optimization method employs a probabilistic agent model that utilizes a Gaussian model. This model is applied to a particular objective function *f*, where the input space was denoted as $x \in X$.

The data set D consists of *n* samples, denoted as $\{(x_1, y_1), (x_2, y_2), \ldots (x_n, y_n)\}$, where y_i is the value of the function f at x_i. Next, the Gaussian probability theory could be formulated in the following manner.

$$f \sim GPT[\mu(x), k(x, x')] \tag{15}$$

The function $\mu(x)$ represents the mean value, which is equal to the expected value of $f(x)$. The average value function is typically assigned a value of 0. The variable $k(x, x')$ represents a covariance function, which states that for any variables x and x', the covariance between $f(x)$ and $f(x')$ is given by $k(x, x')$ [32].

Table 3. BO-Based hyperparameters tuned

Hyperparameter	Value
No. of BiLSTM Layers	2
BiLSTM Hidden Size	256
No. of Attention heads	2
Batch Size	64
Learning rate	0.001
Dropout rate	0.5
Epochs	100

5. Results and Discussion

5.1. Experimental Setup

This section presents the experimentation and performance analysis of the proposed SDN-IDS model. As an experimental setup, the proposed model was experimented in a system that has Windows 10, 64-bit OS, with Intel(R) i7 processor @ 4.60 GHz CPU, and 16GB of RAM specifications. The Python 3.11.4 64-bit tool was used

to develop the model and used the Pandas, Numpy, and Scikit-learn libraries. The results attained for the SDN-IDS model are compared and validated with the current attack detection models in SDN.

5.2. Performance Metrics

The research model is evaluated based on accuracy, detection rate (recall), f1-score, FPR, and precision. These metrics are computed based on the values of True Positive (TP), False Positive (FP), True Negative (TN), and False Negative (FN), which are the four categories that can be used to classify the outcomes of the detection. The number of attack samples that were properly detected as attack samples by the model was denoted by TP. The number of normal samples that were correctly detected as TN denoted normal samples. The number of normal samples that were wrongly detected as attack samples by the model was denoted by FP. The value of FN provides a representation of the count of attack instances that the model wrongly detected to be normal samples.

In the context of the attack detection process, the term "accuracy" refers to the proportion of properly detected outcomes relative to the total number of samples.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (16)$$

Precision can be defined as the proportion of negative samples that were accurately detected as positive out of the total samples that were detected as positive.

$$Precision = \frac{TP}{TP+FP} \quad (17)$$

The detection rate was the proportion of samples that were actually positive to the total positive samples.

$$Detection\ Rate = \frac{TP}{TP+FN} \quad (18)$$

The F1 score is a statistic measure that is utilized for the purpose of determining how accurate optimistic class detections are. The ratio of samples that were accurately detected as positive to all samples that were detected to be positive is represented by the f1-score.

$$F1\ score = \frac{2\times Precision\times Recall}{Precision+Recall} \quad (19)$$

FPR in an IDS refers to the percentage of alerts generated by the system that are false alarms or non-malicious events.

$$FPR = \frac{FP}{FP+TN} \quad (20)$$

False positives can be caused by various factors, such as misconfigurations, outdated rules or signatures, or normal traffic that triggers the IDS. Therefore, minimizing false positives is an important consideration in the design and implementation of an IDS [17].

5.3. Performance Evaluation

This section presents a detailed analysis of the experimental results of the proposed SDN-IDS model. To determine the performance of the research model in identifying attacks in SDN, an assessment of its performance was conducted using the performance metrics.

Table 3. Results of the proposed SDN-IDS Model

Metric	Training Results	Testing Results
Accuracy	99.87	99.61
Detection Rate	99.80	99.53
Precision	99.89	99.70
F1-score	99.74	99.58
FPR	0.38	0.46

Table 3 represents the results of the proposed SDN-IDS model evaluated based on training and test sets. As it can be seen, the model has attained higher performances in the training set compared to the test set. The model attained 99.87% accuracy in training and 99.61% in testing, which has a performance difference of 0.26%. The detection rate of the model was 99.80% in training and 99.53% in testing, which has a variation of 0.27%. The precision rate of the model was 99.89 in training and 99.70 in testing, which has a difference of 0.19%. The f1-score of the model was 99.74% in training and 99.58% in testing, which has a variation of 0.16%. The model obtained 0.38% FAR in training and 0.46% in testing, which has a difference of 0.08%. All these differences will not impact the model's performance in validation. Figures 5 and 6 represent the graphical plot of the SDN-IDS model's performance in training and testing.

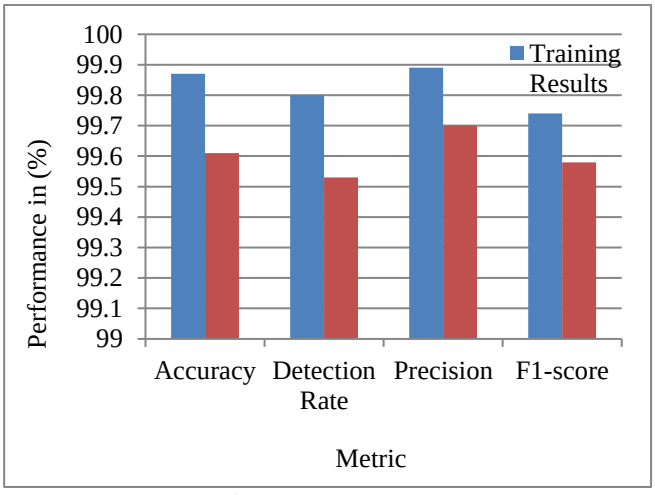

Fig. 5 Graphical plot of SDN-IDS model's training and test results

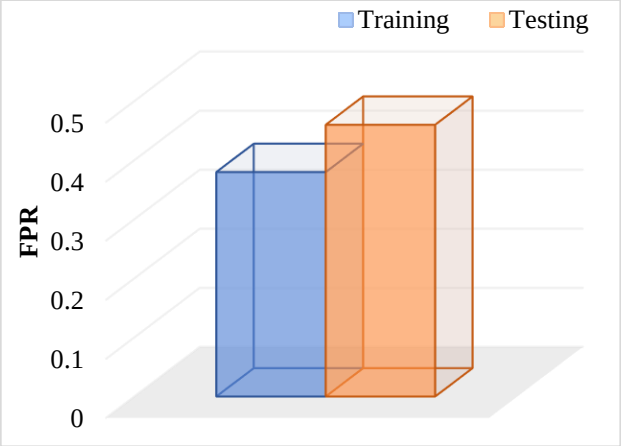

Fig. 6 Graphical plot of SDN-IDS model's FPR results

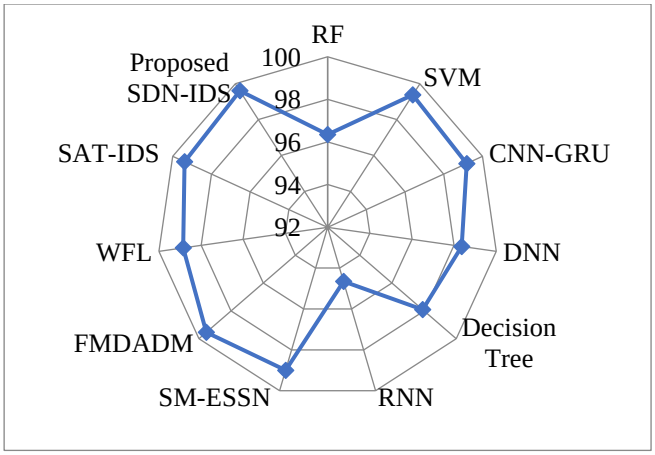

Fig. 7 Graphical plot of accuracy comparison

Table 4. Results comparison of the SDN-IDS model with current models

Models	Accuracy (%)	Detection Rate (%)	Precision (%)	F1-score (%)
RF (CIC-IDS-2018) [14]	96.35	97.23	95.10	96.16
SVM (Own Dataset) [16]	99.38	99.40	99.41	99.39
CNN-GRU (Own Dataset) [17]	99.18	99.34	98.97	99.28
DNN [17]	98.36	99.36	97.50	98.42
Decision Tree (BoT-IoT) [19]	97.90	93.00	93.00	93.00
RNN (SDN Dataset) [20]	94.66	NA	91.70	94.80
SM-ESSN (UNB ISCX IDS 2012) [22]	99.00	99.12	98.76	NA
FMDADM (Edge-IIoT) [23]	99.53	99.13	98.38	99.13
WFL (CAIDA) [25]	98.85	98.13	99.27	94.21
SAT-IDS (InSDN) [33]	99.39	99.40	98.50	98.45
Proposed SDN-IDS	99.61	99.53	99.70	99.58

Table 4 represents the comparison of the SDN-IDS model's results compared with the other current models discussed in the related works section. This comparison is carried out to validate the research model's efficiency and superiority in detecting attacks in SDN. As indicated in this comparison, the research model outperformed all the other models in terms of every parameter with consistent performance. Figures 7 to 10 represent the graphical plots of the comparison of the results starting from accuracy to f1-score.

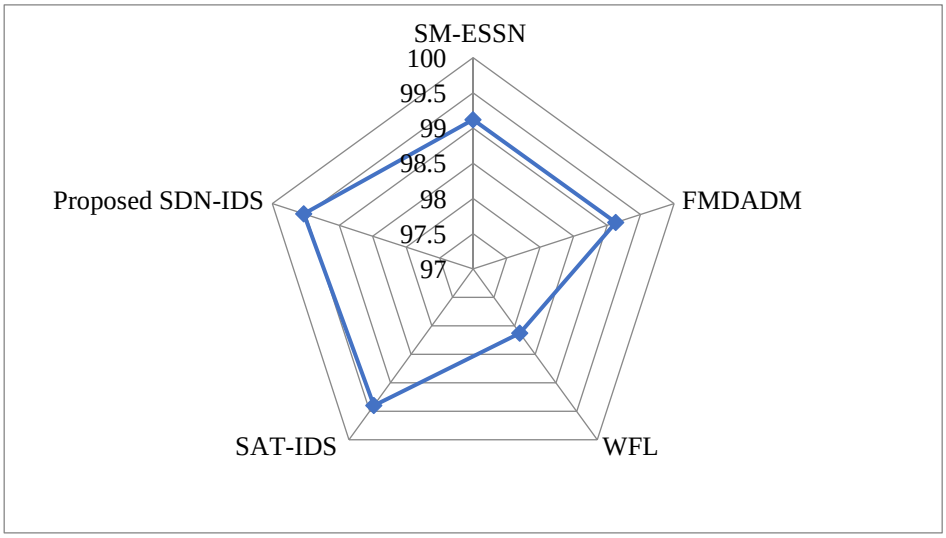

Fig. 8 Graphical plot of detection rate comparison

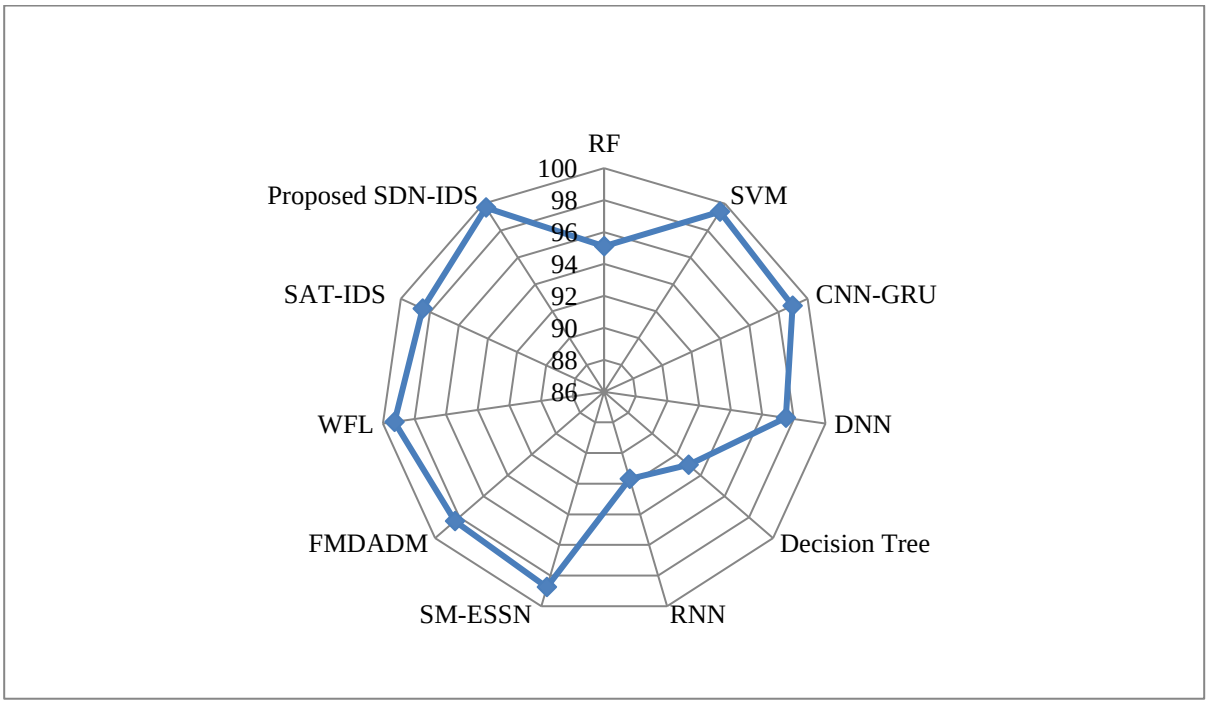

Fig. 9 Graphical plot of precision comparison

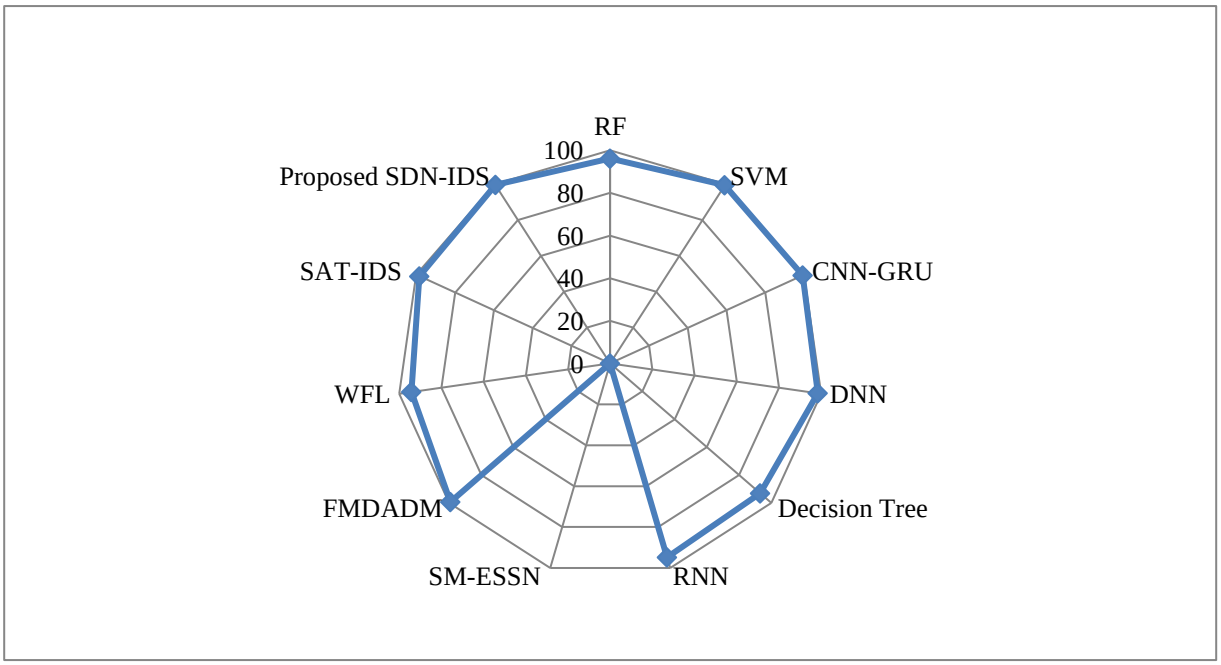

Fig. 10 Graphical plot of F1-score comparison

In this comparison, the SDN-IDS model's test results are used. All the models used in this comparison are mainly developed for detecting DDoS attacks in SDN but evaluated with different datasets as they are mentioned in the table. However, the research model is also developed for detecting DDoS attacks in SDN, the comparison is carried out. The research model attained 99.61% accuracy, which is 0.08% to 4.95% higher than the compared models. The FMDADM, SAT-IDS, SVM, and CNN-GRU models all have obtained an accuracy of over 99%. The least performed model was RNN, with 94.66%. The detection rate of the research model was 99.53%, which is 0.13% to 6.53% improved than the other models. The SVM and SAT-IDS models have a similar detection rate of 99.40%, and some models like

CNN-GRU, DNN, SM-ESSN, and FMDADM models have also obtained more than 99% detection rate. The least performed model was the decision tree, with 93%. The research model has attained a precision rate of 99.70%, which is 0.29% to 8% higher than the compared models. The SVM model has a close performance of 99.41% compared to the research model. The least performed model was RNN, with 91.70% in this comparison. The f1-score of the research model was 99.58%, which is 0.19% to 6.58% improved than the other models. The SVM, FMDADM, and CNN-GRU models have an f1-score of more than 99%. The least performed model was the decision tree, with 93%. Overall, the research model has outperformed all the compared models in terms of accuracy, detection rate, f1-score, and precision. Based on the obtained results and this comparison, it can be concluded that the proposed SDN-IDS model is effective and better at detecting and mitigating attacks in SDN.

The proposed SDN-IDS model exhibits several advantages. These include exceptional accuracy, high detection rates, precision, and F1 scores. Compared to existing models, it demonstrated superior performances in identifying attacks present in the SDN control layer. The model used DL and optimization techniques, resulting in robust identification and classification of attacks, effectively providing a reliable defense mechanism against network threats. Furthermore, the incorporation of feature selection using the BALO algorithm enhanced efficiency by focusing on relevant features. Additionally, BO was employed to fine-tune hyperparameters of ABiLSTM, effectively optimizing classification performance.

However, there are a few limitations to consider in this research. The limitations include the computational complexity inherent to DL models and the substantial computational resources required for both training and inference. The qualities and representativeness of a training data set can also impact the model's performance. This necessitates continuous updates and refinement to adapt to evolving attack patterns and network dynamics. Despite these limitations, the SDN-IDS model represents a significant advancement. It offers robust protection against DDoS attacks in SDN, and there are some considerations for further research and development in network security.

6. Conclusion

This research proposed a novel attack detection model called SDN-IDS for identifying attacks in SDN's control layer. More specifically, the research model is developed and trained to detect DDoS attacks in the SDN's control plane. The SDN-IDS model includes collecting data, preprocessing data, selecting optimal features and classification processes to perform the attack detection task. For data collection, the InSDN dataset was gathered and used for training and evaluating the model. The data set was split into a 70:30 ratio. In data preprocessing, data cleaning, data transformation, and normalization tasks were carried out. For normalization, the Min-Max normalization technique was applied. After preprocessing, the feature selection task was performed using the binary version of the ALO algorithm called BALO, in which the significant optimal features were selected using the binary vector representation. Based on the selected features, the ABiLSTM model performed the classification task to detect the attack from the input instances. The hyperparameters of the ABiLSTM were tuned by the BO technique to improve the model's performance in classifying the attacks. The SDN-IDS model was assessed in terms of detection rate, accuracy, f1-score, FAR, and precision. Based on this analysis, the model attained 99.61% accuracy, 99.53% detection rate, 99.70% precision, 99.58% f1-score, and 0.46% FAR. Overall, these findings indicated that the SDN-IDS model effectively detects and classifies DDoS attacks within the SDN control layer with high accuracy, detection rates, precision, and F1 scores while maintaining a low false positive rate. The SDN-IDS model outperformed comparably to other existing models in terms of each parameter. The future works for this research will include exploring real-time implementation of the SDN-IDS model in real environments, investigating its adaptability and scalability to evolving network threats, and integrating anomaly detection techniques to enhance its capability in detecting novel attack patterns.

Acknowledgments

The authors would like to thank the Department of Computer Science and Engineering, Saveetha School of Engineering, Saveetha Institute of Medical and Technical Sciences, Chennai, Tamil Nadu, India, for their support and motivation throughout this research.

References

[1] Naziya Aslam, Shashank Srivastava, and M.M. Gore, "A Comprehensive Analysis of Machine Learning and Deep Learning-Based Solution for DDoS Attacks Detections in SDN," *Arabian Journal for Science and Engineering*, vol. 49, pp. 3534-3574, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[2] Yinghao Su et al., "A Comprehensive Survey of Distributed Denial of Service Detection and Mitigation Technologies in Software-Defined Network," *Electronics*, vol. 13, no. 4, pp. 1-29, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[3] Lubna Fayez Eliyan, and Roberto Di Pietro, "DoS and DDoS Attacks in Software Defined Network: A Survey of Existing Solution and Research Challenges," *Future Generations Computers System*, vol. 122, pp. 149-171, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[4] Junjie Xie et al., "Control Plane of Software Defined Networks: A Survey," *Computers Communication*, vol. 67, pp. 1-10, 2015. [CrossRef] [Google Scholar] [Publisher Link]

[5] Roya Taheri, Habib Ahmed, and Engin Arslan, "Deep Learning for the Security of Software-Defined Networks: A Review," *Clusters Computing*, vol. 26, pp. 3088-3112, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[6] Abdullah Ahmed Bahashwan et al., "A Systematic Literature Review on Machine Learning and Deep Learning Approach for Detecting DDoS Attacks in Software-Defined Networking," *Sensors*, vol. 23, vol. 9, pp. 1-48, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[7] Qiao Yan, and F. Richard Yu, "Distributed Denial of Service Attacks in Software-Defined Networking with Cloud Computing," *IEEE Communication Magazine*, vol. 53, no. 4, pp. 52-59, 2015. [CrossRef] [Google Scholar] [Publisher Link]

[8] Sarabjeet Kaur, Amanpreet Kaur Sandhu, and Abhinav Bhandari, "Investigations of Application Layers DDoS Attacks in Legacy and Software-Defined Networks: A Comprehensive Review," *International Journals of Information Security*, vol. 22, no. 6, pp. 1949-1988, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[9] Zaheed Ahmed Bhuiyan et al., "On the (in)Security of the Control Plane of SDN Architecture: A Survey," *IEEE Access*, vol. 11, pp. 91550-91582, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[10] Heyu Wang, and Yixuan Li, "Overview of DDoS Attack Detection in Software-Defined Networks," *IEEE Access*, vol. 12, pp. 38351-38381, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[11] Kiran Fatima, Kanwal Zahoor, and Narmeen Zakaria Bawany, "SDN Control Plane Security: Attacks and Mitigation Techniques," *Proceeding of the 4th International Conferences on Networking, Information System & Security*, no. 32, pp. 1-6, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[12] Muhammad Shoaib Farooq, Shamyla Riaz, and Atif Alvi, "Security and Privacy Issues in Software-Defined Networking (SDN): A Systematic Literature Review," *Electronics*, vol. 12, no. 14, pp. 1-37, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[13] Ijaz Ahmad et al., "Security in Software Defined Network: A Survey," *IEEE Communication Survey & Tutorials*, vol. 17, no. 4, pp. 2317-2346, 2015. [CrossRef] [Google Scholar] [Publisher Link]

[14] Zhenpeng Liu et al., "A DDoS Detections Method Based on Feature Engineering and Machine Learning in Software-Defined Networks," *Sensors*, vol. 23, no. 13, pp. 1-24, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[15] Waheed G. Gadallah, Hosny M. Ibrahim, and Nagwa M. Omar, "A Deep Learning Technique to Detect Distributed Denial of Service Attacks in Software-Defined Networks," *Computer & Security*, vol. 137, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[16] Ravindra Kumar Chouhan, Mithilesh Atulkar, and Naresh Kumar Nagwani, "A Framework to Detect DDoS Attack in Ryu Controller Based Software Defined Networks Using Feature Extraction and Classification," *Applied Intelligence*, vol. 53, pp. 4268-4288, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[17] Jin Wang, Liping Wang, and Ruiqing Wang, "A Method of DDoS Attacks Detection and Mitigation for the Comprehensive Coordinated Protection of SDN Controllers," *Entropy*, vol. 25, no. 8, pp. 1-26, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[18] Pinkey Chauhan, and Mithilesh Atulkar, "An Efficient Centralized DDoS Attack Detection Approach for Software Defined Internet of Things," *The Journal of Supercomputing*, vol. 79, pp. 10386-10422, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[19] Mohammed Mujib Alshahrani, "A Secure and Intelligent Software-Defined Networking Framework for Future Smart Cities to Prevent DDoS Attack," *Applied Sciences*, vol. 13, no. 17, pp. 1-16, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[20] Amran Mansoor et al., "Deep Learning-Based Approach for Detecting DDoS Attack on Software-Defined Networking Controller," *Systems*, vol. 11, no. 6, pp. 1-21, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[21] Menghao Zhang et al., "Control Plane Reflection Attacks and Defences in Software-Defined Networks," *IEEE/ACM Transaction on Networking*, vol. 29, no. 2, pp. 623-636, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[22] Panem Charanarur et al., "Designs Optimization-Based Software-Defined Networking Scheme for Detecting and Preventing Attacks," *Multimedia Tool and Applications*, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[23] Walid I. Khedr, Ameer E. Gouda, and Ehab R. Mohamed, "FMDADM: A Multi-Layered DDoS Attack Detection and Mitigation Framework Using Machines Learning for Stateful SDN-Based IoT Networks," *IEEE Access*, vol. 11, pp. 28934-28954, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[24] Hani Elubeyd, and Derya Yiltas-Kaplan, "Hybrid Deep Learning Approach for Automatic DoS/DDoS Attack Detections in Software-Defined Networks," *Applied Sciences*, vol. 13, no. 6, pp. 1-22, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[25] Muhammad Nadeem Ali et al., "Low-Rate DDoS Detection Using Weighted Federated Learning in SDN Control Plane in IoT Network," *Applied Sciences*, vol. 13, no. 3, pp. 1-21, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[26] Mahmoud Said Elsayed, Nhien-An Le-Khac, and Anca D. Jurcut, "InSDN: A Novel SDN Intrusion Dataset," *IEEE Access*, vol. 8, pp. 165263-165284, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[27] Laith Abualigah et al., "Ant Lion Optimizer: A Comprehensive Survey of Its Variants and Applications," *Archive of Computational Method in Engineering*, vol. 28, pp. 1397-1416, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[28] Majdi M. Mafarja, and Seyedali Mirjalili, "Hybrid Binary Ant Lion Optimizer with Rough Set and Approximate Entropy Reducts for Feature Selection," *Soft Computing*, vol. 23, no. 15, pp. 6249-6265, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[29] Jianfeng Deng, Lianglun Cheng, and Zhuowei Wang, "Attentions-based BiLSTM Fused CNN with Gating Mechanism Model for Chinese Long Text Classification," *Computer Speech & Language*, vol. 68, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[30] E. Emary et al., "Binary Ant Lion Approaches for Feature Selection," *Neurocomputing*, vol. 213, pp. 54-65, 2016. [CrossRef] [Google Scholar] [Publisher Link]

[31] Guixian Xu et al., "Aspect-Level Sentiment Classification Based on Attention-BiLSTM Model and Transfer Learning," *Knowledge-Based System*, vol. 245, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[32] A. Helen Victoria, and G. Maragatham, "Automatic Tuning of Hyperparameters Using Bayesian Optimization," *Evolving System*, vol. 12, no. 1, pp. 217-223, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[33] Rania A. Elsayed et al., "Securing IoT and SDN System Using Deep-Learning Based Automatic Intrusion Detection," *Ain Shams Engineering Journal*, vol. 14, no. 10, pp. 1-13, 2023. [CrossRef] [Google Scholar] [Publisher Link]