Computers, Materials & Continua
DOI: 10.32604/cmc.2022.028287
Article

Tech Science Press

HDLIDP: A Hybrid Deep Learning Intrusion Detection and Prevention Framework

Magdy M. Fadel[1,*], Sally M. El-Ghamrawy[2], Amr M. T. Ali-Eldin[1], Mohammed K. Hassan[3] and Ali I. El-Desoky[1]

[1]Computer Engineering and Systems Department, Faculty of Engineering, Mansoura University, Mansoura, 35516, DK, Egypt
[2]Head of Communications and Computer Engineering Department, MISR Higher Institute for Engineering and Technology, Mansoura, 35111, DK, Egypt
[3]Mechatronics Department, Faculty of Engineering, Horus University in Egypt (HUE), New Damietta, 34517, DT, Egypt
*Corresponding Author: Magdy M. Fadel. Email: mfares@mans.edu.eg
Received: 07 February 2022; Accepted: 25 March 2022

Abstract: Distributed denial-of-service (DDoS) attacks are designed to interrupt network services such as email servers and webpages in traditional computer networks. Furthermore, the enormous number of connected devices makes it difficult to operate such a network effectively. Software defined networks (SDN) are networks that are managed through a centralized control system, according to researchers. This controller is the brain of any SDN, composing the forwarding table of all data plane network switches. Despite the advantages of SDN controllers, DDoS attacks are easier to perpetrate than on traditional networks. Because the controller is a single point of failure, if it fails, the entire network will fail. This paper offers a Hybrid Deep Learning Intrusion Detection and Prevention (HDLIDP) framework, which blends signature-based and deep learning neural networks to detect and prevent intrusions. This framework improves detection accuracy while addressing all of the aforementioned problems. To validate the framework, experiments are done on both traditional and SDN datasets; the findings demonstrate a significant improvement in classification accuracy.

Keywords: Software defined networks (SDN); distributed denial of service attack (DDoS); signature-based detection; whale optimization algorism (WOA); deep learning neural network classifier

1 Introduction

Nowadays, the increasing use of Internet services like, data centers, electronic trade and cloud computing [1,2], causes computer network's size increases drastically and becomes hardly managed. Software defined networks (SDN) with its centralized management [3,4], dynamic and programmable architecture becomes more suitable for huge networks rather than using traditional computer networks. The general working strategy of Distributed Denial of Service (DDoS) attacks depends on sending an enormous number of packets to network resources to hamper or even block the reachability of legitimate users [5,6]. However, by deeply studying the structure of both computer networks architecture (traditional and SDN), DDoS attacks specific to SDN characterized by some points. First, SDN attacks aim to exhaust only the network's controller, while traditional networks have many points where attacks can be launched [7]. Second, the attacking packets usually pretend to have fake destination IP addresses, but in traditional networks, the destination IP addresses should be the IP of the targeted server to down [7].

Most of the detection and defense techniques used with SDN are literally a transplanting of traditional network techniques without taking in account the own characteristics of SDN environments. The processes of attack detection and defense are implemented on the SDN controller which increases computation overhead on the processor as well as the communication between SDN controller and switches (south bound) [7]. Several optimization techniques have been deployed to overcome these problems, as Genetic Algorithm (GA) [8], Firefly Algorithm (FF) [9], Particle Swarm Optimization (PSO) [10] and Whale Optimization Algorithm (WOA) [11]. Many of them have their limitations; such GA is more complex, depends on the initial population, and may fail to parameter convergence [8]. PSO has a poor control on discrete optimization problems and easy falls in local optima [10]. Due to these limitations, many hybridized and improved techniques have been applied to the original versions of Machine Learning (ML) to enhance their performance.

This paper deploys the detection of suspicious traffic at the data plane (switches) to alleviate both processing and communication overhead. The process of classifying suspicious packets executed at the controller plane is carried out by two techniques, signature-based [5,12] and deep learning-based to improve the accuracy and reduce the time of the classification process [13–15]. In signature-based technique every packet passes through the network have a unique pattern (signature) that is composed by intermediate routers while traversing through the network [16]. Signatures of malicious packets are stored in an attack signatures database, to be used later for identifying any attack packet that has a signature included in it. This technique has a high accuracy and a low false negative rate, but can't detect new (day zero) attacking packets that are not involved in the attack signatures database [17].

An optimized Neural Network (NN) with a set of optimal extracted features is used to classify and detect new offensive packets as part of the proposed deep learning technique. Combining both techniques allows the network to learn new attack patterns and append them to the attack signature database automatically for future use [13]. The contribution of this paper is as follows:

- Developing DDoS attack detection system that uses both signature-based and deep learning techniques to enhance the detection accuracy in SDN networks.
- Using the Neural Network to select the effective traffic feature set and to tune it (number of hidden layers and number of neurons in each layer), for decreasing the detection time and increasing the accuracy of the detection process.
- Validating the proposed framework using both traditional and SDN datasets to ensure it can handle both environments.

The remainder of this paper is organized as follow: the next section gives an overview of the related work, Section 3 describes the proposed framework in detail, Section 4 presents the experimental results and performance evaluation and finally Section 5 concludes the paper.

2 Related Works

Different detection and defense approaches are implemented by research community of defense since the first reported DDoS attacks in 1999 [18]. This section discusses two integrated categories of anomaly-based detection techniques, statistical approaches and artificial intelligence approaches.

2.1 Statistical Detection Approaches

Also called entropy-based approaches, entropy is the measure of the randomness in a dataset. Since every feature of normal network traffic has a special distribution pattern, for example a balance between the count of source and destination IP addresses in normal flow. This entropy pattern will deviate in attack flows since the number of destination IP addresses greatly increases than the number of source IP addresses. Despite this approach is characterized by a fast response and a low computation overhead of processing a large traffic volume. Its detection accuracy is greatly affected by the proper selection of the threshold value for a certain traffic feature to reduce the ratios of false positive and false negative.

In [19] authors proposed a hybrid system for attack detection that merges both entropy and traffic volume characteristics, which offers good results than using each technique alone. Kalkan et al. [20] introduced a joint entropy-based scoring system (JESS) to defend attacks in SDN environments, by utilizing Joint entropy they can defend even unfamiliar attacks efficiently. Lima et al. [21] suggested a system based on statistical analysis of traffic entropy in SDN environments.

Wang et al. [22] proposed a flow statistics process in SDN switches, then performed lightweight entropy-based detection model executed in the edge switches to reduce the communication overhead between the data plane and controller plane. Ahmed et al. [23] introduced a new structure called application fingerprints to express packet attributes and traffic flow level statistics to differentiate legitimate packets from attack packets. However, this approach is not proper for online systems, since some flow attributes such total bytes, number of packets between source and destination and flow duration cannot compute their statistics while gathering them. In [24] authors introduced new hybrid approaches where flow level statistics or entropy based are combined with some techniques of Machine Learning (ML) or Artificial Neural Networks (ANN) to overcome some limitations of flow statistics approaches. The ML and ANN mechanisms will be introduced in detail in the next subsection.

2.2 Artificial Intelligence (AI) Detection Approaches

Buczak et al. [25] introduced summaries for different methods of Machine Learning (ML) and Data Mining (DM) used for attack detection, such Support Vector Machine (SVM), k-Nearest Neighbor (k-NN), Random Forest (RF), etc. over recent years, with the abundance of real network traffic datasets. These methods perform better results in traffic classification field.

He et al. [26] suggested a new source side (active defense) machine learning technique instead of defending attacks at destination side (passive defense) preventing malicious network flows from being sent outside the attacking network. This showed a high accuracy and low false positive rates. Hoon et al. [27] introduced the concept of dataset feature selection which reduces the feature engineering processes and increasing classification accuracy. Few papers addressed the implementation of Deep Learning (DL) approaches in attack detection; authors in [28] combined the entropy-based techniques with DL methods to easily control the problem of setting accurate threshold. Conducted experiments showed a high accuracy. Yin et al. [29] proposed a deep learning attack detection model deploying a Recurrent Neural Network (RNN) to perform binary and multiclass classification. They compared its performance with a set of known Machine Learning (ML) models such SVM, Random Forest (RF) and Artificial Neural Networks (ANN). Their proposed model showed a higher performance from point of accurate classification. Wu et al. [30] introduced a multiclass Convolutional Neural Network (CNN) intrusion detection system, deploying CNN to select highly related features from the massive data gathered to improve the classification accuracy and reduce computation overload. Over traditional ML algorithms, CNN showed better performance. Kwon et al. [31] studied the performance of three different models of CNN (shallow, moderate and deep) to check the effect of depth of the CNN to the detection performance. The models are verified using two datasets NSL-KDD [32] and MAWILab [33] showing that shallow CNN model with a single convolutional layer and single maximum pooling layer performed best.

Authors in [34] converted the suspicious traffic traces into arrays that contain flow features by combining both CNNs and RNNs, testing their model on ISCX2012 dataset [35] showing good results from point of reducing classification error from 7.517% to 2.103% relative to conventional machine learning algorithms. Despite the efficiency of deep learning algorithms for detecting DDoS attacks, they are time-consuming, resource-intensive, and require large numbers of model parameters to be trained.

3 The Proposed Hybrid Deep Learning Intrusion Detection and Prevention (HDLIDP) Framework

Traditional networks' routers are assigned many customary duties like determining packets routes, assigning priorities, carrying out policies specified by the network administrator and many others, so they cannot detect and respond to DDoS attacks automatically. By leveraging SDN architecture, these attacks can be processed automatically and yield a fast and accurate response. Fig. 1 illustrates that.

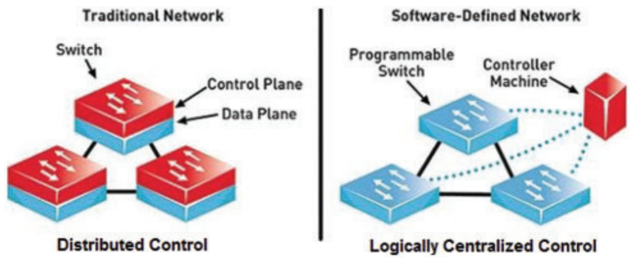

Figure 1: Traditional *vs*. SDN network architectures

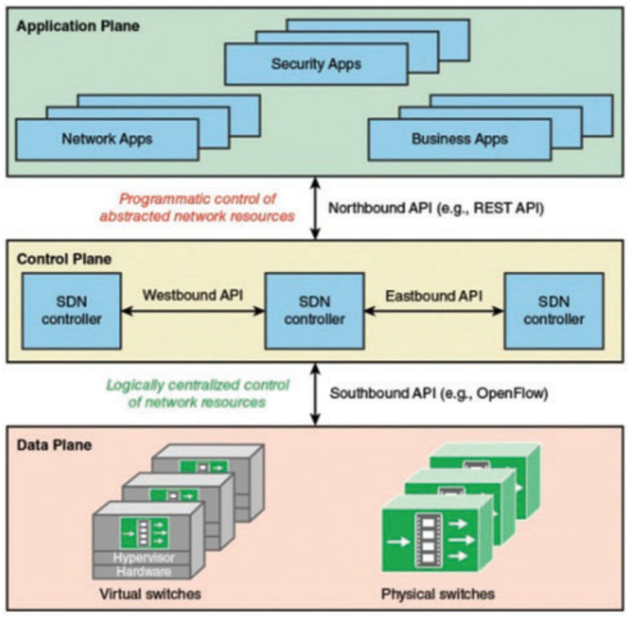

Figure 2: SDN three layers architecture

Three layers SDN architecture is show in Fig. 2, the switching task is split between a data layer and a control layer that are implemented on separate devices. The data plane is mainly responsible for forwarding network packets, while the brain or control plane performs all intelligent tasks in the network. In standard SDN, the control layer performs both the attack detection and defense tasks this may increase the controller's CPU utilization and communication workload through southbound interface, observe hourly the traffic flowing through switches to detect the DDoS attack.

The proposed framework (HDLIDP) may help to overcome this defect. It is made up of two layers, Data Layer Detection (DLD) and Control Layer Defense (CLD) as depicted in Fig. 3.

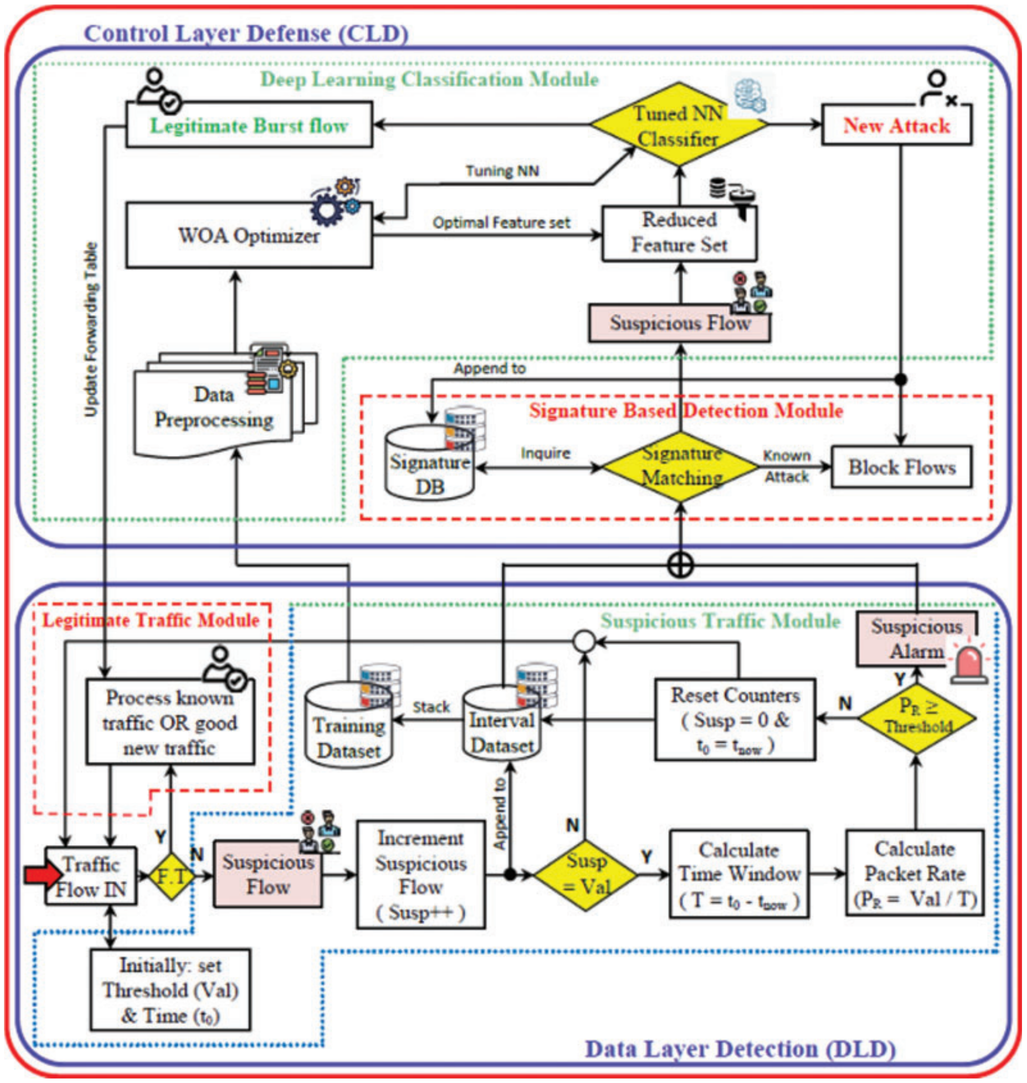
Control Layer Defense (CLD)
Deep Learning Classification Module
Legitimate Burst flow
Tuned NN Classifier
New Attack
Tuning NN
WOA Optimizer
Optimal Feature set
Reduced Feature Set
Suspicious Flow
Update Forwarding Table
Data Preprocessing
Append to
Signature Based Detection Module
Signature DB
Inquire
Signature Matching
Known Attack
Block Flows
Legitimate Traffic Module
Suspicious Traffic Module
Suspicious Alarm
Process known traffic OR good new traffic
Training Dataset
Stack
Interval Dataset
Reset Counters (Susp = 0 & t0 = tnow)
N
Y
PR ≥ Threshold
Traffic Flow IN
F.T
Suspicious Flow
Increment Suspicious Flow (Susp++)
Append to
Susp = Val
Calculate Time Window (T = t0 - tnow)
Calculate Packet Rate (PR = Val / T)
Initially: set Threshold (Val) & Time (t0)
Data Layer Detection (DLD)

Figure 3: The proposed (HDLIDP) framework

3.1 Data Layer Detection (DLD)

This layer detects any suspicious flows and raises an alarm to the control layer for accurately classifying whether it is an actual DDoS attack or legitimate burst traffic then carrying out proper response. This way the load on CPU Controller and the traffic overload through southbound interface may decrease drastically, it is designed as two modules:

3.1.1 Legitimate Traffic Module

Packets in SDN arrive first at the data plane devices, these packets may be classified into four categories: Known traffic that is already found in the switch's forwarding table; hence it is forwarded to its proper destination. The second type is a new legitimate traffic, when the switch does not find matching entries in its forwarding table, and after sending a pktIN message to the controller it will get a reply with a suitable forwarding route to be registered in its forwarding table for future use. The other two categories occur when the switch receives a suspicious packet. Such a packet does not have a matching entry in the forwarding table, and the controller does not able to determine its forwarding route, due to tampering in the source and/or destination IP addresses of it. All DDoS detection systems are concerned about the arrival rate of the last two categories of received packets (suspicious).

3.1.2 Suspicious Traffic Module

The maximum packets counter method is utilized at the data plane to calculate the suspicious packet's arrival rate within a predefined time window. In the framework, when a switch classifies the received packet as a suspicious flow, the suspicious flow counter is incremented (Susp++) and its features appended in both current interval and training datasets. Then, the value of the Susp variable is compared with the value of predefined adaptive maximum attacking packets (Val). The detection is in safe condition if Susp value is less than Val, so dropping this packet and processing any new incoming one.

Otherwise, if Susp equals or exceeds the predefined value (Val), the detection system calculates the time window, packets' arrival rate (PR) and initializes all counters. If the packets' arrival rate (PR) exceeds the predefined value, a suspicious alarm is raised to the control layer.

3.2 Control Layer Defense (CLD):

This layer is composed of two modules, Signature Based Detection and Deep Learning Classification.

3.2.1 Signature Based Detection Module

Traceback enabled routers utilize one of two techniques to insert their identification ID numbers in the packet's header, Deterministic Packet Marking (DPM) [36] or Probabilistic Packet Marking (PPM) [37]. The accumulation of all IDs along the packet's path forms what is called path's signature. It is used to characterize the exact route of received packet regardless of its source IP address that could be easily forged. After isolating attacking traffic, its signature is stored in attack signature database for future use.

3.2.2 Deep Learning Classification Module

Is the conclusive classification stage that identifies the fourth category of suspicious packets. Because of the large-scale exploration of the search space, the simplicity of implementation, the wide range of applications and its potential development [10], the Whale Optimization Algorithm (WOA) is used to select the optimal set of features and to tune parameters of the classification Neural Network.

The WOA is an optimization algorithm that mimics the hunting mechanism of humpback whales in nature. Whales prefer to hunt in a group of whales, initially every whale searches globally for a prey in a random direction, this process is called exploration phase, the following mathematical model illustrates it.

$$\vec{D} = \left|\vec{C}\vec{X}_{rand} - \vec{X}\right| \tag{1}$$

$$\vec{X}(t+1) = \vec{X}_{rand} - \vec{A}\vec{D} \tag{2}$$

where: $\vec{X}_{rand}$ is a random position vector (random whale) and t indicates the current iteration.

Since the optimum position of the prey in the search space is not known previously, the group of whales will communicate to identify the current best elected solution. The other whales will update their direction towards the current elected whale; this step is called exploitation phase, modeled as follows:

$$\vec{X}(t+1) = \vec{D}' e^{bt} \cos(2\pi t) + \vec{X}^*(t) \tag{3}$$

where: $\vec{D}' = \left|\vec{X}^*(t) - \vec{X}(t)\right|$ indicating the i^th of whale the prey (best solution obtained so far), b is constant defining the shape of legitimate spiral and t is a random number in [−1,1].

$$\vec{X}(t+1) = \begin{cases} \vec{X}^*(t) - \vec{A}\vec{D}, & p < 0.5 \\ \vec{D}' e^{bt} \cos(2\pi t) + \vec{X}^*(t), & p \geq 0.5 \end{cases} \tag{4}$$

where: p is a random number in [0,1].

In encircling prey, after defining the best search agent, the other search agents will try to update their positions towards the best search agent. This behavior is represented by the following equations:

$$\vec{D} = \left|\vec{C}.\vec{X}_P(t) - \vec{X}(t)\right| \tag{5}$$

$$\vec{X}(t+1) = \vec{X}_P(t) - \vec{A}.\vec{D} \tag{6}$$

where: $\vec{X}_P$ is the position vector of the prey, $\vec{X}$ is the position vector of a whale and $\vec{A}$ and $\vec{C}$ are coefficient vectors.

The pseudo of WOA is shown in Algorithm 1.

Algorithm 1: Pseudo code of selecting optimal set of traffic features using WOA and NN as the fitness function

```
Input: The full set of network traffic features
Output: The optimal set of features
Initialize the algorithm parameters (SearchAgentsNo = 50, dim = 4, LB = 0, UB = 1023 and MaxIter = 500) as shown in Tab. 1.
Initialize the relevant parameters of the WOA (a, A, C, I, p and the positions X of whales)
1.  Each Whale has a position of 4 random values (ranging from LB to UB) each of 10 bits (= 40 bits) representing the selected set from network traffic 40 features
2.  StartTime = Time( )
3.  Calculate the Optimization (lowest error value) of each Whale in the population of 40 whales using the neural network and its input as the Whale position (or selected set of features)
4.  While ($t <$ MaxIter)
5.      For each Whale (from 50 Whale)
6.    Update a, A, C, I and p
```

(Continued)

Algorithm 1: Continued

```
7.  **If (p < 0.5)**
8.    **If (|A| < 1)**
9.      Update the position of the current Whale by Eq. (6) (i.e., encircling prey)
10.     **Else (i.e., |A| ≥ 1)**
11.       Select a random search agent (X_rand)
12.       Update the position of the current Whale by Eq. (2) (i.e., exploration phase)
13.     **End If**
14.   **Else (i.e., p ≥ 0.5)**
15.     Update the position of the current Whale by Eq. (3) (i.e., exploitation phase)
16.   **End If**
17. **End For**
18.   Check if the position (features value > 40) of any Whale goes beyond the LB or UB and amend it
19.   Calculate the optimization of each Whale using the neural network
20.   Update X∗ (the set of features giving the lowest error value)
21.   t = t +1
22. **End While**
23. Execution Time = StartTime − Time( )
24. Return X∗
```

Table 1: Configuration values for the WOA optimizer

Configuration	Value
MaxIter	500
SearchAgentsNo	50
Dimension	No. of selected features from the dataset
No. of run repetitions	20

Fig. 4 represents the four stages deployed to determine the status of the received packet. The first two stages (searching the switch's forwarding table and inquiring the controller to specify the packet forwarding path) are performed by any standard SDN environment. While the other two stages (search attack signature DB and a deep learning classification) are appended. Algorithm 2 and the logic diagram of the received packet flow status in Fig. 4 clarify the classification process in detail.

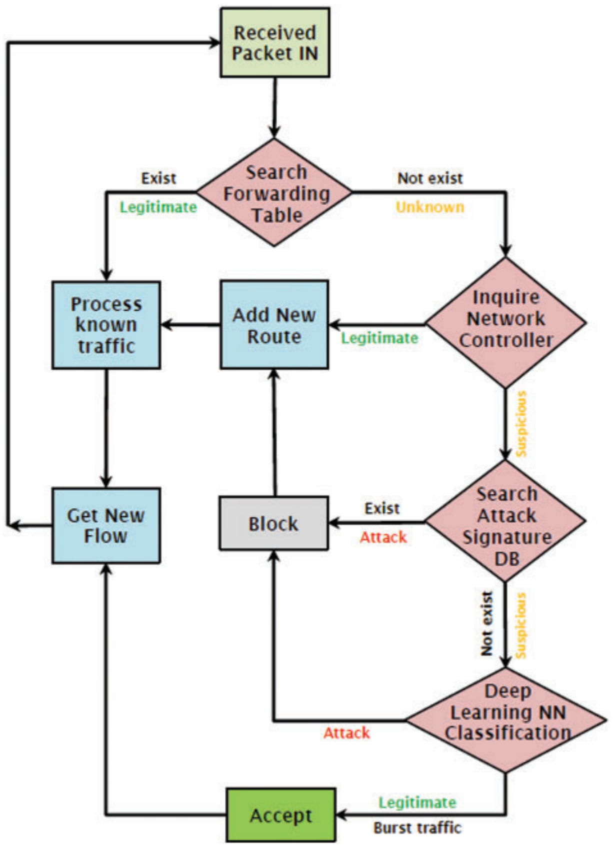

Figure 4: Received packet flow status

Algorithm 2: The pseudo code of the proposed framework after tuning the Neural Network

```
Input: received packet at the network switch.
Output: classifying packet type to correctly direct it.
1.  Stage 1: Search Forwarding Table:
2.  Search for the packet in the Switch forwarding table.
3.  If (exist)
4.      Old and legitimate packet
5.      Forward it to its proper destination
6.      Get a new packet to process
7. Else
8.      New packet
9.      Inquire: the network controller (Stage 2:) to determine its route (if possible)
10.     End if
11.     End Stage 1:
12.
13.   Stage 2: Inquire Network Controller:
```

(Continued)

Algorithm 2: Continued

```
14. Determine the route of the new packet (if possible)
15. **If** (route found)
16.    New and normal packet
17.    Send its route to the inquiring switch
18.    The switch adds this route to its forwarding table for future use
19.    Forward the received packet
20.    Get a new packet to process
21. **Else**
22.    Suspicious packet alarm
23.    **Search:** Attack signature DB (Stage 3:) to determine if it is an attack or suspicious
24. **End if**
25. **End Stage 2:**
26.
27. **Stage 3: Search attack signature DB:**
28. Search the existence of the packet's signature in the signature Database
29. **If** (signature exist)
30.    Old and attack packet
31.    Send it back to the inquiring switch to:
32.       • Block it
33.       • Add it to the forwarding table for future use
34.    Get a new packet to process
35. **Else**
36.    Suspicious packet trigger
37.    Conduct the Deep learning NN classifier (Stage 4:) to determine if it is an attack or legitimate
       burst traffic
38. **End if**
39. **End Stage 3:**
40.
41. **Stage 4: Deep Learning NN Classifier:**
42. Classify the new packet
43. **If** (legitimate burst traffic)
44.    New and legitimate packet
45. **Else**
46.    New but attack packet
47. **End if**
48. Send it to the inquiring switch
49. The switch adds its state (attack or legitimate) to the forwarding table for future use
50. Forward it if a legitimate packet
51. Get a new packet to process
52. End Stage 4:
```

4 Experiments and Evaluation

A hybrid classification algorithm composed of Whale Optimization Algorithm (WOA) in Tab. 1 and a tuned Neural Network (NN) in Tab. 2 is used in experiments. WOA is used to select the most effective set of features from the used datasets, whereas the tuned ANN is used to accurately classify the newly unknown suspicious packets, reducing the computation overhead and increasing the IDS classification accuracy.

Table 2: Parameters of the NN structure

Structure Parameters	Value
No. of hidden layers	1 or 2 or 3
No. of neurons	10
Biases	Random
Activation function	TanH
Initial weights	Default

4.1 Benchmark Dataset

The framework is evaluated using three datasets, NSL-KDD and CSE-CIC-IDS2018 are the most traditional network's datasets commonly used, where the third is SDN specific dataset. First, NSL-KDD dataset contains 4,898,430 records each of 41 features. Feature no. 42 is the records' labels, which may be Normal or attack; attack records are categorized into 4 types DoS, Probe, R2L or U2R.

Despite its simplicity, NSL-KDD dataset is not the ideal representation of the actual network model, so CIC-IDS2018 dataset is used to accurately evaluate any IDS since it represents real attacks. It has 83 features, of 2,830,540 distinct records, and categorized in the label field to 15 classes.

Although the characteristics of CIC-IDS2018 dataset have some weaknesses, first with this huge number of records (3,119,345) and 83 features each, enormous loading and processing overhead is required. Second, it contains some missing data. The last demerit is class imbalance problem [38], in which different attacking classes do not have equal number of instances, some are represented by a large number (BENIGN = 2,359,087 instances) others have few number (HeartBleed = 11 instances). So, when using this dataset for training classifiers or detectors it will make the classifiers biasing toward the majority class [38] degrading the classifier's accuracy with a higher false ratio. Alleviating methods for these shortcomings will be introduced in the Data preprocessing subsection.

Since the architecture of traditional network differs from that of SDN, which results in a major difference in the feature sets of the data gathered from both. Mininet emulator [39] software has been used to create a realistic virtual network, 23 features in the designed SDN topology with the total number and different categories used in following experiments.

4.2 Data Preprocessing

Network collected dataset may include some error values like duplicate, infinity, missing or categorical data, this may cause classification problems. These error values should be eliminated or mitigated before training and testing phase. Duplicate records should be removed, deleting records having outlier's values. Techniques like one-hot encoder are used to convert categorical data into numeric values.

Feature scaling methods (Normalization and Standardization) [40]: features having values of varying degrees of magnitude, may hurdle the performance of some machine learning algorithms especially those types using gradient descent as optimization techniques. So, these scaling methods may be used to scale their values between 0 and 1, or +a and –a.

Data imbalance reduction [41]: where some features are highly underrepresented, causing the classifier to bias towards the majority features. Many techniques are designed to handle class imbalance problem, one of them deployed in this paper is class relabeling. By either splitting the majority classes into more classes or merging some minority classes to form one class.

4.3 Evaluation Metrics

Evaluating the trained model's performance can be done using the confusion matrix and advanced evaluation metrics are shown in Fig. 5.

Confusion matrix		**Predicted Class**		**Advanced evaluation metrics**
		Positive (P)	**Negative (F)**	
Actual Class	**Positive (P)**	True Positive (TP)	False Negative (FN)	Senstivity (Recall) $= \frac{TP}{(TP + FN)}$
	Negative (F)	False Positive (FP)	True Negative (TN)	Specificity(Recall) $= \frac{TN}{(TN + FP)}$
		Precision $= \frac{TP}{(TP + FP)}$	F1 Score $= 2 \times \frac{Precision \times Recall}{Precision + Recall}$	Accuracy $= \frac{TP + TN}{(TP + FN + FP + TN)}$

Figure 5: Confusion matrix and evaluation metrics

True Positive (TP): denotes the no. of positive class correctly judged as positive. False Negative (FN): denotes the no. of positive class mistakenly judged as negative. False Positive (FP): denotes the no. of negative class mistakenly judged as positive. True Negative (TN): denotes the no. of negative class correctly judged as negative.

4.4 Experimental Results

Three experiments have been conducted to validate the performance of the (HDLIDP) framework.

Experiment 1: (NSL-KDD dataset): using a two hidden layers tuned ANN classifier.

Tab. 4 shows the advanced metrics obtained from the confusion matrix shown in Tab. 3. The average results of confusion matrix when utilizing two hidden layers are 97.903, 91.690, 98.271 and 94.798 for Accuracy, Precision, Recall and F1 Score, respectively.

Table 3: Confusion matrix of NSL-KDD classifier of two hidden layers

Actual class	Normal	491	19	5	1	0	516
	DoS	17	358	4	1	0	380
	Probe	3	1	86	0	1	91
	R2L	0	1	0	12	0	13
	U2R	0	0	0	0	11	11
	$\sum$	511	379	95	14	12	**1,011**
		Normal	DoS	Probe	R2L	U2R	$\sum$
		Predicted class					

Table 4: NSL-KDD metrics of the above confusion matrix

No.	New labels	Confusion matrix metrics				Advanced evaluation metrics (%)			
		TP	FN	FP	TN	Acc.	Pre.	Recall	F1
1	Normal	491	25	20	475	95.549	96.086	95.960	96.023
2	DoS	358	22	21	610	95.747	94.459	96.672	95.553
3	Probe	86	5	9	911	98.615	90.526	99.022	94.584
4	R2L	12	1	2	996	99.703	85.714	99.800	92.222
5	U2R	11	0	1	999	99.901	91.667	99.900	95.607
		Summation				489.515	458.452	491.354	473.989
		Average				**97.903**	**91.690**	**98.271**	**94.798**

Tab. 5 and Figs. 6 and 7 show the parameters of the ANN classification in case of single, two and three hidden layers compared with the Genetic Algorithm (GA) and Difficult Set Sampling Technique (DSSTE) algorithm. The DSSTE algorithm employs both Edited Nearest Neighbor (ENN) and K-Means clustering algorithms to reduce the data set's majority class for improving the classifier's training stage consequently enhances performance. The results show, using two hidden layers NN each contains maximum of 10 neurons provides best performance and approximately good time.

Table 5: NSL-KDD comparison results among different classifier structures

Neural network structure	NSL-KDD (%)				
	Acc.	Pre.	Recall	F1	Time(S)
1 Hidden layer	96.396	85.988	97.016	90.989	8
2 Hidden layer	**97.903**	**91.690**	**98.271**	**94.798**	**11**
3 Hidden layer	96.987	89.121	97.821	92.813	15
Genetic algorithm	79.564	76.154	78.841	76.783	21
DSSTE [42]	82.840	84.680	82.780	81.660	–

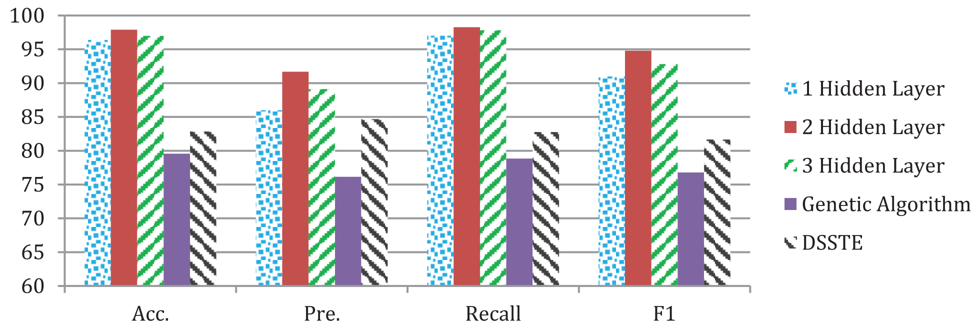

Figure 6: NSL-KDD comparison results among deep learning classifier with different hidden layers

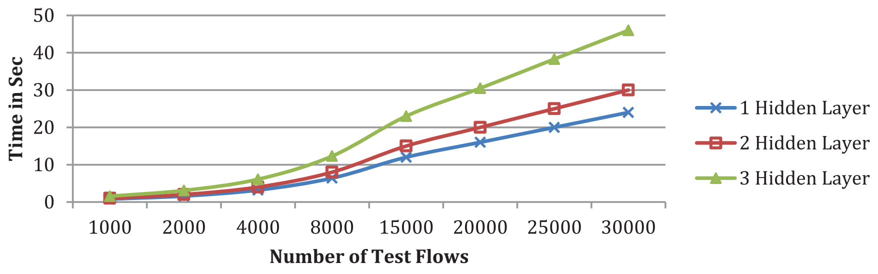

Figure 7: NSL-KDD Time comparison results among DL classifier with different hidden layers

Experiment 2: (CIC-IDS2018 dataset): using a two hidden layers tuned ANN classifier.

Tabs. 6 and 7 show the average results of confusion matrix as 99.849, 93.333, 99.761 and 96.325 for Accuracy, Precision, Recall and F1 Score, respectively.

Table 6: Confusion matrix of CIC-IDS2018 classifier of two hidden layers

Actual class	Normal	588,462	23	98	824	0	349	16	589,772
	Botnet ARES	0	490	0	0	1	0	0	491
	Brute Force	9	0	3,431	0	0	0	19	3,459
	DoS/DDoS	985	7	41	72,247	0	329	17	73,626
	Infiltration	0	0	0	0	9	0	0	9
	Port Scan	592	3	62	324	1	38,722	28	39,732
	Web Attack	0	1	0	1	0	3	540	545
	∑	590,048	524	3,632	73,396	11	39,403	620	**707,634**
		Normal	Botnet ARES	Brute Force	DoS/DDoS	Infiltra-tion	Port Scan	Web Attack	∑
		Predicted class							

Table 7: CIC-IDS2018 metrics of the above confusion matrix

No.	New labels	Confusion matrix metrics				Advanced evaluation metrics (%)			
		TP	FN	FP	TN	Acc.	Pre.	Recall	F1
1	Normal	588,462	1,310	1,586	116,276	99.591	99.731	98.654	99.190
2	Botnet ARES	490	1	34	707,109	99.995	93.511	99.995	96.644
3	Brute Force	3,431	28	201	703,974	99.968	94.466	99.971	97.141
4	DoS/DDoS	72,247	1,379	1,149	632,859	99.643	98.435	99.819	99.122
5	Infiltration	9	0	2	707,623	99.999	81.818	99.999	89.999
6	PortScan	38,722	1,010	681	667,221	99.761	98.272	99.898	99.078
7	Web Attack	540	5	80	707,009	99.988	87.097	99.989	93.099
		Summation				698.945	653.330	698.325	674.273
		Average				**99.849**	**93.333**	**99.761**	**96.325**

Since the framework has been designed to be deployed in real time world, so both performance accuracy and running time should be highly improved. From Tab. 8 and Figs. 8 and 9, show that the best accuracy is obtained when using two hidden layers NN, while the best running time obtained with a single hidden layer NN. According to the results obtained, it is advised to deploy the framework with two hidden layers NN as it has best accuracy and approximately good running time.

Table 8: CIC-IDS2018 Comparison results among different classifiers structure

Neural network structure	CIC-IDS2018 (%)				
	Acc.	Pre.	Recall	F1	Time(S)
1 Hidden layer	99.003	68.714	99.077	78.392	12
2 Hidden layer	**99.849**	93.333	**99.761**	**96.325**	**15**
3 Hidden layer	99.218	78.942	99.214	84.657	23
Genetic algorithm	93.154	71.458	94.345	76.845	34
DSSTE	96.990	**97.460**	96.970	97.040	–

In the next experiment, the proposed framework is evaluated using SDN dataset collected from the Mininet emulator and comparing its results with those of the framework introduced in [43].

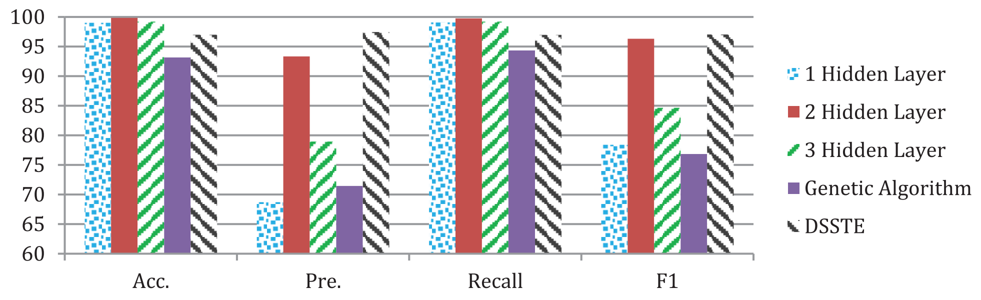

Figure 8: CIC-IDS2018 comparison results among DL classifier with different hidden layers

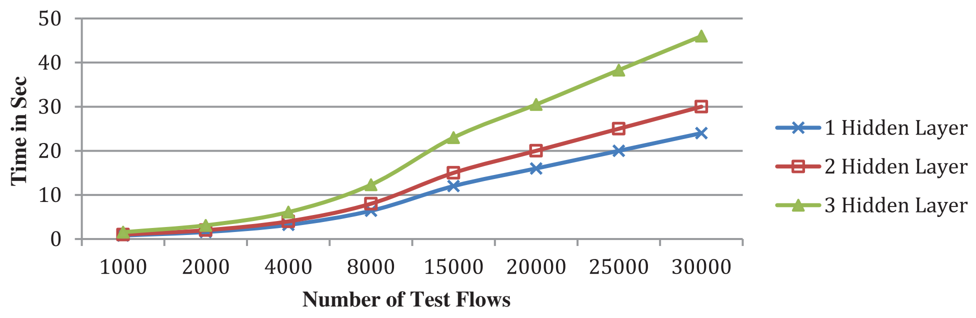

Figure 9: CIC-IDS time comparison results among DL classifier with different hidden layers

Experiment 3: (SDN dataset): the experiment has been done on a two hidden layers ANN classifier. Tabs. 9 and 10 shows the average values of Accuracy, Precision, Recall and F1 Score.

Table 9: Confusion matrix of SDN classifier of two hidden layers

Actual class	Benign ICMP	24,352	125	146	81	178	75	24,957
	Malicious ICMP	119	15,945	91	50	108	51	16,364
	Benign TCP	147	89	18,405	63	132	61	18,897
	Malicious TCP	75	48	56	10,264	71	25	10,539
	Benign UDP	184	122	138	73	22,179	76	22,772
	Malicious UDP	79	51	60	30	72	10,524	10,816
	∑	24,956	16,380	18,896	10,561	22,740	10,812	**104,345**
		Benign ICMP	Malicious ICMP	Benign TCP	Malicious TCP	Benign UDP	Malicious UDP	∑
		Predicted class						

Table 10: SDN metrics of the above confusion matrix

No.	Traffic labels	Confusion matrix metrics				Advanced evaluation metrics (%)			
		TP	FN	FP	TN	Acc.	Pre.	Recall	F1
1	Benign ICMP	24,352	605	604	78,784	98.841	97.580	99.239	98.403
2	Malicious ICMP	15,945	419	435	87,546	99.182	97.344	99.506	98.413
3	Benign TCP	18,405	492	491	84,957	99.058	97.402	99.425	98.403
4	Malicious TCP	10,264	275	297	93,509	99.452	97.188	99.683	98.420
5	Benign UDP	22,179	593	561	81,012	98.894	97.533	99.312	98.414
6	Malicious UDP	10,524	292	288	93,241	99.444	97.336	99.692	98.500
	Summation					594.871	584.383	596.857	590.553
	Average					**99.145**	**97.397**	**99.476**	**98.430**

Tab. 11 and Fig. 10 show a comparison in case of one, two and three hidden layers with the GA and Automated DDoS attack detection in SDN [43] framework.

Table 11: SDN Comparison results among different classifier structures

Neural network structure	NSL-KDD			
	Acc.	Pre.	Recall	F1
1 Hidden Layer	99.000	96.950	99.388	98.153
2 Hidden Layer	**99.145**	**97.397**	**99.476**	**98.430**
3 Hidden Layer	99.060	97.102	99.413	98.211
Genetic Algorithm	96.201	94.253	97.142	95.981
Automated detection [43]	98.800	98.270	98.180	97.650

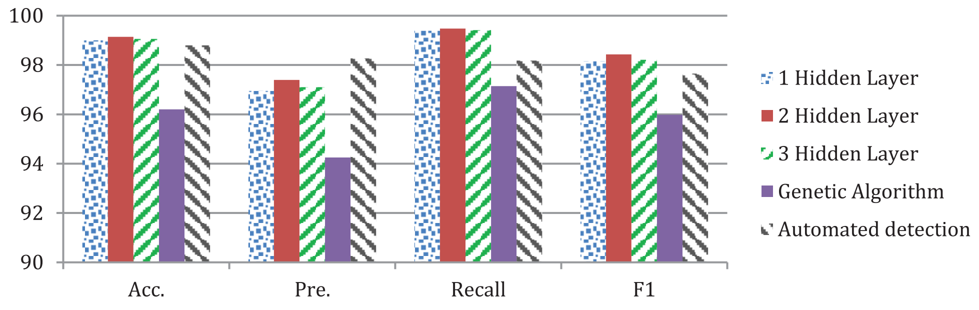

Figure 10: SDN comparison results among DL classifier with different hidden layers

5 Conclusions and Future Work

Despite the importance of computer networks and its different services, it is difficult to manage and secure a huge number of distributed devices. A Hybrid Deep Learning Intrusion Detection and Prevention framework (HDLIDP) that is suitable for use with SDN networks has been proposed in this paper. Signature-based and deep learning detection techniques have been deployed to improve framework performance. A signature based technique ensure that a packet is an attack, but not that it is legitimate, while deep learning technique classifies packets based on their type, if it is an attacker or not, successfully taking the needed action. Both techniques may improve attack detection accuracy and speed. The outcomes are determined by important factors such as classification accuracy and system responsiveness. A comprehensive study has been conducted using three datasets that have been applied to single, two, and three layers NN classifiers. Additionally, we cannot ignore the role played by using the WOA optimizer in selecting the effective set of dataset's features for improving the classification process. Results revealed the superiority of the proposed framework, especially in cases of double NN hidden layers. In future, the proposed framework could be improved by deploying different optimization algorithms and evaluated by using more SDN environment datasets.

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

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

References

[1] S. Azizi, N. Hashemi and A. Khonsari, "A flexible and high-performance data center network topology," *Supercomputing*, vol. 73, no. 4, pp. 1484–1503, 2017.

[2] M. Birje, P. Challagidad, R. H. Goudar and M. T. Tapale, "Cloud computing review: Concepts, technology, challenges and security," *International Journal of Cloud Computing*, vol. 6, no. 1, pp. 32–57, 2017.

[3] D. S. Rana, S. A. Dhondiyaland and S. K. Chamoli, "Software defined networking (SDN) challenges, issues and solution," *International Journal of Computer Science and Engineering*, vol. 7, no. 1, pp. 884–889, 2019.

[4] S. H. Haji, S. R. M. Zeebaree, R. H. Saeed, S. Y. Ameen, H. Shukur *et al.,* "Comparison of software defined networking with traditional networking," *Asian Journal of Computer Science and Information Technology*, vol. 9, no. 2, pp. 1–18, 2021.

[5] M. M. Fadel, A. I. El-Desoky, A. Y. Haikel and L. M. Labib, "A low-storage precise IP traceback technique based on packet marking and logging," *Oxford University Press, The Computer Journal*, vol. 59, no. 11, pp. 1581–1592, 2016.

[6] M. M. Fadel, M. Areed and A. I. El-Desoky, "A hybrid approach for detecting, preventing, and traceback DDoS attacks," *WSEAS Transactions on Computers*, vol. 11, no. 7, pp. 191–196, 2014.

[7] J. Singh and S. Behal, "Detection and mitigation of DDoS attacks in SDN: A comprehensive review, research challenges and future directions," *ElSevier Computer Science Review*, vol. 37, no. 2, pp. 1–25, 2020.

[8] S. Katoch, S. S. Chauhan and V. Kumar, "A review on genetic algorithm: Past, present, and future," *Multimedia Tools and Applications*, vol. 80, no. 17, pp. 8091–8126, 2021.

[9] W. A. Khan, N. N. Hamadneh, S. L. Tilahun and J. M. T. Ngnotchouye, "A review and comparative study of Firefly algorithm and its modified versions," in *Optimization Algorithms-Methods and Applications*, First ed., vol. 1. London, United Kingdom: IntechOpen Press, pp. 281–313, 2016.

[10] Z. Li, R. Tan and B. Ren, "Research on particle swarm optimization of variable parameter," in *Proc. Int, Conf. Advances on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC 2016)*, Lecture Notes on Data Engineering and Communications Technologies, Springer, Cham, Germany, vol. 1, pp. 25–33, 2016.

[11] S. Mirjalili and A. Lewis, "The whale optimization algorithm," *Advances in Engineering Software*, vol. 95, no. 12, pp. 51–67, 2016.

[12] P. Kaur, M. Kumar and A. Bhandari, "A review of detection approaches for distributed denial of service attacks," *Systems Science & Control Engineering*, vol. 5, no. 1, pp. 301–320, 2017.

[13] J. Xie, F. R. Yu, T. Huang, R. Xie, J. Liu *et al.,* "A survey of machine learning techniques applied to software defined networking (SDN): Research issues and challenges," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 1, pp. 393–430, 2019.

[14] W. Sun, G. Z. Dai, X. R. Zhang, X. Z. He and X. Chen, "TBE-Net: A three-branch embedding network with part-aware ability and feature complementary learning for vehicle re-identification," *IEEE Transactions on Intelligent Transportation Systems*, vol. 99, pp. 1–13, 2021.

[15] W. Sun, L. Dai, X. R. Zhang, P. S. Chang and X. Z. He, "RSOD: Real-time small object detection algorithm in UAV-based traffic monitoring," *Applied Intelligence*, vol. 92, no. 6, pp. 1–16, 2021.

[16] M. M. Fadel, "HDSL: A hybrid distributed single-packet low-storage IP traceback framework," *Mansoura Engineering Journal (MEJ)*, vol. 46, no. 4, pp. 75–89, 2021.

[17] A. Khraisat, I. Gondal, P. Vamplew and J. Kamruzzaman, "Survey of intrusion detection systems: Techniques, datasets and challenges," *Cybersecurity*, vol. 2, no. 20, pp. 1–22, 2019.

[18] P. J. Criscuolo, "Distributed denial of service, tribe flood network 2000, and stacheldraht, CIAC-2319," *Department of Energy Computer Incident Advisory Capability (CIAC), UCRLID-136939, Rev. 1*, vol. 1, pp. 1–18, 2000.

[19] P. D. Bojovic, I. Basicevic, S. Ocovaj and M. Popovic, "A practical approach to detection of distributed denial-of-service attacks using a hybrid detection method," *Computers and Electrical Engineering*, vol. 73, pp. 84–96, 2019.

[20] K. Kalkan, L. Altay, G. Gur and F. Alagoz, "JESS: Joint entropy-based DDoS defense scheme in SDN," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 10, pp. 2358–2372, 2018.

[21] N. A. S. Lima and M. P. Fernandez, "Towards an efficient DDoS detection scheme for software-defined networks," *IEEE Latin America Transactions*, vol. 16, no. 8, pp. 2296–2301, 2018.

[22] R. Wang, Z. Jia and L. Ju, "An entropy-based distributed DDoS detection mechanism in software-defined networking," in *Proc. IEEE Int. Conf. on Trust, Security and Privacy in Computing and Communications (TrustCom)*, Helsinki, Finland, pp. 310–317, 2015.

[23] M. E. Ahmed, S. Ullah and H. Kim, "Statistical application fingerprinting for DDoS attack mitigation," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 6, pp. 1471–1484, 2019.

[24] E. Min, J. Long, Q. Liu, J. Cui and W. Chen, "TR-IDS: Anomaly-based intrusion detection through text-convolutional neural network and random forest," *Security and Communication Networks*, vol. 2018, no. 1, pp. 1–9, 2018.

[25] A. L. Buczak and E. Guven, "A survey of data mining and machine learning methods for cyber security intrusion detection," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 2, pp. 1153–1176, 2016.

[26] Z. He, T. Zhang and R. B. Lee, "Machine learning based DDoS attack detection from source side in csloud," in *Proc. IEEE 4th Int. Conf. on Cyber Security and Cloud Computing (CSCloud)*, New York, NY, USA, pp. 114–120, 2017.

[27] K. S. Hoon, K. C. Yeo, S. Azam, B. Shunmugam and F. D. Boer, "Critical review of machine learning approaches to apply big data analytics in DDoS forensics," in *Proc. Int. Conf. on Computer Communication and Informatics (ICCCI)*, Coimbatore, India, pp. 1–5, 2018.

[28] A. Koay, A. Chen, I. Welch and W. K. G. Seah, "A new multi classifier system using entropy-based features in DDoS attack detection," in *Proc. IEEE Int. Conf. on Information Networking (ICOIN)*, Chiang Mai, Thailand, pp. 162–167, 2018.

[29] C. Yin, Y. Zhu, J. Fei and X. He, "A deep learning approach for intrusion detection using recurrent neural networks," *IEEE Access*, vol. 5, pp. 21954–21961, 2017.

[30] K. Wu, Z. Chen and W. Li, "A novel intrusion detection model for a massive network using convolutional neural networks," *IEEE Access*, vol. 6, pp. 50850–50859, 2018.

[31] D. Kwon, K. Natarajan, S. C. Suh, H. Kim and J. Kim, "An empirical study on network anomaly detection using convolutional neural networks," in *Proc. IEEE 38th Int. Conf. on Distributed Computing Systems (ICDCS)*, Vienna, Austria, pp. 1595–1598, 2018.

[32] University of New Brunswick benchmark dataset, [Accessed: 28-Dec-2021]. *Available:* https://www.unb.ca/cic/datasets/nsl.html.

[33] MAWILab dataset, [Accessed: 3-Jan-2022]. *Available:* http://www.fukuda-lab.org/mawilab/data.html.

[34] X. Yuan, C. Li and X. Li, "Deep defense: Identifying DDoS attack via deep learning," in *Proc. IEEE Int. Conf. on Smart Computing (SMARTCOMP)*, Hong Kong, China, pp. 1–8, 2017.

[35] Intrusion detection evaluation dataset (ISCXIDS2012), [Accessed: 23-Dec.-2021]. *Available:* ttps://www.unb.ca/cic/datasets/ids.html.

[36] S. Suresh and N. S. Ram, "A review on various DPM traceback schemes to detect DDoS attacks," *Indian Journal of Science and Technology*, vol. 9, no. 47, pp. 1–8, 2016.

[37] Y. Bhavani, V. Janaki and R. Sridevi, "Survey on packet marking algorithms for IP traceback," *Oriental Journal of Computer Science & Technology*, vol. 10, no. 2, pp. 507–512, 2017.

[38] S. Wang, L. L. Minku and X. Yao, "A systematic study of online class imbalance learning with concept drift," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 10, pp. 4802–4821, 2018.

[39] Mininet emulator software, [Accessed: 11-Jan.-2022]. *Available:* mininet.org.

[40] M. M. Ahsan, M. A. P. Mahmud, P. K. Saha, K. D. Gupta and Z. Siddique, "Effect of data scaling methods on machine learning algorithms and model performance," *Technologies*, vol. 9, no. 52, pp. 1–17, 2021.

[41] D. Zhao, X. Wang, Y. Mu and L. Wang, "Experimental study and comparison of imbalance ensemble classifiers with dynamic selection strategy," *Entropy*, vol. 23, no. 7, pp. 1–22, 2021.

[42] L. Liu, P. Wang, J. Lin and L. Liu, "Intrusion detection of imbalanced network traffic based on machine learning and deep learning," *IEEE Access*, vol. 9, pp. 7550–7563, 2021.

[43] N. Ahuja, G. Singal, D. Mukhopadhyay and N. Kumar, "Automated DDOS attack detection in software defined networking," *Journal of Network and Computer Applications*, vol. 187, no. 6, pp. 1–42, 2021.