Microprocessors and Microsystems 89 (2022) 104412

Contents lists available at ScienceDirect

Microprocessors and Microsystems

journal homepage: www.elsevier.com/locate/micpro

An optimized weighted voting based ensemble model for DDoS attack detection and mitigation in SDN environment

Aastha Maheshwari *, Burhan Mehraj, Mohd Shaad Khan, Mohd Shaheem Idrisi

Department of Computer Science, Delhi Technological University, Delhi, India

ARTICLE INFO

Keywords:
Software defined networking
Distributed denial of service (DDoS)
Network security
Ensemble machine learning
Metaheuristic optimization

ABSTRACT

In recent years, software defined networking (SDN) has risen to prominence as a cutting-edge and promising networking approach. SDN is more secure and immune to DDoS attacks than traditional networks due to the bifurcation of the control and data planes, a global perspective of the whole network, centralized network control, and many other features. SDN still seems to be vulnerable to new DDoS attacks that can target the Application plane, control plane, or data plane, as well as the SDN architectural communication channels. In this paper, we propose a novel optimized weighted voting ensemble model to detect DDoS attack in an SDN environment. The proposed ensemble employs six base classifiers (two SVMs, two Random forests, and two Gradient Boosted Machines) that are differentiated by hyperparameter values. The optimal set of weights are identified by a novel hybrid metaheuristic optimization algorithm (BHO). Our approach eliminates false negatives by employing a novel dynamic fitness function. For training the ensemble framework we used CIC-DDoS2019 dataset. To examine the proposed model's performance, a testbed is developed using mininet and POX controller and loaded with multiple datasets. Our model outperforms all the recent approaches, achieving a low variance and high classification accuracies of 99.4163% and 99.3591% on CIC-DDoS2019 and CAIDA-2007 datasets respectively.

1. Introduction

The popularity of Web-based services and software has increased dramatically over the last two decades. Currently, the Internet is used by about 57 percent of the world's population [1]. As a result, there is a huge increase in Internet security concerns. Different security risks have often existed on the Internet. Worms, port scans, denial of service attacks, and Trojans, among other things, are all popular Internet outliers [2]. For such vast and complex networks, traditional network architectures and infrastructure do not have a flexible and obvious fix.

The three planes that make up network's work ability are the data plane, management plane and control plane. The vertical unification and ordination of the data and control planes into network devices using certain algorithms allow them to route, manage and alter network traffic in traditional IP-based networks. As a result, the network's entire structure becomes highly decentralized [3]. Devices are integrated with control logic in traditional networks. SDN is a dynamic programmable network replacing the fixed and complex traditional network. It is considered as one of the most significant networking breakthroughs [4].

SDN can be simply defined as the physical segregation of the control plane and the data forwarding plane of the network, and multiple devices in the network being controlled by control plane. SDN as stated by Kreutz et al. [5]:

1. Creating a separate unit called SDN controller by disengaging control functionality from network devices, and through this bifurcation of data and control plane network devices becomes fundamental forwarding components.
2. Flow-based, rather than destination-based, forwarding decisions are made using the packet's field values as a match criterion and collection of actions.
3. The network is programmable via APIs that communicate with underlying network devices in the data plane and run-on top of the SDN controller. APIs that run-on top of the SDN controller and interacts with fundamental network devices in the data plane.

The separation of data plane and control plane is at the heart of SDN architecture, which opens up a lot of possibilities for network design creativity [6]. SDN networks gain more programmability as a result

* Corresponding author.
E-mail addresses: aasthamaheshawari.am@gmail.com (A. Maheshwari), burhanmrj@gmail.com (B. Mehraj), shaadk7865@gmail.com (M.S. Khan), mohdshaheem91@gmail.com (M.S. Idrisi).

https://doi.org/10.1016/j.micpro.2021.104412
Received 2 August 2021; Received in revised form 4 October 2021; Accepted 10 December 2021
Available online 1 January 2022

of this bifurcation. As a result, network structure of SDN devices has become more manageable and organizable . In traditional networks, the control logic is embedded in the machines. If the policy has to be changed, vendor-specific attributes could be used to alter the logic on each system however that is a laborious task. Conversely SDN separates the control logic and puts it under unified control making it easy for system admins to modify control logic through APIs (southbound). Traditional switches are difficult to enforce new disruptive policies because they have minimal hardware potential and are vendor-specific. However, due to the unified controller in SDN, policy updation becomes very quick and convenient. In a conventional network, testing new policies is difficult due to limited area, but in an SDN network, there are far more possibilities for research.

A DoS attack is defined as a concerted effort to obstruct the use of specific network services by benign users.

To launch a Distributed Denial of Service (DDoS) attack, assailant start recruiting several devices which are geographically distributed. These infected computers are also referred to as bots and the collection of these bots is called a botnet [5]. The entire botnet is managed remotely by the bot-master [7–9] user. The attacker sends instructions to all the affected devices to initiate a DDoS attack, which would then be followed by unnecessary traffic on the target.

In the conventional network, the scale, frequency, intensity and complexity of DDoS attacks are increasing. This implies that the contemporary IDS for DDoS attacks are insufficient to tackle the obstacle. Attackers also strained techniques to circumvent current guarding shields [10,11]. However, SDN has features that have some benefits in defeating the DDoS attacks [11–15].

SDN can mitigate DDoS attacks by bifurcating the control plane and the data plane. But, because of its architecture, SDN itself is prone to DDoS attacks. SDN is divided vertically into three functional layers, as previously mentioned. DDoS attacks can be targeted to any of these layers. Layer wise vulnerabilities are summarized below:

- Data plane: In the data plane each switch has a limited sized flow table which makes it vulnerable to DDoS attack as an attack can cause flow table overflow [16]. Another potential vulnerability is buffer saturation [17]. Buffer memory is used in new flow rule generation process and its limited size makes data-plane vulnerable to DDoS attacks.
- Control-Data plane link: Sending numerous dummy packets to the switches, which transfers them to the controller through the control-data plane link. This flooding of packets through the channel can cause congestion in the channel.
- Control plane: Ultimately, as the attacker directs a flood of dummy packets to switches, all the requests are transferred by the switches to the controller, which causes the controller to become overburdened with fake requests, known as controller saturation.

In this paper, we propose an optimized weighted voting ensemble method to detect and mitigate DDoS attack in SDN environment that uses two SVMs, two random forests, and two Gradient boosted machines base classifiers to classify data. Two base learners of each type are distinguished in terms of the relevant hyperparameter values. The major contributions of our paper are:

1. A novel optimized weighted voting ensemble model.
2. A novel dynamic fitness function is proposed to reduce the false negatives while finding the optimal set of weights for the proposed ensemble.
3. BHO - A novel hybrid metaheuristic optimization algorithm is proposed which has improved exploration and exploitation capabilities and can utilize parallel processing to reduce the time complexity.

The remainder of this paper is arranged accordingly. A summary of the related works is given in Section 2. The main concepts used in our model are described in Section 3. The proposed model has been explained in Section 4. Experimental results are discussed in Section 5 using various performance metrics. Finally conclusion and future work is stated in Section 6.

2. Related work

DDoS identification and prevention have been a top priority for researchers in recent years. Many of the other researchers suggested DDoS attack prevention mechanisms, as well as strategies for minimizing the impact of DDoS attacks. This section provides a comprehensive discussion of current research in DDoS attack detection and prevention in the sense of SDN.

2.1. Information theory based approach

Giotis et al. [18] used a sFlow protocol to detect DDoS attack. This protocol limits the communication between the OpenFlow switch and controller thereby lowering the overload of controllers during the large network traffic conditions. These flow rules are also used by the mitigation module to block malicious flow i.e., DDoS attacks. However, the use of flow sampling using the sFlow protocol also affected the accuracy of detection.

Tsai et al. [19] employed a DDoS detection and prevention system based on entropy in which, an application on the controller monitors incoming traffic and computes the Entropy which is compared against a fixed entropy threshold. If it detects an intrusion, it will assist in the implementation of a prevention technique to block the relevant port, and this programme will then transfer this information to Network Intrusion Detection System (NIDS) for further inspection, and NIDS will be able to avoid this malicious traffic. One major limitation is choosing an appropriate entropy threshold value as it highly affects the detection rate.

Kalkan et al. [20] proposed a novel model JESS which is an acronym for Joint Entropy-based Security Scheme having three states, nominal, preparatory, and active mitigation stage. It is an entropy-dependent scoring system for detection and mitigation of DDoS attacks. It is based on the target IP address entropy, and dynamic combinations of TCP and IP layer attributes. This system resolves the problem of capable switch, which is a contentious problem in the literature, since it contradicts the fundamental properties of SDN such as distributed forwarding and centralized control. When the controller detects a threat, it notifies the switch. Then the attack packets are discharged by the Attack Mitigation module while protecting lawful packets.

Bawany et al. [21] introduced an flexible architecture known as SEcure and AgiLe (SEAL) that includes 3 different modules for solving the problem of DDoS attack detection and prevention : a-defence, c-defence, and d-defence. By using a customized category of Estimated weighted moving average filters, SEAL attains adaptability. Commonly used filter includes active, constructive and passive filters. For result analysis they generated traffic in controlled environment, which is the major shortcoming of proposed work.

Cui et al. [22] suggested using cognitive-inspired computation with dual address entropy to detect DDoS attacks. SVM learning was used for classification. Information of the switch from the flow table are obtained. DDoS attacks are observed where the entropy of the target is smaller and the entropy of the origin is exceeding the usual threshold of the traffic. To mitigate a DDoS threat, the DDoS attack defence dumps all table records. The proposed technique identifies attacks quickly and has a good detection rate and low FPR

Nada et al. [23] proposed a hybrid method which assigns a node trust value after detecting if any nodes are involved in DDoS attacks. Offending nodes are banned for certain period of time. But the flexibility of algorithm gives them an opportunity to reintegrate. But, if malevolent conduct is repeated, the nodes are forbid forever. The proposed approach is more robust as it takes into consideration the

dynamics of current network traffic and can be adapted in real time as the traffic complexity of a network varies. The major drawback is the choice of threshold value, which is selected experimentally in this work. The threshold value may be calculated using machine learning.

A common downside of these entropy-based techniques is the need to select the optimal detection threshold and poor accuracy at lower attack rates

2.2. Machine learning based approach

Li et al. [24] suggested a two-tier intrusion detection system (IDS) that identifies network anomalies by capturing global network flows. Two algorithms were used for extraction of features, i.e. swarm bat algorithm and Binary differential. Random forest algorithm is then used in order to characterize the movement of traffic to determine the system's possible interference by adapting the weights of samples using a voting ensemble. The downside is that this approach is not implemented and tested in real-world situations.

Phan et al. [25] proposed hybrid machine learning model to defend against a DDoS attack. They used modified History based IP Filtering known as enhanced HIPF (eHIPF) thereby empowering the model to detect and classify the network quickly. During the creation of eHIPF attack, to drop each individual packet at the border of the cloud, a flow mod message is sent with a command of drop down action. The proposed approach is inefficient to mitigate probe/reconnaissance attacks because the traffic volume generation of these attacks is very low

Myint et al. [26] used a tool for detecting DDoS attacks with minimal overhead using an advanced support vector machine. Volumetric and asymmetric characteristics are used to limit preparation time. Two forms of flooding: UDP flooding and ICMP flooding have been successfully identified. No information on the mitigation approach was provided by the authors.

Polat et al. [27] compared different ML algorithms based on feature selection. Authors used SVM, KNN, Naïve Bayes and ANN. In order to observe the accuracy they used multiple ML methods with varied feature selection. Since the emphasis was also on the best feature selection, a certain approach was used to pick many features.

Jia et al. [28] suggested a detection approach that would combine many multi-classifier de-compositions with a Single Value Decomposition (SVD). They believe that building different classifications is more accurate than static classifications. It would be insightful to measure the efficiency of the process when applied to a big dataset. The model produces downgrading results as compared to the K-NN algorithm.

Ahuja et al. [29] identified novel features for DDoS attack detection and logged them onto a CSV file to create a dataset specifically for DDoS attack. In the training phase, they used a hybrid consisting of a support vector classifier combined with an additional filtration by random forest (SVC-RF) and achieved a high accuracy while maintaining a low false alarm rate. The major drawback of the proposed hybrid is that it emphasizes more on the decision of random forest as compared to that of Support vector classifier.

2.3. Artificial neural network based approach

Cui et al. [30] developed a software-defined networking SD-Anti-DDoS defence mechanism. The proposed scheme includes 4 modules in sequence. Trigger module for a DDoS attack detection raise the reaction rate for the event trigger. BPNN technology is used to detect a traffic anomaly after activation by the trigger module to detect the attack. According to the path of detection module, the migration module obstruct traffic to minimize the impact of recent module. The other remaining modules are attack detection and traceback. The findings show that SD-Anti-DDoS can detect and identify the source of the attack easily within a second. However, the tool TFN2k they used to construct the traffic is outdated in current times.

Li et al. [31] used deep learning to solve the problem of DDoS attack detection in SDN environment. This software features a functionality analyser that analyses the unprocessed data units and generates detailed training information. The Flow Table Generator calculate flood entries and priorities of numerous Attack sets and sends them on the basis of statistics provided by the statistical module to the OpenFlow switch. Authors used a DDoS defence architecture tested by real-time attachments and used the ISCX dataset for training purposes. It was using hardware to design the experimental set-up. The major drawback of the proposed model was the problem of processing overhead, which can slow down the network significantly.

Novaes et al. [32] suggested a composite system for the detection and mitigation of port scanning and DDoS attacks in SDN network. This scheme is used in three stages to track attacks, including characterization, deviation and prevention. Authors use entropy metric for quantifying the network attribute and use LSTM (Last Short-Term Memory) for modelling the signature of each standard traffic attribute. The anomalies on the network were then detected using furious logic. They validate their approach by implementing floodlight control via Mininet simulation.

Gharvirian et al. [33] used Fast entropy rather than entropy for detecting attacks in SDN particularly DDoS, and therefore the detection time for the calculation is minimized and the threshold value is modified according to traffic conditions. Another good approach for improving the accuracy of the attack detection used in these experiments is the use of the neural perceptron network for the analysis of flux statistics. The neural network improves the accuracy of the detection and false warning rate in the course of this study and demonstrates the nature of an attack in examining the 3 travel functions of the network. Given the potency of the detection period, the proposed model used the neural network just for suspicious flows. Also, they have not mentioned the way to mitigate the attack.

Liu et al. [34] proposed a method for DDoS detection that combines the entropy method with the Particle Swarm Optimization based Back Propagation neural network (PSO-BP). The method uses the generic entropy method distributed on the switch to pre-diagnose the traffic. After identifying the switch at which the abnormal activity was first addressed, if the calculated entropy is greater than a certain threshold, the model uses PSO-BP neural network to extract the flow features of that switch which would be utilized for detection of DDoS attack. The major drawback of this approach is that the model might misclassify random entropy bursts in realistic networks as attacks. The change in network schema results into highly varied threshold value making this approach network scenario dependent.

Hannache et al. [35] introduced a Back Propagation Neural-network based Traffic Flow Classifier for real-time DDoS detection in SDN. The author trained the model using a custom-made dataset containing features extracted from both incoming packet messages received on SDN controller, and Openflow (OF) messages reciprocating between SDN controller and Virtual Open Flow Switches (VOFS). The model is capable of exploring the OF traffic by utilizing the fact that SDN allows deep packet analysis thereby controlling the comprehensive behaviour of the virtual environment. Along with detection, the proposed model also possesses the capability of mitigating the DDoS attacks. This paper lacked the extensive experimentation on multiple standard datasets which makes it susceptible to high variance problem.

2.4. Miscellaneous approach

Wang et al. [36] proposed DaMask which is a cloud-based DDoS detection approach. DaMask has is capable of detecting as well as mitigating the DDoS attack, for that it has two modules: DaMask-D and DaMask-M which are the detection module and mitigation module respectively. If the attack is detected by the DaMask-D module, along with the generation of alert, the information regarding the malicious packet is sent to DaMask-M to mitigate the attack. The author put

more emphasis on tackling DDoS detection in SDN for cloud-based environments rather than any arbitrary environment.

In order to shield SDN network from flow table overloading DDoS attacks, Bhushan et al. [37] proposed a novel flow table sharing based approach. The authors suggested using other space in the flow table for relatively little overhead contact. As a consequence, the rise in network resistance to flow table overflows DDoS attacks. The efficiency of the process in the attack is influenced to establish the holding time effectively. The efficacy of the procedure depends, however, on the time required.

Aleroud et al. [38] proposed a novel graph theory based approach. They used a catalogue of attack signatures from conventional network sets of data. This technique uses regularly labelled flows in order to analyse a new OpenFlow sample and to see how the existing signature is used for flow analysis. The drawback however is that the detection is based only on the signatures of the attack

Wang et al. [39] used a several path rerouting strategy to minimize the impact of LFA (Link Flooding Attacks), which at the time contains many target links. The method selects multiple optional routes and decides how much traffic to the optional connections should be redirected. The attacker has to locate a new target connection after rerouting. This will find the LFA bots and the malicious traffic from the LFA bots is then eliminated.

We learned from the literature review that ensemble learning employs weak base learners (with low accuracy) and that majority voting may result to high accuracy and low false alarm rate. Since, in an ensemble, base learners cannot perform equally well. Therefore, different weights should be assigned to the base learners in order to improve classification performance. To date, many different approaches have been proposed to find optimal weights of base learners in an ensemble. R. Burduk. [40] proposed a fuzzy set based approach in which interval valued fuzzy set is utilized to represent weights. There also exists several uses of probabilistic framework to find the optimal weights of base learners [41–43].

To find optimal weights, in addition to these analytical techniques, metaheuristic techniques can also be used. Zhang et al. [44] proposed differential evaluation based approach to find optimal weights and build an ensemble model with strong generalization ability. Onan et al. [45] defined a multi-objective function based on the classification accuracy of each base learners and used differential evaluation to optimize it to solve the problem of text sentiment classification and achieved a high classification accuracy. But, due to no data preprocessing their model underperformed.

3. Major concepts

3.1. DDoS attack on SDN

Software Defined Networking (SDN) network architecture provides many benefits thereby making it is resilient to DDoS attacks. However, SDN has some security issues, specifically its controller is vulnerable against the attacks of the Distributed Denial of Service (DDoS). When DDoS attacks occur against the SDN controller, the mechanism and connectivity capability of the controller are overwhelmed. Layer-by-layer overview of potential DDoS attacks on SDN are shown in Table 1.

Table 1
Layer-by-layer overview of potential DDoS attacks.

S.No.	SDN layer	DDoS attacks	Description
1	Application	Un-Authorized Application	Malicious applications are given access by instances of other applications and change network behaviour and are caused by network performance degradation.
2	Data plane	Flow table overflow	To store the flow tables, the switches have TCAM memory. The high cost and strength of memory makes this memory minimal.
		Spoofing of switch	IP address of the switch is spoofed and used to send control messages with changed address. Second switch is used to establish a connection with the controller [9].
		Buffer saturation	Packet header field must be buffered by switches. OpenFlow switches has restricted memory to buffer the data.
3	Control plane	Packet-in flooding	Overloading of the controller due to switch stimulated by spoofed IP's sending bulk packet_in messages [8].
		Saturation of controller	The central controller shall draw up a new flow rule for each switch in the network. The controller can be overloaded with several malicious requests [7,12].
4	Communicat--ions link	Congestion of south bound API	Switch requests new flow rules with Packet in messages from the controller for each new traffic flow and receive the responses with Packet out messages that trigger high traffic in the connection [6].

3.2. Support vector machine

One of the most effective binary classification techniques is the Support Vector Machine (SVM) [46]. SVM is widely used for its high generalization ability and strong resistance towards overfitting. SVMs tend to be resistant to overfitting because, rather than minimizing the squared error or an absolute value of an error they minimize structural risk.

Formulation of structural risk minimization (SRM) problem can be done in terms of data, given a dataset the objective $L(\theta)$ can be written as:

$$L(\theta) = \frac{1}{2n}\sum_{i=1}^{n}(h_\theta(x^i) - y^i)^2 + \frac{\lambda}{2}\sum_{j=1}^{d}\theta_j^2 \quad (1)$$

The first term in the Eq. (1) is mean squared error (MSE) term between learned model h_θ, and given labels y. The second term is the regularization term, which penalize larger weights and supports sparsity. λ is trade-off coefficient, a hyperparameter which affect the emphasis of regularization term.

By establishing a hyperplane or a collection of hyperplanes SVM classifies data into different classes. In SVM a hyperplane is described as a surface whereby the data points are completely sectioned in two classes to prevent the fall of data points of one class to the other, as shown in Fig. 1. The hyperplane with the greatest distance between the nearest points in the two groups is considered to be the best out of an excessive extent of hyperplanes [46]. A hyperplane is the collection of points $\vec{x}$ that satisfies the equation:

$$\vec{w}.\vec{x} + b = 0 \quad (2)$$

in Eq. (2) b and w are the bias and the weight vector respectively. The Lagrangian multipliers in the linear case can be used to solve these. The original maximum-margin hyperplane was introduced for linearly separable cases. But, SVM can also be useful in nonlinear classification problems, it does so by utilizing kernel trick with maximum margin hyperplanes. This enables the algorithm to transform feature space to a higher dimension and fit the maximum-margin hyperplane [47], as shown in Fig. 1. Commonly used kernel functions are listed in Table 2.

3.3. Random forest

Decision trees [48] are a powerful and widely utilized method for different machine learning tasks. However, they are rarely accurate due

Table 2
Commonly used kernel function.

Function	Equation
Linear kernel	$f(a_n, a_i) = (a_n, a_i)$
Polynomial kernel	$f(a_n, a_i) = (\gamma(a_n, a_i) + r)^d$
Sigmoid Kernel	$f(a_n, a_i) = \tanh(\gamma(a_n, a_i) + r)$
RBF kernel	$f(a_n, a_i) = exp(-\gamma \|\|a_n, a_i\|\|^2 + C)$

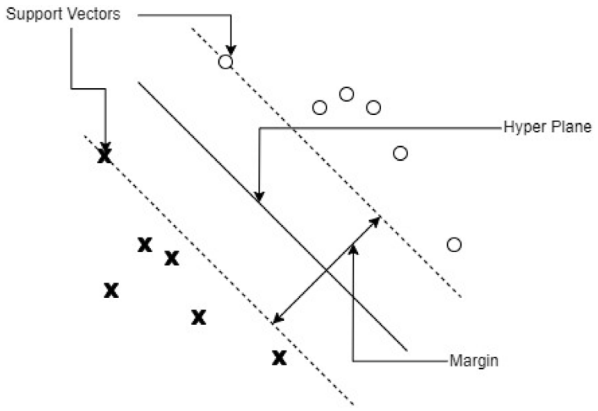


Fig. 1. Support vector machine.

to their low bias [49] yet high variance. Random forests [50] are a method of integrating a number of deep-seated decision-making trees trained in different sectors on the same training set with the aim of diminishing variance and enhancing model performance.

Brieman [50] integrates Bagging [51] with node-by-node random variable selection [52], given a training dataset $(x_i, y_i)_{i=1}^n$, repeatedly bagging (M times) chooses random subset samples of training set (using random replacement strategy) and fit trees to the chosen subset:

For m = 1, . . . , M:

1. Sample subset, with random replacement of n training data points from X, Y; say X_m, Y_m.
2. Using modified tree learning algorithm train a classification tree f_m on X_m, Y_m.

The modified tree learning algorithm picks a random sub-set of attributes at each division of the feasible candidate. "Feature bagging" is a term used to describe this procedure and it is shown in the algorithm 1.

Algorithm 1: Find spl^* (the best spilt) that divides f_t, among a random subset of D≤p input variables.

Function `FindBestSplitRandom`(f_t, D):
  $\Delta = -\infty$
  Draw D random indicies j_d from 1,...,p
  for $d = 1, \cdots, d$ **do**
    Find the best binary split $spl^*_{j_d}$ defined in X_{j_d}
    if $\Delta i(spl^*_{j_d}) > \Delta$ **then**
      $\Delta = \Delta i(spl^*_{j_d}, t)$
      $spl^* = spl^*_{j_d}$
    end
  end
  return spl^*
End

3.4. Gradient boosted machine

Gradient boosting [53–55] is a learning technique that combines the results of a large number of simple predictors to create an efficient committee with better performance than single members. Gradient boosting creates additive classification models by iteratively fitting a basic parameterized function (base learner) to current pseudo-residuals using least squares. As the iteration proceeds further, pseudo-residuals are being minimized as they are the gradient of loss function.

This machine learning algorithm is well-known for producing state-of-the-art results on a variety of complex datasets. GBM can be used for both regression and classification task.

Algorithm 2: Gradient boosted machine

Input : $(x_i, y_i)_{i=1}^n, L(y_i, F(x)$
1 **Begin:**
2 $F_o(x) = \underset{\gamma}{argmin} \sum_{i=1}^n L(y_i, \gamma)$
3 **for** $m = 1, ..., M$ **do**
4 **for** *i=1 to n* **do**
5 $r_{im} = -[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}]_{F(x)=F_{m-1}(x)}$
6 **end**
7 **for** $j=1,...,J_m$ **do**
8 Fit a regression tree to the r_{im} values and create terminal regions R_{jm}
9 **end**
10 **for** $j=1,...,J_m$ **do**
11 $\gamma_{jm} = \underset{\gamma}{argmin} \sum_{x_i \in R_{ij}} L(y_i, F_{m-1}(x_i) + \gamma)$
12 **end**
13 $F_m(x) = F_{m-1}(x) + \nu \sum_{j=1}^{J_m} \gamma_m I(x \in R_{jm})$
14 **end**
15 **End**
Output: $F_M(x)$

As inputs, the algorithm 2 takes a Dataset and a differentiable loss function. In step 2, the model is initialized with a constant value $F_0(x)$. M base learners are sequentially trained from lines 3 to 14. After calculating pseudo-residuals in steps 4–6, a base learner is trained to learn these pseudo residuals and create terminal regions in steps 7–9. In steps 10 to 12, the optimum value of γ_{jm} is calculated. In Step 13, using additive expansion technique Fm(x) is updated.

The procedure's learning rate is regulated by the "shrinkage" parameter ν, where $0 < \nu \leq 1$. Empirically, small values ($\nu \leq 0.1$) have been found to lead to a much better generalization error. I($x \in R_{jm}$) counts the number of data-points falling under the terminal region R_{jm}.

4. Proposed approach

This section describes our approach for detecting DDoS attacks in an SDN environment. The proposed approach consists of a learning phase followed by a detection and mitigation phase.

4.1. Learning phase

The architecture of the proposed optimized weighted voting ensemble model is given in Fig. 2. For training the model we used DDoS Evaluation Dataset (CIC-DDoS2019) [56] and for testing we used both CIC-DDoS2019 and CAIDA-2007 [57]. It is among the latest datasets for DDos Evaluation and is available in both CSV and PCAP format which makes it ideal for training the machine learning model as well as injecting it to SDN simulation (Mininet). The dataset consists of 12 different DDoS attacks using SYN PortScan, WebDDoS, SNMP, MSSQL, DNS, LDAP, SSDP, UDP, UDP-Lag, NetBIOS, NTP, and TFTP at different time. The availability of 80 features gives a huge amount of versatility in the precise selection of features and fine-tune it for our study.

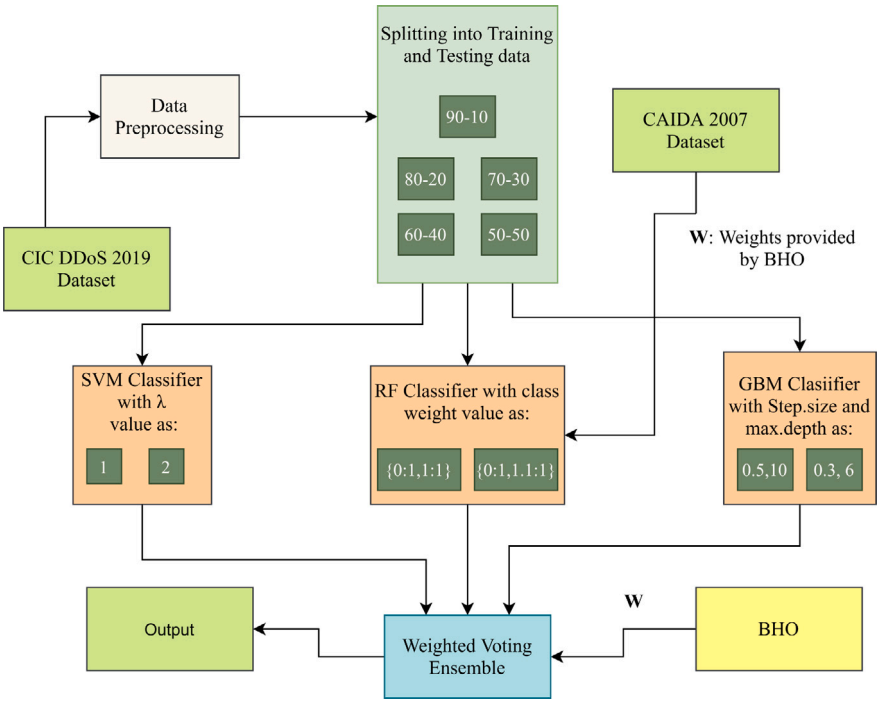


Fig. 2. Architecture of the proposed model.

In the first step, data is preprocessed to make it fit for training the ensemble. The dataset is highly unbalanced with more than 90% of packets related being DDoS which significantly increases the number of false positives. So, we balanced the dataset by undersampling it. We intended to balance the dataset without losing the versatile attack nature of the dataset. So, we subsampled the packets from each type of attack and used all the benign packets available in the dataset.

We chose a total of 20 features shown in Table 3. We wanted the model to be more generalized so we avoided the testbed related features like IP addresses and ports of both victim and attacker. From the definition, in a DDoS attack, the perpetrator tries to make the resources unavailable by flooding them with requests which make features related to the duration of each flow, packets per second and bytes per second most distinctive among all. During a DDoS attack, the machine becomes unresponsive which makes the Download versus Upload Ratio High for the host and makes the attack predictable. After analysing all such factors we procured the 20 most distinctive yet not machine-specific features for our study as shown in Table 3. After data preprocessing step dataset is divided into training and testing datasets.

Table 3
List of features used for training and testing the model.

Feature	Explanation
Flow duration	Duration of the flow (microsecs)
Tot Fwd Pkts	No. of total forward pkts
Fwd Pkt Len Max	Maximum size of forward pkts
Fwd Header Len	Total length for forward headers in bytes
Bwd Pkt Len Mean	Mean size of backward pkts
Tot Bwd Pkts	No. of total backward pkts
having Flow IAT Mean	Inter-Arrival Time of two pkts sent in the flow
Fwd IAT Tot	Total Inter-Arrival Time of two forward packets
Bwd IAT Tot	Total Inter-Arrival Time of two forward packets
Bwd PSH Flags	No. of times the PSH flag was set in backward packets
Fwd Pkts/s	pkts per second in forward direction
Bwd Pkts/s	pkts per second in backward direction
Pkt Len Std	Standard deviation length of a pkt in bytes
RST Flag Count	No. of pkts having RESET Flag
CWE Flag Count	No. of pkts having CWE Flag
Down/Up Ratio	Download and upload ratio of the flow
Fwd Byts/b Avg	Avg bytes bulk rate (ABBR) towards forward direction
Fwd Seg Size Min	Avg size of segment (ASoS) towards forward direction
Active Mean	Mean of the time a flow was active
PSH Flag Count	No. of pkts having PUSH Flag

In the second step, data is trained on each base learner. Since the performance of base learner depends on the choice of hyperparameter value. Therefore, to ensure that the base learners are used to their full potential, for each base learner we trained two different types with different hyperparameter values. We used linear SVM which uses linear kernel function for classification. Two linear SVM base learners are trained with hyperparameter λ (trade-off coefficient) value as 1 and 2. Similarly, two random forest (RF) base learners are trained with hyperparameter class_weight value as {0: 1, 1: 1} and {1: 1.2, 0: 1}. And the two gradient boosted machines (GBM) base learners are trained with hyperparameter step size value as 0.5 and 0.3, and max_depth as 10 and 6 respectively.

In the third step, the outcomes of SVM, Random forest and GBM base learners are ensembled using weighted majority voting (WMV). Littlestone and Warmuth [58] showed that providing weights to the base learners reduces the errors made by the ensemble system.

In this study, we use Optimized weighted voting ensemble which provide weights to all the base learners based on their classification accuracy. To find the optimal weights we propose a novel dynamic fitness function and BHO, which is a novel metaheuristic optimization algorithm. BHO is a hybrid of Brain Storm optimization algorithm (BSO) [59] and Henry's gas solubility optimization algorithm (HGSO) [60].

4.1.1. Proposed novel dynamic fitness function

Meta-heuristic optimization algorithms use fitness functions to guide simulations towards an optimal solution. The two main categories of fitness functions are static and dynamic fitness functions. The output of a static fitness function does not change with the same input over the course of iteration. A dynamic fitness function, on the other hand, may have different outputs for the same input as iteration changes.

Static fitness function for Optimized weighted voting ensemble:

Since the weights assigned to each base learner depends on their accuracy. Therefore, a fitness function can be defined as:

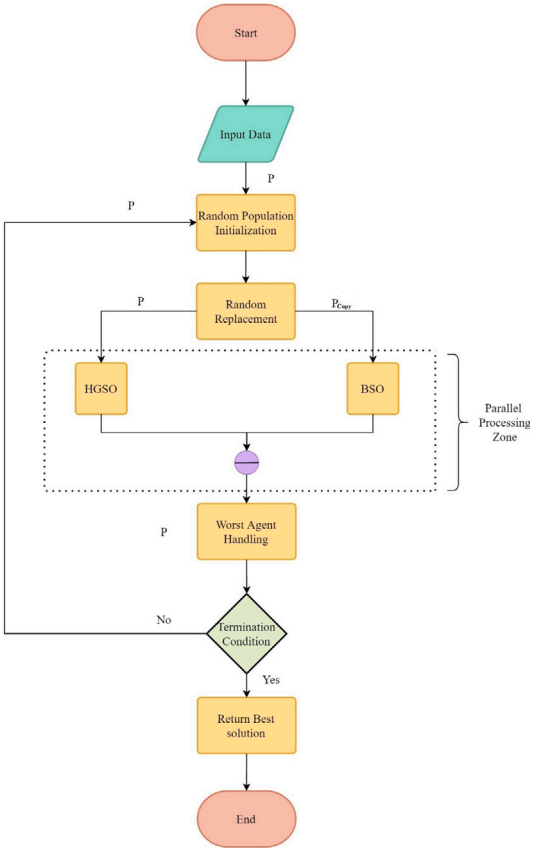


Fig. 3. Flowchart of the proposed model.

F (W, base learners, test dataset) =

$$1 - \text{Accuracy} = \frac{FN + FP}{TP + TN + FN + FP}$$

where, TN is true negative, TP is true positive, FP is false positive and FN is false negative.

Proposed dynamic fitness function for Optimized weighted voting ensemble:

Experimentally it was found that false negatives were more as compare to false positives, as show in experiment Section 5.2.3. Therefore, to reduce the number of false negatives we designed a dynamic fitness function as:

F (W, base learners, test dataset, σ) = $\frac{\alpha FN + \beta FP}{TP + TN + \alpha FN + \beta FP}$

The aim of this dynamic fitness function is to assign penalty weights α and β to FN and FP predictions respectively. Initially α will be more than β to reduce rate of FN, but as iteration proceeds both penalties approach each other to become equal to 1. σ is a hyper parameter which sets the maximum gap between α and β.

If T is the maximum number of iteration and t is the current iteration, then value of α and β are calculated as:

$\alpha = 1 + \gamma[1 - \sin(\frac{\pi t}{2T})]$; $\beta = 1 - \gamma[1 - \sin(\frac{\pi t}{2T})]$

$\gamma = \frac{\sigma(1-1/N)}{2}$; where γ takes into account the population size N and adjust σ accordingly.

4.1.2. BHO - novel hybrid meta-heuristic optimization algorithm

Metaheuristic algorithms are stochastic, population based computational intelligence paradigms that are specifically designed to solve complex optimization problems. Fig. 3 depicts flow chart of the proposed BHO algorithm.

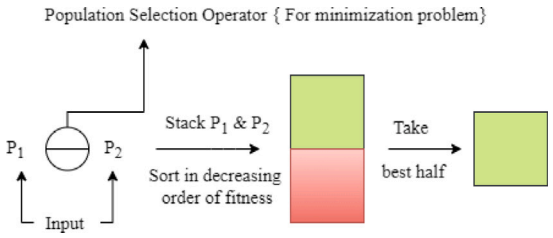


Fig. 4. Population selection operator.

BHO starts by taking in population size (N), number of groups (M), problem dimension (D), lower and upper bounds of solutions (LB and UB), maximum iterations (T) and other algorithm specific parameters as its input. Then it randomly initializes the population members in the limit of LB and UB and divides the population into M groups. Third step of BHO is Random replacement, where a group is chosen at random and best member of that group is replaced by a randomly generated solution using a greedy selection approach. This facilitates the exploration of new solutions.

Now in the fourth step of BHO, two copies of population are created (say P_1 and P_2), P_1 will undergo Henry's gas solubility optimization (HGSO) [60] while P_2 will undergo Brain storm optimization (BSO) [59]. In HGSO algorithm, α is a hyper parameter and it affects influence of other gases on ith gas of jth cluster [60] and in original implementation its value is taken as 1. We propose a constant called as amplification factor (AF) and it is used to increase α slowly in each iteration, which ultimately improves the convergence rate. α is initialized with a value of 1 and in each iteration, alpha is updated as $\alpha = \alpha * AF$, where $AF = 1 + \frac{0.1}{N}$

Fourth step can be parallelly processed to reduce time complexity. Now Population selection operator $\ominus$ is used to select N members out of P1 and P2. This step helps to improve the convergence rate of algorithm (see Fig. 4).

In the fifth step of BHO, population will be passed through worst agent handling module in which N_w worst solutions are selected and replaced with randomly generated solutions using greedy selection strategy. This step helps algorithm to escape local optima trap. Finally, after iterating for T times, best solution found so far is returned as an output of the algorithm.

4.2. Detection and mitigation phase

Firstly, results from each base learner are obtained. Then, using the weights obtained during the training phase, a weighted majority vote of the results obtained from each base learner is used as a final prediction to classify a new flow as either malicious or normal.

Final prediction = $majority[W_1.SVM_1, W_2.SVM_2, W_3.RF_1, W_4.RF_2, W_5.GBM_1, W_6.GBM_2]$

After deploying our model on POX controller, it monitors the traffic and as soon as the DDoS attack is detected, it blocks the victim port and prevent resources from further misuse.

5. Results and discussion

This section focuses on the experimental setup and performance evaluation of the proposed model for various dataset distributions. The proposed model is compared against base learners using metrics based on the confusion matrix. In the latter half of this section, the performance of the proposed dynamic fitness function is compared to that of static fitness function. Lastly, the proposed BHO optimizer is compared with HGSO and BSO optimizers.

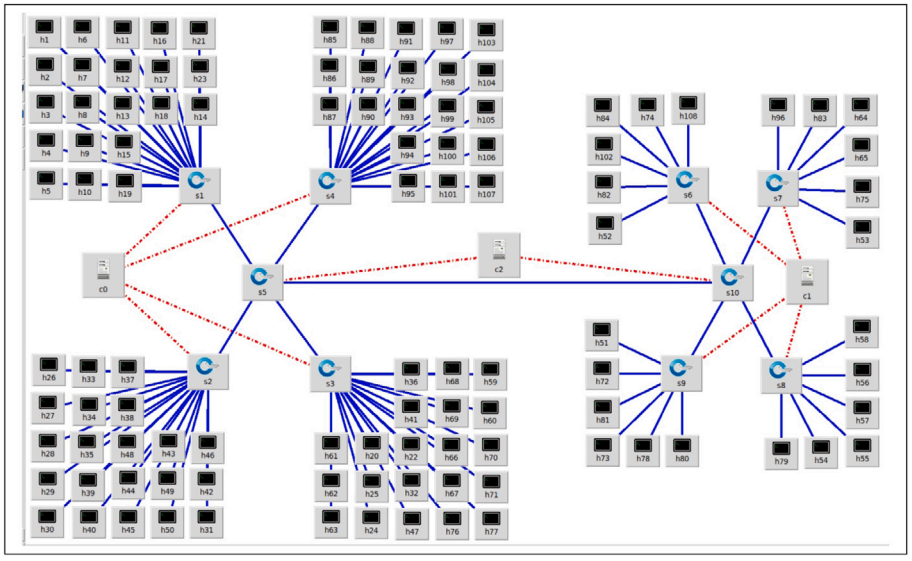


Fig. 5. Testbed.

5.1. Experimental setup

The experiments in this work are carried on a testbed created using mininet emulator and POX controller. Mininet is a network simulator that constructs a virtual network of hosts, switches, controllers, and links, while POX is an open source OpenFlow/Software Defined Networking (SDN) controller written in Python.

Fig. 5 shows our implemented testbed. It contains 108 hosts (h1-h108), 10 switches (s1 to s10), 3 controllers (c0-c2). 8 subnets are arranged in our testbed.

5.2. Performance evaluation

After creating the network setup, packets from CIC-DDoS2019 dataset were fed to the network. Fig. 6 shows the feature-wise-analysis of traffic on our SDN simulation.

To evaluate the performance of classifiers we have used the following metrics:

- Classification accuracy (Ac) = $\frac{TN+TP}{TP+FP+FN+TN}$
- Sensitivity/Recall (R) = $\frac{TP}{TP+FN}$
- Specificity (S) = $\frac{TN}{TN+FP}$
- Precision (P) = $\frac{TP}{TP+FP}$
- F-measure (F)= $2(\frac{PR}{P+R})$
- Detection rate (DR) = $100P$
- False alarm rate (FAR) = $100(\frac{FP}{FP+TN})$

where, TN is true negative, TP is true positive, FP is false positive and FN is false negative.

5.2.1. Comparison with base learners and majority voting ensemble (MVE)

Table 4 shows the comparison of base learners, MVE (majority voting ensemble) and the proposed OWVE (optimized weighted voting ensemble) in terms of above discussed performance metrics. Our proposed model is tested on CIC-DDoS2019 and CAIDA-2007 datasets and achieved an accuracy of 99.41% and 99.36% respectively.

SVM base learners performed poorly with a high false alarm rate (FAR). Random forest (RF) and Gradient boosted machines (GBM) performed much better than SVM base learners but still suffers from the problem of high FAR. MVE combines the strength of all base learners and outperformed each individual base learners in terms of classification accuracy as well as FAR. The proposed-OWVE model outperformed MVE and achieved very low FAR and high accuracy while maintaining a proper balance between Sensitivity and Specificity. The high performance of our proposed ensemble model on multiple datasets indicates the low variance of the framework.

Table 4
Base learners comparison.

Classifier	Ac (%)	R (%)	S (%)	F (%)	DR	FAR
SVM1	92.9609	96.3364	88.8576	93.7569	91.312	11.1424
SVM2	92.9648	96.3329	88.8706	93.7600	91.3209	11.1294
RF1	98.6802	99.7470	97.3834	98.8085	97.8876	2.61664
RF2	98.9279	99.1055	98.7117	99.0243	98.9433	1.28834
GBM1	98.9891	99.2766	98.6397	99.0806	98.8854	1.36031
GBM2	99.0751	99.3264	98.7697	99.1586	98.9913	1.23034
MVE	99.1201	99.3549	98.8346	99.1994	99.0443	1.16536
OWVE (CAIDA)	**99.3591**	**99.392**	**99.3221**	**99.4249**	**99.4595**	**0.6778**
OWVE (CIC-DDoS)	**99.4163**	**99.347**	**99.5022**	**99.4719**	**99.5971**	**0.4978**

Table 5
Performance of the proposed model.

Dataset distribution (CIC-DDoS)	Accuracy (%)
50–50	99.3896
60–40	99.3858
70–30	99.3432
80–20	99.4163
90–10	99.3155

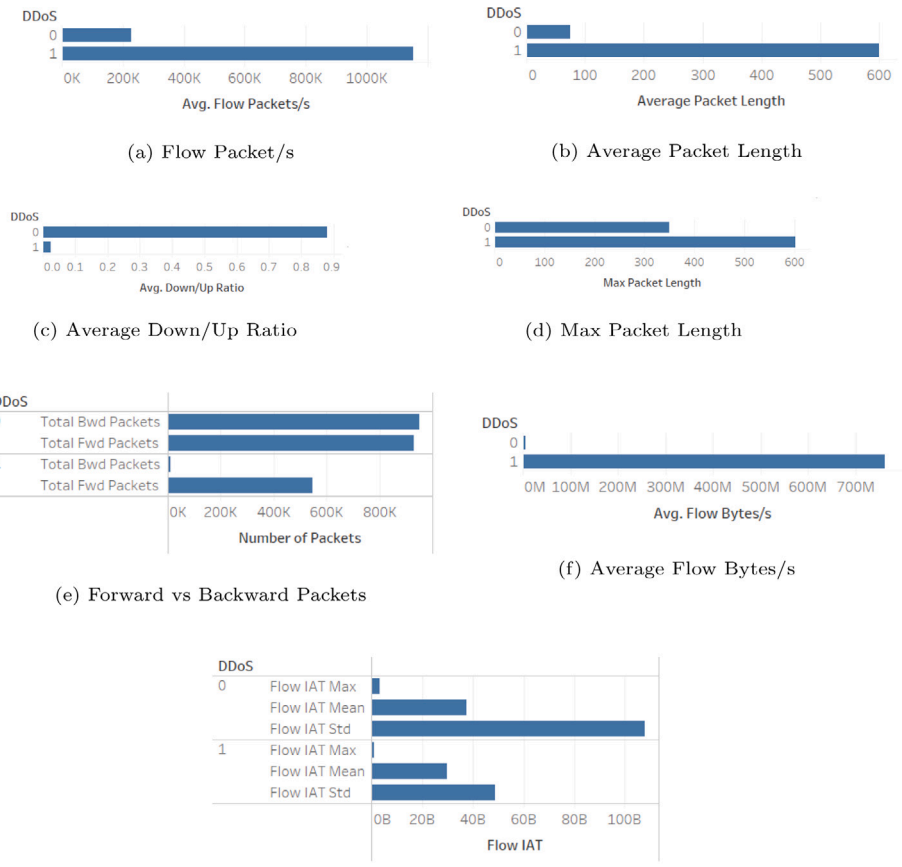
5.2.2. Dataset distribution

When it comes to assessing the effectiveness of a machine learning model, the distribution of dataset between training and testing is crucial. Table 5 shows how the performance of the proposed model varies with dataset distribution.

5.2.3. Comparison of proposed dynamic fitness function with static fitness function

The Figs. 7a, 7b shows the confusion matrix of the proposed model with static fitness function and dynamic fitness function for both CIC-DDoS2019, CAIDA-2007 datasets respectively.

The proposed design of the dynamic fitness function helps the model to increase the overall accuracy by decreasing the false negatives.



(a) Flow Packet/s

(b) Average Packet Length

(c) Average Down/Up Ratio

(d) Max Packet Length

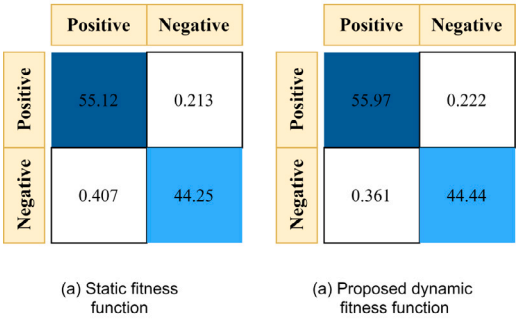
(e) Forward vs Backward Packets

(f) Average Flow Bytes/s

(g) Packet Flow Inter Arrival Time

Fig. 6. Feature-wise analysis of traffic.

(a) Confusion Matrix using CIC-DDoS2019



(b) Confusion Matrix using CAIDA-2007

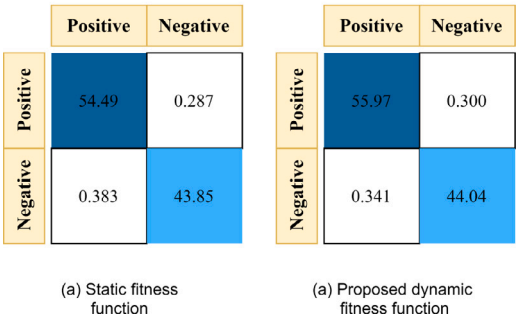


Fig. 7. Obtained confusion matrix (%).

Table 6
Comparison with previous studies.

Study	Year	Type	Method	Detection	Mitigation	Accuracy (%)
Myintoo et. al.	2019	ML	ASVM	Yes	No	97
Liu et. al.	2019	DL	Entropy + PSO-BP	Yes	No	97.7
Huseyin et. al.	2020	ML	Wrapped feature selection + KNN	Yes	No	98.3
Hannache et. al.	2020	DL	ANN + Traffic flow classifier	Yes	Yes	96.1
Makuvaza et. al.	2021	DL	DNN	Yes	No	97.6
Ahuja et. al.	2021	ML	SVM + RF	Yes	No	98.8
Ramprasath et. al.	2021	ML	Multinomial regression	Yes	Yes	98.7
Our proposed model	**2021**	**ML**	**OWVE**	**Yes**	**Yes**	**99.41**

With the static fitness function, the model achieved an accuracy of 99.384% and with dynamic fitness function the accuracy of the model is 99.416%.

5.2.4. Comparison with previous studies

The accuracy of the results obtained from our proposed model are compared to the accuracy of some recently proposed machine learning and deep learning based models (from 2019 to 2021) as shown in Table 6. A few of the models have been effective in enhancing levels of accuracy. In the field of machine learning based models, Ramprasath et al. [61] proposed a multinomial regression based approach to know the traffic alterations responsible for anomalies and achieved a notable

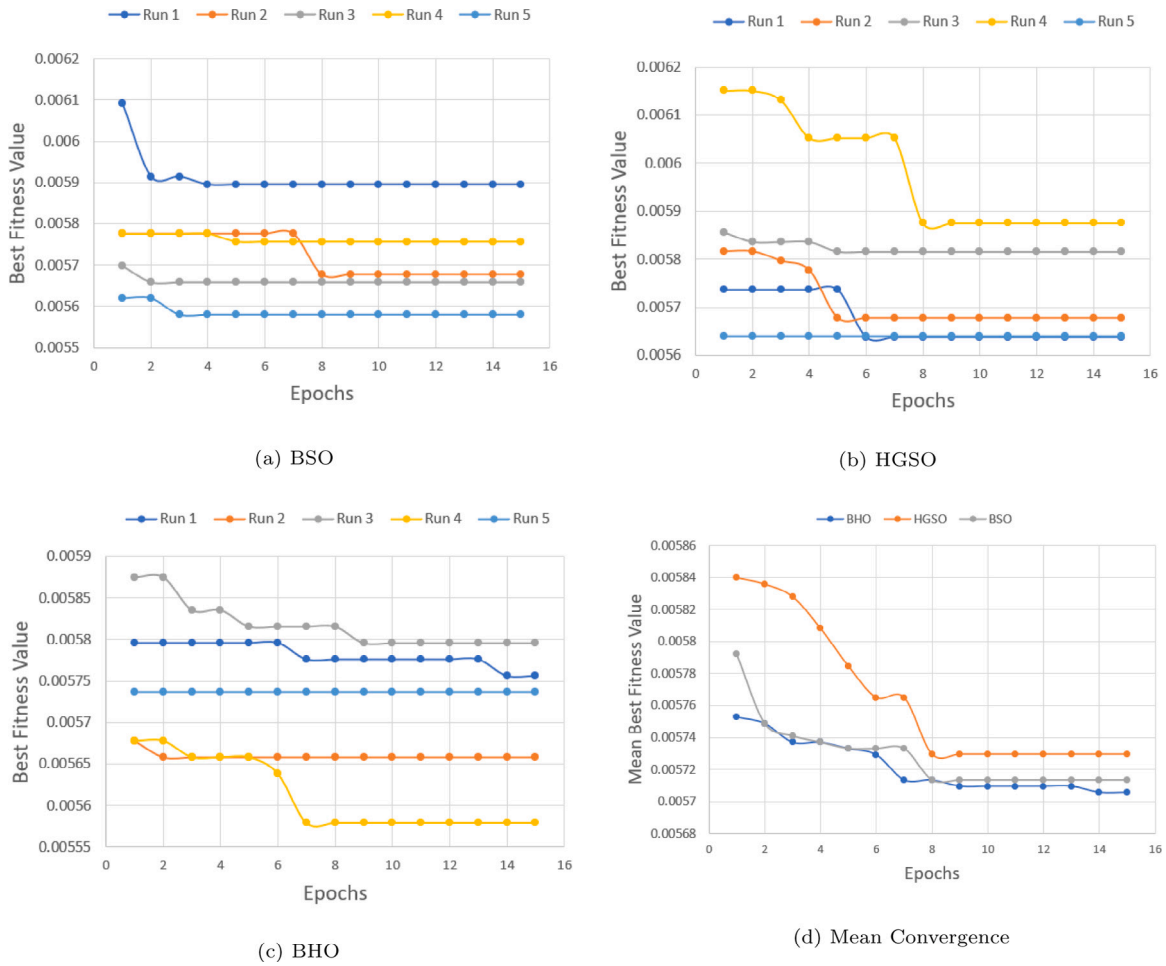


Fig. 8. Performance of the algorithms over 5 runs.

accuracy of 98.7% and Ahuja et al. [62] proposed a hybrid approach which utilizes support vector classifier followed by a random forest filtering and attained an accuracy of 98.8%. Coequally deep learning based models such as Hannache et al. [35] proposed an artificial neural network based traffic flow classifier and achieved an accuracy of 96.1% with mitigation capabilities.

Our proposed model, with an accuracy of 99.41% and 99.36% on CIC-DDoS, CAIDA datasets respectively, outperformed other shown recent models and has capabilities to detect and mitigate DDoS attacks on SDN with low variance.

5.2.5. Comparison of proposed BHO with HGSO and BSO

To compare the proposed hybrid metaheuristic optimizer (BHO) with the individual optimizers HGSO [60] and BSO [59], best fitness function value v/s iteration curve is drawn for 5 runs. Using the best fitness value attained in each run, a statistical table is generated which shows the performance of the algorithm over 5 runs (Table 7). Fig. 8: a, b and c shows the performance of BSO, HGSO and BHO respectively.

With the minimum best fitness value over 5 runs while maintaining a minimum mean, median, and standard deviation, BHO outperformed both BSO and HGSO. In the worst case scenario, BHO does not suffer too much loss and outperformed both BSO and HGSO.

The graph in Fig. 8d is a mean convergence curve (over 5 runs) which, at an iteration, takes mean of best fitness values achieved in 5 runs. As can be seen from the graph, in terms of best mean fitness value and convergence rate, BHO outperformed both HGSO and BSO.

Table 7
Statistical analysis to compare BHO with BSO and HGSO (derived using best fitness value over 5 runs).

Statistical quantity	Algorithm		
	BSO	HGSO	BHO
Best	0.005579323	0.005638468	0.005579323
Worst	0.006091911	0.006151056	0.005875047
Mean	0.005728370	0.005764140	0.005721530
Median	0.005697613	0.005737000	0.005737042
Standard deviation	0.000114556	0.000137538	7.9806E−05

6. Conclusion and future work

In this paper, we proposed a new approach to the detection and prevention of DDoS attacks in the environment of software defined network. The proposed optimized weighted voting ensemble model includes 6 base learners (2 SVMs, 2 RFs, 2 GBMs) differentiated by hyperparameters. To find an optimal set of weights for the proposed ensemble, a novel dynamic fitness function is proposed along with a novel hybrid metaheuristic optimization algorithm (BHO). The design of proposed dynamic fitness function helped the model to reduce false negatives and increasing overall accuracy. For training the ensemble framework we used CIC-DDoS2019 dataset. The model's efficiency is assessed on two datasets, CIC-DDoS2019 and CAIDA-2007 which are fed to a testbed created using mininet and POX controller. The proposed model achieves an accuracy of 99.4163% and 99.3591% respectively. High accuracy on multiple datasets shows low variance of the proposed ensemble framework.

In the future, we will explore novel ways of designing dynamic fitness functions for minimization problem and novel ways to create efficient hybrids that can combine strengths of existing metaheuristic optimization algorithms and use them in different domains to create efficient systems.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] Internet growth usage statistics, 2019, https://www.clickz.com/internetgrowth-usage-stats-2019-time-online-devices-users/235102/.

[2] J. Singh, S. Behal, Detection and mitigation of DDoS attacks in SDN: A comprehensive review, research challenges and future directions, Comp. Sci. Rev. 37 (2020) 100279.

[3] T. Benson, A. Akella, D.A. Maltz, Unraveling the complexity of network management, in: NSDI, 2009, pp. 335–348.

[4] H. Uppal, D. Brandon, OpenFlow based load balancing, in: CSE561: Networking Project Report, University of Washington, 2010.

[5] D. Kreutz, F.M. Ramos, P.E. Verissimo, C.E. Rothenberg, S. Azodolmolky, S. Uhlig, Software-defined networking: A comprehensive survey, Proc. IEEE 103 (1) (2014) 14–76.

[6] S. Scott-Hayward, G. O'Callaghan, S. Sezer, SDN security: A survey, in: 2013 IEEE SDN for Future Networks and Services (SDN4FNS), IEEE, 2013, pp. 1–7.

[7] M. Feily, A. Shahrestani, S. Ramadass, A survey of botnet and botnet detection, in: 2009 Third International Conference on Emerging Security Information, Systems and Technologies, IEEE, 2009, pp. 268–273.

[8] M. Abu Rajab, J. Zarfoss, F. Monrose, A. Terzis, A multifaceted approach to understanding the botnet phenomenon, in: Proceedings of the 6th ACM SIGCOMM Conference on Internet Measurement, 2006, pp. 41–52.

[9] B. Saha, A. Gairola, Botnet: An Overivew. CERT-In White Paper, Tech. rep., CIWP-2005-05, 2005, June.

[10] Crippling cyber-attacks, 1998, https://www.bbc.com/news/technology35376327.

[11] S. Jajodia, K. Kant, P. Samarati, A. Singhal, V. Swarup, C. Wang, Secure Cloud Computing, Springer, 2014.

[12] P. Zhang, H. Wang, C. Hu, C. Lin, On denial of service attacks in software defined networks, IEEE Netw. 30 (6) (2016) 28–33.

[13] W. Xia, Y. Wen, C.H. Foh, D. Niyato, H. Xie, A survey on software-defined networking, IEEE Commun. Surv. Tutor. 17 (1) (2014) 27–51.

[14] S. Bu, F.R. Yu, X.P. Liu, H. Tang, Structural results for combined continuous user authentication and intrusion detection in high security mobile ad-hoc networks, IEEE Trans. Wireless Commun. 10 (9) (2011) 3064–3073.

[15] S. Sezer, S. Scott-Hayward, P.K. Chouhan, B. Fraser, D. Lake, J. Finnegan, N. Viljoen, M. Miller, N. Rao, Are we ready for SDN? Implementation challenges for software-defined networks, IEEE Commun. Mag. 51 (7) (2013) 36–43.

[16] M. Goossens, F. Mittelbach, A. Samarin, Open Networking Specifications 1.5.1, Vol. 3, Open Networking Foundation, 2015.

[17] OpenFlow switch, 2020, https://www.opennetworking.org/wp-content/uploads/2014/10/openflow-switch-v1.5.1.pdf. (Accessed on 11 March 2020).

[18] K. Giotis, C. Argyropoulos, G. Androulidakis, D. Kalogeras, V. Maglaris, Combining OpenFlow and sFlow for an effective and scalable anomaly detection and mitigation mechanism on SDN environments, Comput. Netw. 62 (2014) 122–136.

[19] S.-C. Tsai, I.-H. Liu, C.-T. Lu, C.-H. Chang, J.-S. Li, Defending cloud computing environment against the challenge of DDoS attacks based on software defined network, in: Advances in Intelligent Information Hiding and Multimedia Signal Processing, Springer, 2017, pp. 285–292.

[20] K. Kalkan, L. Altay, G. Gür, F. Alagöz, JESS: Joint entropy-based DDoS defense scheme in SDN, IEEE J. Sel. Areas Commun. 36 (10) (2018) 2358–2372.

[21] N.Z. Bawany, J.A. Shamsi, SEAL: SDN based secure and agile framework for protecting smart city applications from DDoS attacks, J. Netw. Comput. Appl. 145 (2019) 102381.

[22] J. Cui, M. Wang, Y. Luo, H. Zhong, DDoS detection and defense mechanism based on cognitive-inspired computing in SDN, Future Gener. Comput. Syst. 97 (2019) 275–283.

[23] N.M. AbdelAzim, S.F. Fahmy, M.A. Sobh, A.M.B. Eldin, A hybrid entropy-based DoS attacks detection system for software defined networks (SDN): A proposed trust mechanism, Egypt. Inform. J. 22 (1) (2021) 85–90.

[24] J. Li, Z. Zhao, R. Li, H. Zhang, Ai-based two-stage intrusion detection for software defined iot networks, IEEE Internet Things J. 6 (2) (2018) 2093–2102.

[25] T.V. Phan, M. Park, Efficient distributed denial-of-service attack defense in SDN-based cloud, IEEE Access 7 (2019) 18701–18714.

[26] M. Myint Oo, S. Kamolphiwong, T. Kamolphiwong, S. Vasupongayya, Advanced support vector machine-(ASVM-) based detection for distributed denial of service (DDoS) attack on software defined networking (SDN), J. Comput. Netw. Commun. 2019 (2019).

[27] H. Polat, O. Polat, A. Cetin, Detecting DDoS attacks in software-defined networks through feature selection methods and machine learning models, Sustainability 12 (3) (2020) 1035.

[28] B. Jia, X. Huang, R. Liu, Y. Ma, A DDoS attack detection method based on hybrid heterogeneous multiclassifier ensemble learning, J. Electr. Comput. Eng. 2017 (2017).

[29] N. Ahuja, G. Singal, D. Mukhopadhyay, N. Kumar, Automated DDOS attack detection in software defined networking, J. Netw. Comput. Appl. (2021) 103108.

[30] Y. Cui, L. Yan, S. Li, H. Xing, W. Pan, J. Zhu, X. Zheng, SD-Anti-DDoS: Fast and efficient DDoS defense in software-defined networks, J. Netw. Comput. Appl. 68 (2016) 65–79.

[31] C. Li, Y. Wu, X. Yuan, Z. Sun, W. Wang, X. Li, L. Gong, Detection and defense of DDoS attack–based on deep learning in OpenFlow-based SDN, Int. J. Commun. Syst. 31 (5) (2018) e3497.

[32] M.P. Novaes, L.F. Carvalho, J. Lloret, M.L. Proença, Long short-term memory and fuzzy logic for anomaly detection and mitigation in software-defined network environment, IEEE Access 8 (2020) 83765–83781.

[33] F. Gharvirian, A. Bohlooli, Neural network based protection of software defined network controller against distributed denial of service attacks, Int. J. Eng. 30 (11) (2017) 1714–1722.

[34] Z. Liu, Y. He, W. Wang, B. Zhang, DDoS attack detection scheme based on entropy and PSO-BP neural network in SDN, China Commun. 16 (7) (2019) 144–155.

[35] O. Hannache, M.C. Batouche, Neural network-based approach for detection and mitigation of DDoS attacks in SDN environments, Int. J. Inf. Secur. Priv. (IJISP) 14 (3) (2020) 50–71.

[36] B. Wang, Y. Zheng, W. Lou, Y.T. Hou, DDoS attack protection in the era of cloud computing and software-defined networking, Comput. Netw. 81 (2015) 308–319.

[37] K. Bhushan, B.B. Gupta, Distributed denial of service (DDoS) attack mitigation in software defined network (SDN)-based cloud computing environment, J. Ambient Intell. Humaniz. Comput. 10 (5) (2019) 1985–1997.

[38] A. AlEroud, I. Alsmadi, Identifying cyber-attacks on software defined networks: An inference-based intrusion detection approach, J. Netw. Comput. Appl. 80 (2017) 152–164.

[39] J. Wang, R. Wen, J. Li, F. Yan, B. Zhao, F. Yu, Detecting and mitigating target link-flooding attacks using sdn, IEEE Trans. Dependable Secure Comput. 16 (6) (2018) 944–956.

[40] R. Burduk, Recognition task with feature selection and weighted majority voting based on interval-valued fuzzy sets, in: N.-T. Nguyen, K. Hoang, P. Jędrzejowicz (Eds.), Computational Collective Intelligence. Technologies and Applications, Springer Berlin Heidelberg, Berlin, Heidelberg, 2012, pp. 204–209.

[41] A. Rojarath, W. Songpan, Cost-sensitive probability for weighted voting in an ensemble model for multi-class classification problems, Appl. Intell. (2021) 1–25.

[42] J. Large, J. Lines, A. Bagnall, A probabilistic classifier ensemble weighting scheme based on cross-validated accuracy estimates, Data Min. Knowl. Discov. 33 (6) (2019) 1674–1709.

[43] L.I. Kuncheva, J.J. Rodríguez, A weighted voting framework for classifiers ensembles, Knowl. Inf. Syst. 38 (2) (2014) 259–275.

[44] Y. Zhang, H. Zhang, J. Cai, B. Yang, A weighted voting classifier based on differential evolution, in: Abstract and Applied Analysis, Vol. 2014, Hindawi, 2014.

[45] A. Onan, S. Korukoğlu, H. Bulut, A multiobjective weighted voting ensemble classifier based on differential evolution algorithm for text sentiment classification, Expert Syst. Appl. 62 (2016) 1–16.

[46] C. Cortes, V. Vapnik, Support-vector networks, Mach. Learn. 20 (3) (1995) 273–297, http://dx.doi.org/10.1007/BF00994018.

[47] B.E. Boser, I.M. Guyon, V. Vapnik, A Training Algorithm for Optimal Margin Classifiers, ACM, New York, 1992, pp. 144–152.

[48] J.R. Quinlan, Simplifying decision trees, Int. J. Man-Mach. Stud. 27 (3) (1987) 221–234.

[49] J. Friedman, T. Hastie, R. Tibshirani, et al., The Elements of Statistical Learning, Vol. 1, Springer series in statistics New York, 2001.

[50] L. Breiman, Random forests, Mach. Learn. 45 (1) (2001) 5–32, http://dx.doi.org/10.1023/A:1010933404324.

[51] L. Breiman, Bagging predictors, Mach. Learn. 24 (2) (1996) 123–140.

[52] Y. Amit, D. Geman, K. Wilder, Joint induction of shape features and tree classifiers, IEEE Trans. Pattern Anal. Mach. Intell. 19 (11) (1997) 1300–1305.

[53] J. Friedman, T. Hastie, R. Tibshirani, et al., Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors), Ann. Statist. 28 (2) (2000) 337–407.

[54] J.H. Friedman, Greedy function approximation: a gradient boosting machine, Ann. Statist. (2001) 1189–1232.

[55] J.H. Friedman, Stochastic gradient boosting, Comput. Statist. Data Anal. 38 (4) (2002) 367–378.

[56] I. Sharafaldin, A.H. Lashkari, S. Hakak, A.A. Ghorbani, Developing realistic distributed denial of service (DDoS) attack dataset and taxonomy, in: 2019 International Carnahan Conference on Security Technology (ICCST), IEEE, 2019, pp. 1–8.
[57] CAIDA dataset, 2007, https://www.caida.org/catalog/datasets/ddos-20070804_dataset.
[58] N. Littlestone, M. Warmuth, The weighted majority algorithm, in: 30th Annual Symposium on the Foundations of Computer Science, 1994.
[59] Y. Shi, Brain storm optimization algorithm, in: International Conference in Swarm Intelligence, Springer, 2011, pp. 303–309.
[60] F.A. Hashim, E.H. Houssein, M.S. Mabrouk, W. Al-Atabany, S. Mirjalili, Henry gas solubility optimization: A novel physics-based algorithm, Future Gener. Comput. Syst. 101 (2019) 646–667.
[61] A. Makuvaza, D.S. Jat, A.M. Gamundani, Deep neural network (DNN) solution for real-time detection of distributed denial of service (DDoS) attacks in software defined networks (SDNs), SN Comput. Sci. 2 (2) (2021) 1–10.
[62] J. Ramprasath, V. Seethalakshmi, Improved network monitoring using software-defined networking for DDoS detection and mitigation evaluation, Wirel. Pers. Commun. 116 (3) (2021) 2743–2757.

Aastha Maheshwari obtained her Master's degree from NIT Hamirpur and currently pursuing Ph.D. in Computer Science and Engineering Department from Delhi Technological University, Delhi. Her specializations Internet of Things, sensor network and wireless communication.
Email: aasthamaheshwari.am@gmail.com

Burhan Mehraj is currently pursuing a Bachelors' degree in Computer Science and Engineering from Delhi Technological University, Delhi. His interests are in Machine Learning, Deep learning, Metaheuristic optimization, data mining.
Email: burhanmrj@gmail.com

Mohd Shaad Khan is currently pursuing a Bachelors' degree in Computer Science and Engineering from Delhi Technological University, Delhi. His interests are in Machine Learning, Deep learning, Metaheuristic optimization, data mining.
Email: shaadk7865@gmail.com

Mohd Shaheem Idrisi is currently pursuing a Bachelors' degree in Computer Science and Engineering from Delhi Technological University, Delhi. His interests are in Machine Learning, Web development and App Development.
Email: mohdshaheem91@gmail.com