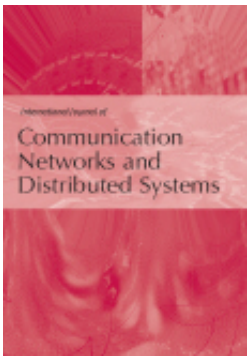


International Journal of Communication Networks and Distributed Systems

ISSN online: 1754-3924 - ISSN print: 1754-3916
https://www.inderscience.com/ijcnds

An efficient attack detection approach for software defined Internet of Things using Jaya optimisation based feature selection technique

Pinkey Chauhan, Mithilesh Atulkar

DOI: 10.1504/IJCNDS.2025.10062282

Article History:

Received:	28 May 2023
Last revised:	17 December 2023
Accepted:	18 December 2023
Published online:	02 December 2024

An efficient attack detection approach for software defined Internet of Things using Jaya optimisation based feature selection technique

Pinkey Chauhan* and Mithilesh Atulkar

Department of Computer Applications,
NIT Raipur,
Raipur, 492010, India
Email: pchauhan.phd2018.mca@nitrr.ac.in
Email: matulkar.mca@nitrr.ac.in
*Corresponding author

Abstract: Software defined Internet of Things (SD-IoT) has drawn several attacks because of its novelty. To counter such attacks, this paper presents a study on feature selection using Jaya Optimisation for making the lightweight intrusion detection system (IDS) for the data plane of SD-IoT. To check the effectiveness of the selected features, an ensemble of tree-based classifiers that uses a boosting approach called light gradient boosting machine (LGBM) is trained and tested with all features (AF) and then with selected features (SF) using 10-fold cross-validation. It was found that LGBM gave better performance when it was trained with SF. For performance evaluation, some well-known metrics have been used, namely recall, accuracy, false alarm rate (FAR), F1, precision, Cohen's Kappa coefficient (CKC), and prediction time. This trained model is deployed for attack detection in the OpenFlow-enabled devices of data plane of SD-IoT where it can detect the attacks in a distributed manner.

Keywords: OpenFlow, Ryu Controller; SD-IoT; software defined Internet of Things; Jaya Optimisation; machine learning; distributed denial of service attack.

Reference to this paper should be made as follows: Chauhan, P. and Atulkar, M. (2025) 'An efficient attack detection approach for software defined Internet of Things using Jaya optimisation based feature selection technique', *Int. J. Communication Networks and Distributed Systems*, Vol. 31, No. 1, pp.19–41.

Biographical notes: Pinkey Chauhan holds a Master's degree in Computer Applications from Madhav Institute of Technology & Science, Gwalior and Master of Philosophy from School of Studies in Computer Science & IT, Pt. Ravi Shankar Shukla University Raipur. Currently, she is pursuing PhD from the Department of Computer Applications, National Institute of Technology Raipur (Chhattisgarh), Raipur, C.G. Her area of interest includes computer networks, network security, computer programming, software defined networking, IoT and artificial intelligence.

Mithilesh Atulkar, Professor, Department of Computer Applications, National Institute of Technology Raipur (Chhattisgarh) India, has more than 30 years of teaching and research experience. His primary domain of teaching and research includes artificial intelligence, parallel processing, and software defined network. He has guided more than 70 MCA projects, seven PhD scholar. He is a Life

member of ACM, IETE, ISTE, CSI, ORSI, IE (India) and member of some other professional bodies. He has extensive publication record in international/national journals and conferences. He has authored four books and many book chapters. He is reviewer of many reputed journals. He has participated in various national and international seminars, workshops and conferences. He has received "Pratibha Samman" Award by ISTE Chapter Raipur for 2001. He has been involved in various administrative responsibilities since 1994. He has served as Controller of examination for six years at NIT Raipur (C.G.).

1 Introduction

Internet of Things (IoT) technology is regarded as the most fascinating in the era of information technology. In the modern world, which is interconnected by the Internet and other secure networks, hacking is a significant issue (Taylor et al., 2020; Singh et al., 2020; Kabacinski and Abdulsahib, 2020; Roesch, 1999; Wagner and Soto, 2002). Hackers continue to develop new techniques for breaking into networks to steal sensitive data about victims or to install harmful software to watch on their financial behaviour (Saeed et al., 2016; Fu et al., 2017; Chaabouni et al., 2019). A hacker's target is to find the weaknesses in the computer or computer networks and exploit those for making attacks. Hackers may be software engineers motivated by a variety of factors, including monetary gain, ill will towards the victim, testing their own hacking skills or network security, or simply for the sheer fun of it.

SDNs are gaining popularity in a wide range of fields like virtualisation, big data analytics, cloud computing, security, monitoring, etc.

SDN-based IoT (SD-IoT) refers to the use of SDN architecture to link IoT devices to the network (see Figure 1). The IoT devices are connected with the gateway which is further connected to the data plane devices of SDN. Whenever a new request comes to the switch from the IoT devices related to forwarding the packets, the switch searches its flow tables to check whether any entry related with this type of request exists or not. If any match is found, the packet is dealt according to the rules already available in the flow table. If no entry is matched, the switch sends the packet_in request to the controller. Upon receiving the request, the controller responds back with the packet_out response. This response consists of the rule which is installed in the flow table of the switch and the packet is dealt according to this installed rule. For making this communication, OpenFlow protocol is used whose important fields are shown in Figure 2.

Many attacks are possible in the SD-IoT network which may target any layer of the SD-IoT. One of such attacks is DDoS attacks which essentially overwhelm network resources with illegitimate and unwanted requests in order to make them unavailable to legitimate ones. By saturating networks with ICMP, TCP, or UDP traffic, a DDoS attack can be conducted. A DDoS attack can target any tier of the SDN, as stated in Liu et al. (2019), but most of the attacks are focused on the SDN controller. A compromised IoT device sends a flood of requests to the SDN network with some changes in the header of the data packets data.

To counter the attacks in the networks, two types of attack detection devices have been developed, namely anomaly based intrusion detection system (AIDS) and signature based intrusion detection system (SIDS) (Khraisat et al., 2019). In one hand where SIDS requires signature of the attack for detecting the attacks, AIDS require the training of the model with

the network traffic before they are used. The details about these systems is given in Khraisat et al. (2019) and Chauhan and Atulkar (2023a, 2023b).

Figure 1 SDN based IoT (see online version for colours)

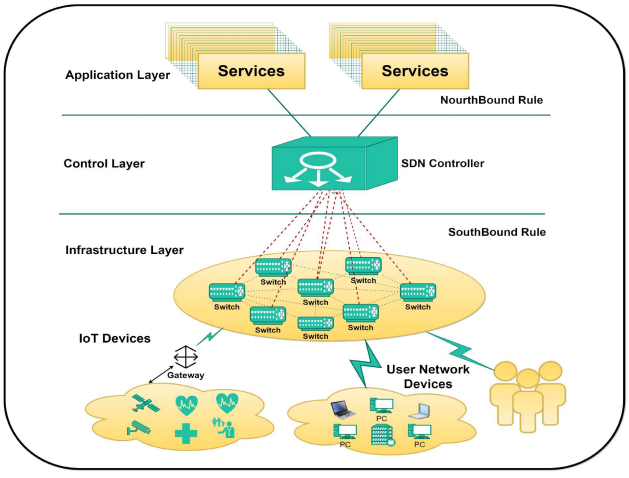
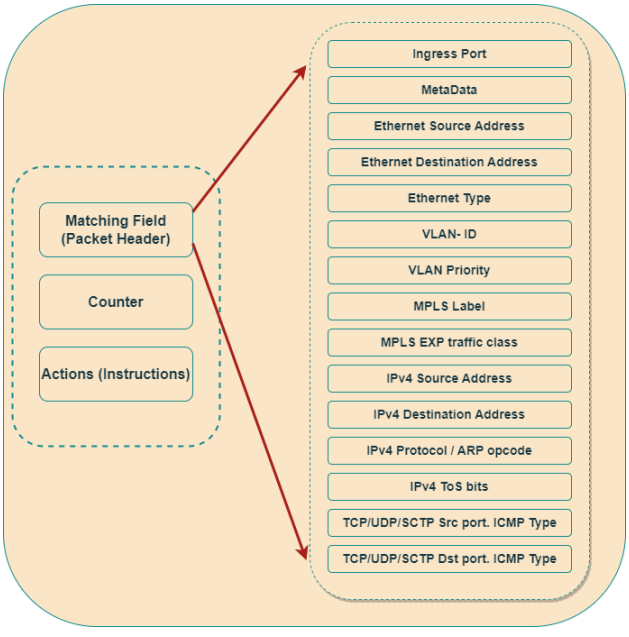


Figure 2 The field of Openflow protocol (see online version for colours)



The main contributions of the work are listed below:

1 feature selection using Jaya algorithm from the BoT-IoT dataset

2 with the objective of finding the best performing model, training and testing of the models

3 deployment of the trained model in the OpenFlow enabled devices of SD-IoT where they can detect the attacks for which it has been trained.

The rest of the paper is organised as follows. Section 2 discusses the work done in the field of feature selection and attack detection. The motivation behind doing the work is given in Section 3. The methodology how this has been done is shown in Section 4. Section 5 describes the implementation environment. Section 6 discusses the experiments. Conclusion and Future work has been described in Section 7.

2 Related work

In this section, various attack detection systems on data plane of SD-IoT have been studied. To recognise and thwart DDoS attacks on the SD-IoT, a semi-supervised deep Extreme Learning Machine was developed by Ravi and Shalinie (2020). They stated that they have implemented the attack detection system using a distributed controller environment but they have not cleared how they derived the features from the data and control plane. They used both their own internal dataset and publicly accessible datasets from UNB-ISCX to assess performance. By going through their literature, it is found that they employed the switched port analyser (SPAN) feature of the Cisco Nexus Switch to extract 155 features without providing any further details on how they obtained the features. These properties mostly apply to data packets, hence only the data plane can extract them. They have also exclusively used UDP packets to demonstrate their investigation. They achieved the value of F1 of 97.2%, the value of accuracy 97.9%, the value of precision 97.2%, the value of recall 97.6%, and the value of precision 97.2% with success.

Khanday et al. (2023) have developed a light weight attack detection system for IoT devices. They have used Logistic Regression, ANN, LSTM, and Linear SVC classifiers and used TON-IoT and BoT-IoT datasets for their result evaluation. The result of LSTM is the highest among other classifiers.

Saba et al. (2022) have developed CNN-based approach for anomaly detection. They have also used BoT-IoT for their result evaluation and achieved accuracy of 92.85%. The work done in Almaraz-Rivera et al. (2022) have performed binary and multiclass classification for attack detection in IoT networks. They have concated BoT-IoT dataset with their own dataset and checked the results. They got the values of recall, F1 ,accuracy, and precision, 98.908%, 98.98.605%, 98.908%, and 98.857% respectively.

A controller based centralised DDoS attack detection approach has been developed by Chauhan and Atulkar (2023b) for SD-IoT. They have implemented their work using Ryu controller in remote mode and they have created the SDN topology using Mininet emulator. For generating the attack and normal traffics, they used distributed internet traffic generator (DITG) and the hping3 tools. From these generated traffics, they have dumped some statistics and finally extracted six features from these statistics and created a dataset. Using this dataset they trained and tested some well known classifiers, namely LGBM, SVM, KNN, and RF to determine the effectiveness of the features extracted. LGBM's performance was found better than several other cutting-edge works. Its performance has been demonstrated under a wide variety of metrics and it achieved the value of accuracy 99.088%, value of precision 99.089%, value of recall 99.088%, value of F1 99.084%, value of prediction time 22.116 Seconds, value of false alarm rate (FAR) 0.329%, value of CKC 98.250%, and AUC value 0.996%.

In order to choose the best features, the work presented in Das et al. (2022) introduces a unique Jaya optimisation algorithm based feature selection approach. By updating the worst features to shrink the size of the feature space, this strategy employs a search technique to locate the best suited features. As a result, supervised machine learning algorithms perform better. Ten benchmark datasets are used to assess the performance of the suggested approach, which is then compared to other FS approaches such genetic algorithm-based FS, particle swarm optimisation algorithm-based FS, and differential evolutionary-based FS. The experimental findings indicate that FSJaya outperforms other approaches like FSGA, FSPSO, and FSDE in terms of average classification accuracy across the majority of datasets. The Friedman and Holm test has been used to validate the suggested approach's statistical significance proof. In comparison to its equivalents, the suggested strategy is proven to be effective at choosing an ideal subset of attributes.

The wrapper-based feature selection approaches are still controversial among academics because many of them are constantly working to create new feature selection methods by utilising various optimisation techniques. Several surveys on various feature selection approaches and their significance in the categorisation process have been made in Chandrashekar and Sahin (2014) and Venkatesh and Anuradha (2019). These surveys examined the difficulties as well as the various tactics used in feature selection methods in the past.

Chauhan and Atulkar (2023a) have proposed a stacking hybrid classifier-based system using Support Vector Machine and K-Nearest Neighbour at level 0 and LogisticRegression at level 1 to fight DDoS attacks. The NSL-KDD dataset for detecting DDoS attacks is used to test the performance of the hybrid classifier stack.

Trigger based attack detection system has been proposed in Tan et al. (2020). They are using a cooperative approach where the attack is first detected in data plane devices and then a trigger is sent to the controller which finally decides whether the attack detected by data plane device is a correct prediction or not. Once an attack is detected in the data plane device, the controller extracts the five features from the incoming traffic and checks for the attack. Once an attack is detected by the controller, steps to mitigate it are taken. For implementing their work, they have used Mininet and ONOS controller.

A control plane based approach for a centralised attack system has been developed by Ye et al. (2018) Floodlight controller has been used. They extracted 6 features from the synthetic dataset and checked the performance using SVM. (Tan et al., 2020) also developed a control plane based approach for attack detection. They used ONOS controller for attack detection, checked the performance on KNN and K-Means and blocked the attack in both the planes.

An approach to detect the attack in the controller of SDN has been developed by Kalkan et al. (2018). For their work, they have used Ryu controller, generated different type of attacks namely TCP SYN attack, DNS Amplification Attack, NTP Attack, Generic Attack and Mixed Attack to create their own dataset and finally trained models with the created dataset. For the same objective of attack detection, Chen et al. (2018) used POX controller and used Hyenae for performing attack. They found XGBoost performing best among other classifiers.

Niyaz et al. (2017) developed a control plane based approach for attack detection. They used stacked auto encoder (SAE) for feature reduction derived from network traffic headers. Karan et al. (2018) developed a data plane based approach for attack detection. They used DNN and SVM for feature selection on KDD CUP 99 dataset, and found that DNN is performing better than SVM.

The raw packet capture files of four datasets, namely BoT-IoT, UNSW-NB15, TON-IoT, and CSE-CIC-IDS2018 have been used by Sarhan et al. (2021) where they are converting these into corresponding NetFlow format. They labelled the created dataset for both binary and multiclass traffic. They evaluated the performance using an Extra Tree ensemble classifier. They achieved the accuracy of 93.82% and F1 value of 97%.

Banitalebi Dehkordi and Soltanaghaei (2020) have developed a control plane based approach to detect the attack in SDN. They used Floodlight controller on ISCX-SlowDDoS-2016, ISCX-IDS2012, CTU-10 and CTU-11 datasets. They used BaysOut, J48, Simple Logistic, Random Tree, Naive Bayes, RepTree classifiers. The issue here is that all the datasets are meant for dataplane and not for control plane.

For SD-IoT based 5G network, an intrusion detection and prevention system is proposed by Sarica and Angin (2020). Their solution is based on automatic feature extraction and classification of the attack using Random Forest classifier in the application layer of SDN. They have used BoT-IoT dataset and achieved the value of accuracy, precision, F1, recall as 96%, 96.71%, 94.69%, and 92.75% respectively. An approach for cyber defence using attack graphs prediction and visualisation has been proposed by Mishra (2023). To detect an attack and assess the performance of the suggested system, machine learning techniques such as SVM, RF, KNN, LR, and multilayer perceptron (MLP) are used. The top classifiers in terms of accuracy, recall, precession, and F-score were RF and MLP.

The prior key works in this area which are related to the current work have been mentioned in Table 1.

3 Motivation

The main objective of detecting the attack in the OpenFlow enabled devices of data plane of SD-IoT is to make the controller free for performing some other important tasks. There are some works like Chouhan et al. (2023) which works on detecting the attack in the controller of SDN. The problem with this solution is that the major parts of the controller resources are consumed in checking the attack in a frequent interval while the resources of data plane devices remain idle. If these resources are used for attack detection, it will make the controller free for performing some other tasks. The second problem with detecting the attack centrally is that in case of attack the channel connecting the data plane devices with controller becomes chocked like in DDoS attack making the controller irresponsive for legitimate requests. If this attacks had been detected in the data plane devices, channel would never chock.

The motivation of this work is the above mentioned two reasons.

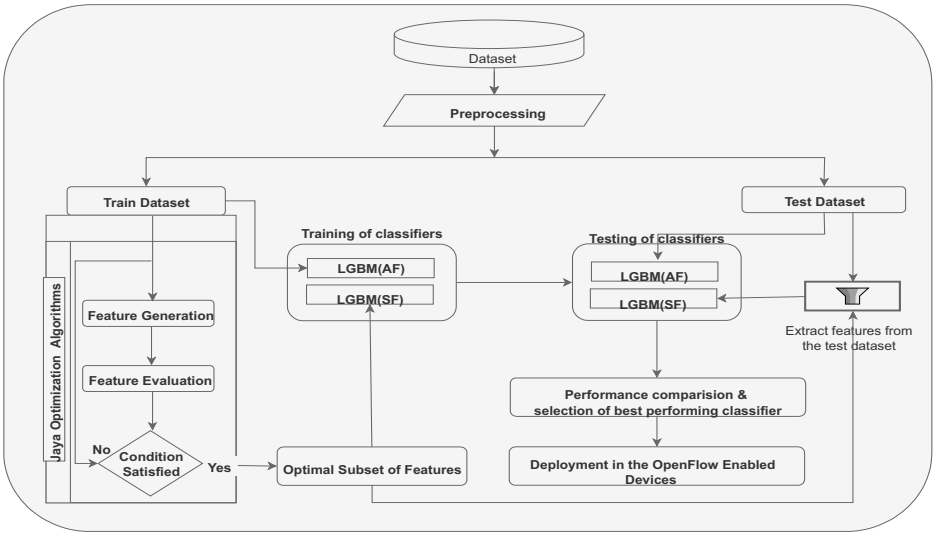
Table 1 Literature survey on key works

Ref. no.	*SDN layer*	*Description*
Haider et al. (2020)	Data plane	Performed work on CICIDS2017 dataset and found CNN ensemble outperforms other classifiers.
Sarhan et al. (2021)	Data plane	The raw packet capture files of four datasets, namely BoT-IoT, UNSW-NB15, ToNIoT, and CSE-CIC-IDS2018 have been used by the authors where they are converting these into corresponding NetFlow format. They labeled the created dataset for both binary and multiclass traffic. They evaluated the performance using an Extra Tree ensemble classifier.
Saba et al. (2022)	Data plane	Developed CNN-based approach for anomaly detection. They have also used BoT-IoT for their result evaluation and achieved accuracy of 92.85%.
Almaraz-Rivera et al. (2022)	Data plane	authors have performed binary and multi-class classification for attack detection in IoT networks. They have concated BoT-IoT dataset with their own dataset and checked the results.
Friha et al. (2022)	Data plane	They propose a system for securing agricultural-IoT infrastructures. They use three deep learning classifiers: deep neural networks, convolutional neural networks, and recurrent neural networks. They investigate the proposed IDS's performance using three different sources: CSE-CIC-IDS2018, MQTTset, and InSDN. The results show that the FELIDS system outperforms traditional/centralised machine learning systems.
Khanday et al. (2023)	Data plane	The authors have developed a light weight attack detection system for IoT devices. They have used Logistic Regression, ANN, LSTM, and Linear SVC classifiers and used TON-IoT and BoT-IoT datasets for their result evaluation. The result of LSTM is the highest among other classifiers
Chauhan and Atulkar (2023a)	Data plane	To fight DDoS attacks, a stacking hybrid classifier-based system using Support Vector Machine and K-Nearest Neighbour at level 0 and LogisticRegression at level 1 has been deployed. The NSL-KDD dataset for detecting DDoS attacks is used to test the performance of the hybrid classifier stack.
Polat et al. (2020)	Control plane	They used Pox controller and extracted 6 features from the synthetic dataset and checked the performance on SVM, NB, ANN, KNN.
(Chouhan et al., 2023)	Control plane	Created their own dataset by extracting 7 features from the attack and normal traffics, trained SVM, RF, KNN, XGBoost, and Naive Bayes and tested their performance where SVM found outperforming other classifiers.
Chauhan and Atulkar (2023b)	Control plane	They used their own dataset to implement a centralised IDS for IoT. Using this dataset they trained and tested some well known classifiers, namely LGBM, SVM, KNN, and RF. LGBM's performance was found better than several other cutting-edge works.
Wang et al. (2019)	Both data and control plane	Used Multiple Ryu controller and used safe guards in both; Data Plane and Control. BPNN has been used for performance measurement.
Ravi and Shalinie (2020)	Both data and control plane	Used Ryu controller and UNB-ISBX datasets, features are basically concerned with data packet of data plane. Used Deep ELM for classification
Yu et al. (2021)	Both data and control plane	A Ryu controller based cooperative approach between the data plane and control plane is adopted. They used Entropy and RF for classification
Ahuja et al. (2021)	Control plane	Used Ryu controller, extracted 8 features from 2 generated features from synthetic dataset. They used SVC-RF classifier

4 Methodology

Figure 3 shows the complete flow of the work done in this research. First of all, the dataset is preprocessed and then broken into two parts; train and test datasets. Train dataset is given to the Jaya algorithm to find the optimal subset of features. By using the process of removing noisy, irrelevant, and redundant features, it explores the solution space. Now, one LGBM classifier, named as $LGBM_{(SF)}$, is trained and tested using these selected features using 10-fold cross validation. Once this process is over, another LGBM classifier, named as $LGBM_{(AF)}$, is trained and tested using all the features and then the performance of both the classifiers is compared under the metrics recall, accuracy, FAR, F1, precision, CKC, and prediction time. Finally, the best performing model is deployed in all the OpenFlow enabled data plane devices.

Figure 3 Methodology of proposed work



4.1 Jaya Algorithm based feature selection

The population-based algorithms need some controlling parameters, including initial weight, population size, generational frequency, mutation probability, and crossover probability. To get at the best answer, these parameters must be correctly tweaked. The success of the optimisation method depends heavily on the right tuning of these algorithm-specific parameters; on the other hand, improper tuning of these parameters may trap in local optimum and also raise the computational cost of the problem.

In order to solve the aforementioned problems, the Jaya optimisation method is utilised (Venkata Rao, 2016). This is a specific parameter-less algorithm i.e., it takes only the compulsory parameters which are required for population based optimised algorithm, there is no algorithm specific parameter. This feature overcomes the above mentioned problem as no parameter tuning is required for algorithm specific parameters.

Let the total number of candidate solutions (population) is P, and the total number of features is D (design variable). Initially random candidate solution is picked and in it a

subset of features is taken let j where $j = 1, 2, 3, \ldots, D$. These selected features are sent to the classifier and the fitness value is calculated. To get the optimal subset of features the value of fitness function should be minimised. Each output given by the classifier is matched with actual output and the error is calculated using equation (1) where $\hat{y}(k)$ is the actual output and $y(k)$ is the predicted output. Equation (2) is used for calculating the fitness value where k might take values $1, 2, 3, \ldots, m$. Here m is the total number of samples in testing dataset. Thus, fitness value is calculated by dividing the total error with the number of samples available is testing dataset.

$$\text{Error}(o) = (\hat{y}(k) \neq y(k)) \tag{1}$$

$$\text{f}(i) = \frac{\sum_{i=1}^{m} \text{Error}(o)}{m} \tag{2}$$

Let $f(x)_{\text{best}}$ and $f(x)_{\text{worst}}$ represent the best and worst value of fitness function, X_{best} is the best value of the variable and X_{worst} is the worst value of the variable. Similarly, $X_{j,k}$ represents the value of the j^{th} variable for the k^{th} candidate and $X_{j,k}^{t}$ represents the $X_{j,k}$ value for t^{th} iteration. This algorithm works by updating the value of individual by using equation (3) where X_{new} is the new value, $X_{j,k}$ is the present value, $r1$ and $r2$ are two numbers whose values range between 0 and 1. The term $r_1\ (X_{\text{best}} - |X_{j,k}|)$ drives the individual towards best solution whereas the term $r_2\ (X_{worst} - |X_{j,k}|)$ drives to discard the worst solution. The equation that is used to update the value can be represented by equation (4) for an individual iteration t.

$$X_{new} = X_{j,k} + r_1\ (X_{\text{best}} - |X_{j,k}|) - r_2\ (X_{worst} - |X_{j,k}|) \tag{3}$$

$$X_{\text{new}}^{t} = X_{j,k}^{t} + r_1\left(X_{\text{best}}^{t} - \left|X_{j,k}^{t}\right|\right) - r_2\left(X_{\text{worst}}^{t} - \left|X_{j,k}^{t}\right|\right) \tag{4}$$

The binary conversion of X_{new} is done using equation (5) and new candidate is selected if its probability is given 1 (Das et al., 2022).

$$X_{new} = \begin{cases} 1 & if\ \ X_{j,k} > threshold \\ 0 & Otherwise \end{cases} \tag{5}$$

Now for X_{new} the fitness value is calculated as $f_{new} = f(X_{new})$.
For minimisation problem, following steps are performed

If $f_{\text{new}} < f_i$, then replace X_i with X_{new} and f_i with f_{new}.

For maximisation problem, following steps are performed

If $f_{\text{new}} > f_i$, then replace X_i with X_{new} and f_i with f_{new}.

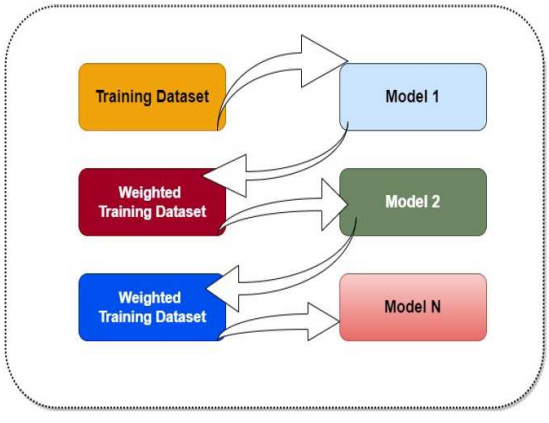
The current work's objective is to minimise the error so this comes under minimisation problem and hence has been dealt accordingly as mentioned above.

4.2 Light gradient boosting machine (LGBM)

LGBM is a tree based ensemble classifier that uses boosting approach in its working. LGBM uses gradient-based one side Sampling (GOSS) and exclusive feature bundling (EFB) which make it different from other boosting algorithms. GOSS keeps the samples which have large gradient and removes other with lower gradient. EFB is used in a dataset where sparse feature space exists. There might be many mutual exclusive features which take nonzero values simultaneously. In such cases, exclusive features are bundled into a single feature resulting to fast processing. Further, it is a boosting approach so it reduces the bias of decision tree. Due to the reason mentioned above, LGBM provides better accuracy and less training time, less memory. Apart from these advantages, it is found good to work with both type of datasets i.e., small and big size datasets. It provides parallel learning and removes overfitting even when working with small size dataset. The speed of LGBM training is found 20 times faster than other GBDT (Ke et al., 2017; Khonde and Ulagamuthalvi, 2020; Khammassi and Krichen, 2017).

A typical working of boosting approach is shown in the Figure 4. Weak learners are used in the first level of series of learners. The wrong predictions given by lower level learners are sent to the learning phase of higher level learners which they try to remove. This process is repeated till the termination condition is not reached.

Figure 4 Working of boosting method (see online version for colours)



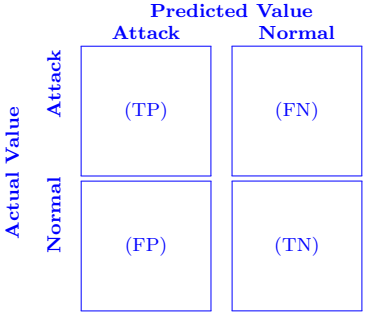
4.3 Training and performance evaluation of the classifier

Before the work of model training starts, the dataset is preprocessed where among many other tasks, the work of feature scaling is done using MinMax scaling due to its faster nature (Abdullah et al., 2018; Wang et al., 2020a). Equation (6) is used for performing MinMax scaling. After this, 10-Fold cross validation is performed for training and testing of the classifiers. After going through the work (Priyadarsini, 2021; Alhaj et al., 2016; Wang et al., 2020b), it is found that if the value of K is kept 10, better performance is achieved so here the value of K is taken 10.

$$F_i = \frac{(F - F_{min})}{(F_{max} - F_{min})} \tag{6}$$

where, F, $F_{\min}$, $F_{\max}$, and F_{i} are the actual, minimum, maximum, and scaled values, respectively. In this work 10-Fold cross validation has been performed for classifier training and validation. Names of the metrics under which performance is evaluated are recall, accuracy, FAR, F1, precision, CKC, and prediction time. For calculating majority of these metrics, the confusion matrix given in Table 2 has been used.

Table 2 Confusion matrix (see online version for colours)



True positive (TP) and false negative (FN) are terms used to describe whether an attack was predicted to occur or not. Similar to this, when a regular flow of traffic is expected to be normal, it is referred to as a true negative (TN), and when it is predicted to be an attack, it is referred to as a false positive (FP).

Equations (7)–12 provide the formulas to calculate all the metrics. These are covered in Chouhan et al. (2023); Chauhan and Atulkar (2023b) in great detail.

1 **Accuracy**: The ratio of accurate predictions to total predictions is known as accuracy. The model's performance is assessed using a variety of indicators in addition to this one, thus a greater score does not automatically indicate better performance.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \times 100 \tag{7}$$

2 **Precision**: Precision is the proportion of correctly anticipated positives to all positives. Less inaccurate positive predictions are made by the model as accuracy increases. This is the statistic to employ if the expense of a false positive is more important.

$$\text{Precision} = \frac{TP}{TP + FP} \times 100 \tag{8}$$

3 **Recall**: Recall is defined as the proportion of correctly predicted positives to all positives in the actual class. The model's anticipated number of false negatives is thought to be low if the recall value is higher. This statistic is the most significant if the cost of a false negative outweighs the benefit of memory.

$$\text{Recall} = \frac{TP}{TP + FN} \times 100 \tag{9}$$

4 **F1-Value**: The weighted average of recall and precision is represented mathematically by the expression F-1. If there are different numbers of classes, then this statistic is important.

$$\text{F1-Score} = \frac{2 * Recall * Precision}{Recall + Precision} \text{ X } 100 \tag{10}$$

5 **False alarm rate (FAR)**: FAR measures the ratio of accurate forecasts to all inaccurate predictions made for the actual class.

$$\text{False Alarm Rate} = \frac{FP}{FP + TN} \text{ X } 100 \tag{11}$$

6 **Cohen's Kappa Coefficient**: When working with multiclass and unbalanced datasets, this statistic is especially helpful. In these circumstances, precision, recall, accuracy, and other metrics cannot provide a whole picture, but Kappa can. CKC is calculated using the following formula:

$$\kappa = \frac{p_o - p_e}{1 - p_e} = 1 - \frac{1 - p_o}{1 - p_e} \tag{12}$$

7 **Testing time**: Once a model has been trained, it is tested using testing dataset. The time a model takes to predict the output of the training data is called the testing time. Testing time presents the more important information than the training time as the it shows the reality about how much time a model will take once it is deployed in the real world.

4.4 Deployment of the classifier

Once the best performing LGBM model is found, the next step is to deploy it in the OpenFlow enabled data plane devices of SD-IoT. Now, the required features are extracted from the live traffic and sent to the model for prediction. The next course of action can be chosen based on the forecast made by the model. Algorithm 1 demonstrates the complete actions.

5 Implementation environment

This section provides the details of the hardware and software used for performing the experiment. All the experiments have been carried out in Ubuntu 18.04 Operating System. The system was equipped with the RAM of size 8GB, and the processor was an AMD PRO A8- 8650B R7 processor. All the software used for performing the tasks are Python based software like Jupyter was used as IDE, Python has been used for implementing the work. The supportive libraries are also Python based such as Numpy, Pandas, Scikit-learn, etc. Ryu has been used as a remote controller in an SDN system that has been emulated using Mininet and Ryu. Ryu was chosen as the SDN controller for two reasons: first, its performance has been reported more quickly than that of other comparable controllers (Chouhan et al., 2019); and second, it was written in Python, making it simple to integrate with all other python libraries utilised in this work.

6 Experiments and discussion

This section includes information on the experimental procedures for feature extraction, dataset development, model training, and model testing. The comparison with other cutting-edge works and a discussion follow.

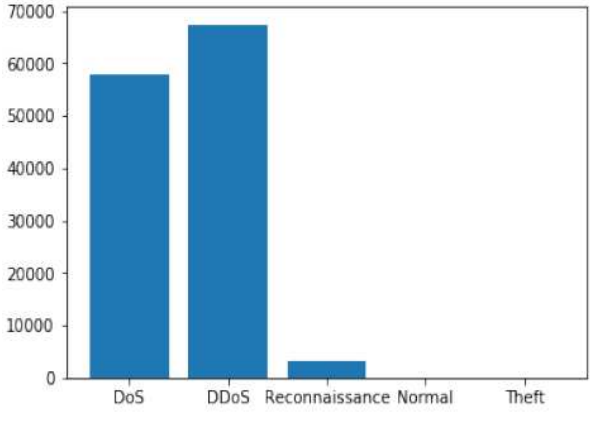
6.1 Dataset preprocessing

In 2018, Bot-IoT (Koroniotis et al., 2019) became available. Contrary to the majority of the other intrusion detection datasets now in use, the dataset's most significant characteristic is that it consists of the traffic related to IoT. For this work, 5% of 5% of the entire dataset has been used which is available in four CSV files where total number of records are 1,83,426. This dataset consists of total 43 features and 3 labels which can be used for binary and multi-class classification of the attacks. The dataset includes regular as well as attack traffic. The names of the attacks are DoS, DDoS, Reconnaissance, and Theft. The number of records corresponding to these attacks are shown in Table 3 and in Figure 5. The dataset's primary issue is that some attack types do not have enough samples to be considered. To remove this issue, SMOTE technique has been used to balance the dataset. After balancing the dataset, the balanced dataset has been shown in Figure 6.

Table 3 Table showing the count of labels

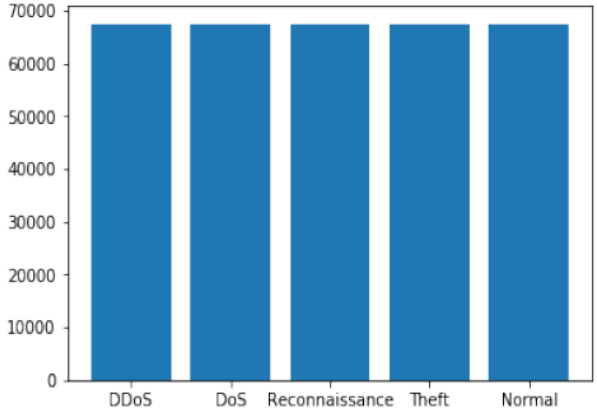
S. no.	*Name of label*	*Count*
1	DDoS	96331
2	DoS	82513
3	Reconnaissance	4554
4	Normal	24
5	Theft	8

Figure 5 Dataset before balancing (see online version for colours)



Additionally, there are not many records for typical traffic. The dataset listed above was used in the majority of the currently available research on IDSs for SDN and IoT contexts. This dataset was not produced on SDN-managed networks, nevertheless.

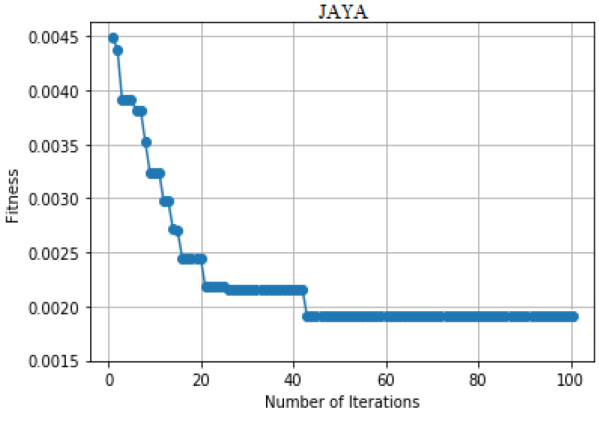
Figure 6 Dataset after balancing (see online version for colours)



6.2 *Selection and analysis of the features*

The objective here is to find the most suitable features using Jaya algorithm. For this, the number of particles taken is 10, and for selecting the best solutions the number of iterations is 100. The fitness function for this has been shown in Figure 7. Here, it is found that the best solution is found in 42^{nd} iteration. After running the process, 6 features are selected as the best features. Names of these features are 'pkSeqID', 'pkts', 'bytes', 'ltime', 'rate', 'Pkts_P_State_P_Protocol_P_SrcIP'.

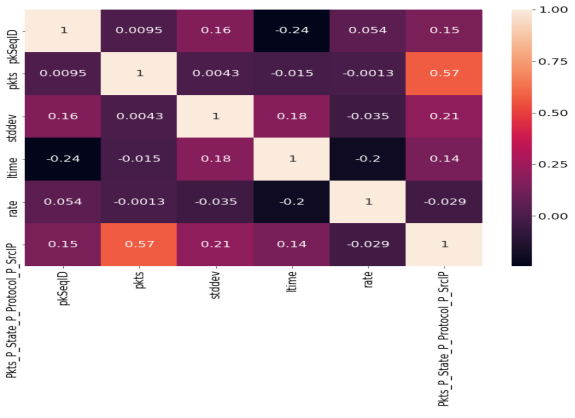
Figure 7 Fitness function (see online version for colours)



To check if the selected features are effective or not, a heat matrix is drawn. In heat matrix, relation among the features is shown; a value of 0(zero) between two features shows no relation between them whereas a value of 1 shows that they are 100% related and hence they are giving same information. The diagonal elements of the matrix would always be 1 because it shows the relation between same feature. The features which are completely related are withdrawn and only one among them is taken. Figure 8 shows the heat matrix of the selected features in the current work. From the figure, it can be stated that the selected

features are effective. Further, effectiveness of the selected features is checked in the next section where model training and testing is done using these features.

Figure 8 Heat matrix of the selected features



6.3 Evaluation of the models

To check the effectiveness of the selected features, the LGBM classifier is trained with two sets of features, first with selected features(SF) called as $LGBM_{SF}$, and second with all features (AF) called as $LGBM_{AF}$. For getting the best values, they are trained and tested using 10-fold cross validation. The results have been compared under recall, accuracy, FAR, F1, precision, CKC, and prediction time. Formulas to calculate all the metrics are shown in equations (7)–(12). These values have been shown in Table 4 and explained in detail in Elhag et al. (2019). The confusion matrix of $LGBM_{AF}$ and $LGBM_{SF}$ are shown in Figures 9 and 10, respectively.

Table 4 Table containing the values of metrics of both the classifiers

	$LGBM_{AF}$	$LGBM_{SF}$
Accuracy	95.508	99.088
Precision	97.313	99.089
Recall	95.508	99.088
F1-Value	96.351	99.084
FAR	1.246	0.329
Prediction	19.862	22.116
CKC	91.554	98.250

Figure 11(a) shows the comparative values of accuracy of both the models named as $LGBM_{SF}$, and $LGBM_{AF}$. The values of accuracy given by $LGBM_{AF}$ and $LGBM_{SF}$ are 95.508% and 99.088%, respectively which shows that $LGBM_{SF}$ is performing better than $LGBM_{AF}$.

Figure 11(b) shows the comparative values of precision of both the models. The values of precision given by $LGBM_{SF}$ is 99.089% and that of $LGBM_{AF}$ is 97.313% which shows that $LGBM_{SF}$ is performing better than $LGBM_{AF}$.

Figure 9 Confusion matrix of $LGBM_{AF}$ (see online version for colours)

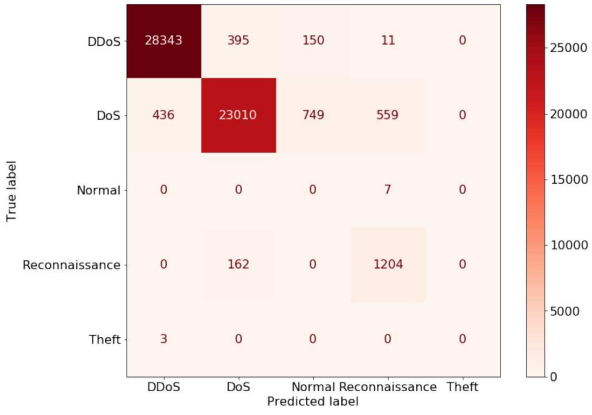


Figure 10 Confusion matrix of $LGBM_{SF}$ (see online version for colours)

Figure 11(c) shows the comparative values of recall of both the models. The values of recall given by $LGBM_{SF}$ is 99.088% and that of $LGBM_{AF}$ is 95.508% which shows that $LGBM_{SF}$ is performing better than $LGBM_{AF}$. Similarly, the values of F1 and FAR given by $LGBM_{AF}$ are 96.351% and 1.246% and those for $LGBM_{SF}$ are 99.084% and 0.329%, which again shows that $LGBM_{SF}$ is performing better. The figures of these are shown in Figure 11(d) and (e), respectively.

Figure 11(f) and (g) show the comparative values of testing time and CKC of $LGBM_{AF}$ and $LGBM_{SF}$. The values given by $LGBM_{SF}$ for these metrics are 22.116 Seconds and 98.250%, respectively whereas these values for $LGBM_{AF}$ are 19.862 Seconds and 91.554%, respectively. It can be observed that $LGBM_{AF}$ is performing better than $LGBM_{SF}$ in testing time whereas $LGBM_{SF}$ is performing better than $LGBM_{AF}$ in case of CKC.

The reason of better performance of LGBM is that LGBM uses gradient-based one side sampling (GOSS) and exclusive feature bundling (EFB) which make it different from other boosting algorithms. GOSS keeps the samples which have large gradient and remove other with lower gradient. EFB is used in a dataset where sparse feature space exists. There might

be many mutual exclusive features which take nonzero values simultaneously. In such cases, exclusive features are bundled into a single feature resulting to fast processing. Further it is a boosting approach so it reduces the bias of decision tree. Due to the reason mentioned above LGBM provides better accuracy and less training time, less memory, working good with both type of datasets i.e., small and big size datasets. It provides parallel learning and removes overfitting even when working small size dataset. The speed of LGBM training is found 20 times faster than other GBDT (Ke et al., 2017; Khonde and Ulagamuthalvi, 2020; Khammassi and Krichen, 2017).

6.4 Model deployment in the OpenFlow enabled devices

It is found that $LGBM_{SF}$ is performing best so it is deployed in all the OpenFlow enabled devices of SD-IoT. For this, the pickle file of the trained model and scaler are taken and deployed. The algorithm to deploy the trained model in all the OpenFlow devices is given in Algorithm 1. After deployment, all the devices of data plane can extract the features from the live traffic using Bro or Argus tools and then these selected features can be given to scaler function which can scale the values. Finally, these values are sent to the classifier for checking whether this is an attack or not. In case of detection of attack, the incoming traffic is stopped immediately. The diagram given in Figure 12 shows the complete work of attack detection in an SD-IoT environment. The working of the detection has been shown in Algorithm 2.

Algorithm 1: Algorithm showing the process of model deployment in the OpenFlow enabled data plane devices of SD-IoT.

Input: **'lgbm.pkl:'** Pickle file of trained model
'scaler.pkl:' Pickle file of scaler function that has been used for feature scaling
Output: Distributed IDS deployed SD-IoT

```
/* Load the Trained Model and Scaler from the
   Memory.                                          */
trained_lgc ← joblib.load(lgbm.pkl)
scaler_fun ← joblib.load(scaler.pkl)
/* Get the ID of all the OpenFlow enabled devices
   */
for i ← 0 , dp in datapath_devices_list do
    datapath_array[i] ← dp
    i ← (i + 1)
/* Deploy the MinMax scaler function and LGBM
   model in each device                             */
for i ← 0 To count(DP_devices) do
    datapath_array[i].deploy ← trained_lgc /* Deploy
       the trained model                            */
    datapath_array.deploy ← scaler_fun /* Deploy the
       standard scaler                              */
```

Algorithm 2: Algorithm for attack detection.

Input: Traffic: Live Network Traffic.
Output: 1: If prediction is an attack
0 If prediction is a normal traffic.

```
/* Load the tool and extract the features */
Extractor = Bro or Argus
Features = Extractor(Traffic)
F = Π(f1,f2,f3,.....,fn) (Features)
/* Load the scaler function and trained model for
   prediction */
trained_lgc ← joblib.load(lgbm.pkl)
scaler_fun ← joblib.load(scaler.pkl)
scaled_F = scaler_fun.transform(F)
/* Get the prediction */
prediction = trained_lgc.predict(scaled_F)
/* return the prediction */
Return prediction
```

6.5 Comparison with other similar works and discussion

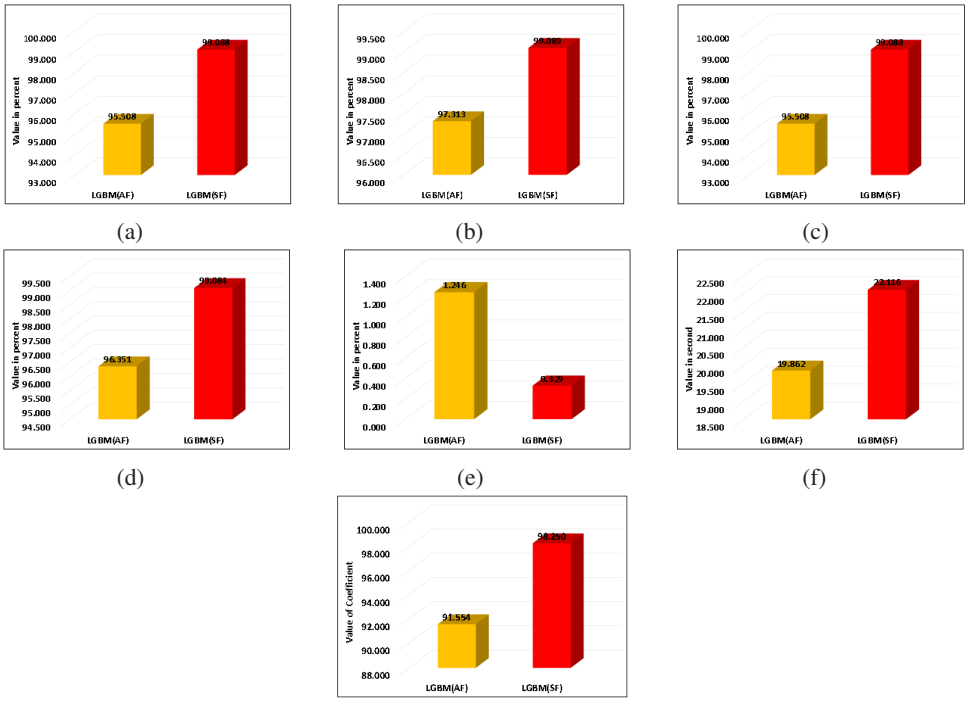
The similar studies have been compared and the results have been shown in Table 5. Khanday et al. (2023) have developed a light weight attack detection system for IoT devices. They have used Logistic Regression, ANN, LSTM, and Linear SVC classifiers and used TON-IoT and BoT-IoT datasets for their result evaluation. The result of LSTM is the highest among other classifiers. These values have been mentioned in Table 5.

Table 5 Comparison with other similar works

Ref. and Year	*Accuracy*	*Recall*	*Precision*	*F1_Score*	*FAR*	*Testing time (in second)*	*Kappa coefficient*
Sarhan et al. (2021)	93.82%	NA	NA	97%	NA	NA	NA
Sarica and Angin (2020)	96.0%	92.75	96.7	94.69	NA%	NA	NA
Friha et al. (2022)	93.29%	NA %	1 %	1%	NA	NA	NA
Khanday et al. (2023)	98.0%	100%	98%	99%	NA%	NA	NA
Saba et al. (2022)	92.85%	NA%	NA	NA	NA%	NA	NA
Almaraz-Rivera et al. (2022)	98.908%	98.857%	98.908%	98.605%	NA	NA	NA
[This work]	**99.088 %**	**99.088%**	**99.089%**	**99.084%**	**0.329%**	**22.116**	**98.250%**

Saba et al. (2022) have developed CNN-based approach for anomaly detection. They have also used BoT-IoT for their result evaluation and achieved accuracy of 92.85%. The work done in Almaraz-Rivera et al. (2022) have performed binary and multiclass classification for attack detection in IoT networks. They have concated BoT-IoT dataset with their own dataset and checked the results. They got the values of recall, F1 ,accuracy, and precision, 98.908%, 98.98.605%, 98.908%, and 98.857%, respectively.

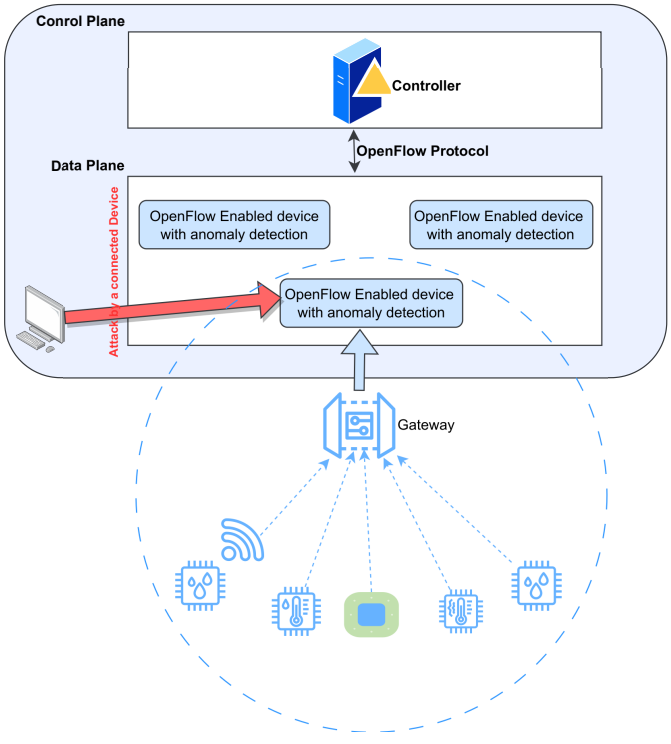
Figure 11 Performance comparison of all the classifiers: (a) accuracy value of all the classifiers; (b) precision value of all the classifiers; (c) recall value of all the classifiers; (d) F1 Value of all the classifiers; (e) FAR value of all the classifiers; (f) testing time of all the classifiers and (g) value of Cohen's Kappa coefficient of all the classifiers (see online version for colours)



The raw packet capture files of four datasets, namely BoT-IoT, UNSW-NB15, TON-IoT, and CSE-CIC-IDS2018 have been used by Sarhan et al. (2021) where they are converting these into corresponding NetFlow format. They labeled the created dataset for both binary and multiclass traffic. They evaluated the performance using an Extra Tree ensemble classifier. They achieved the accuracy of 93.82% and F1 value of 97%.

For SD-IoT based 5G network, an intrusion detection and prevention system is proposed by Sarica and Angin (2020). Their solution is based on automatic feature extraction and classification of the attack using Random Forest classifier in the application layer of SDN. They have used BoT-IoT dataset and achieved the value of accuracy, precision, F1, recall as 96%, 96.71%, 94.69%, and 92.75%, respectively.

Figure 12 Deployment of the trained classifier in SDN controller (see online version for colours)



7 Conclusion and future work

In this work, Jaya optimisation algorithm has been used for the selection of the features that selects 6 features from the BoT-IoT dataset. Now, the LGBM model is trained with two sets of features. First, it is trained with the set of 6 selected features which is named as $LGBM_{SF}$, and then it is trained with the set of all the features which is named as $LGBM_{AF}$. After each training, the model's performance is evaluated under the metrics recall, accuracy, FAR, F1, precision, CKC, and prediction time. The values given by $LGBM_{SF}$ under these metrics are 99.088%, 99.088%, 0.329%, 99.084%, 99.089%, 98.250%, and 22.116 seconds, respectively. The values given by $LGBM_{AF}$ for same metrics are 95.508%, 95.508%, 1.246%, 96.351%, 97.313%, 91.554%, and 19.862 seconds, respectively. It is found that $LGBM_{SF}$ is performing better under all the metrics except prediction time but the difference is manageable, so it is selected for the deployment in the data plane of SD-IoT. Now, all the OpenFlow devices of data plane are deployed with $LGBM_{SF}$ model, so they are ready to detect the attacks which are coming from the infrastructure layer.

Undoubtedly, there is scope of advancement in this work. Some of the suggested advancements are; first, a cooperative approach can be adopted for attack detection where the task of the current work would be to detect the attack in the data plane devices of SD-IoT and then it could send the signal to the controller from where the final decision would be taken. This work would secure the core property of SDN, i.e., centralised decision making.

Second work might be to do the same thing using deep learning approach with an objective to check the performance of the current work i.e., to check how the response time of the model is changing with deep learning approach. Third work could be to do the same work in the multi-controller environment where the auxiliary controller can be assigned the work of rechecking the attack once an attack signal is sent by the data plane devices using the current work. This work would maintain the core property of SDN without degrading the performance. Last enhancement could be to use the machine learning approach in the data plane devices for attack detection while the deep learning model can be used in the controller of SDN for final decision. Reason of doing this is that the controller has sufficient amount of resources to run the deep learning model while the data plane devices have less resources which can be capable to run machine learning models.

References

Abdullah, M., Al-Shannaq, A.S., Almabdy, S., Balamash, A. and Alshannaq, A. (2018) 'Enhanced intrusion detection system using feature selection method and ensemble learning algorithms', *Article in International Journal of Computer Science and Information Security*, Vol. 16, No. 2, pp.48–55.

Ahuja, N., Singal, G., Mukhopadhyay, D. and Kumar, N. (2021) 'Automated DDOS attack detection in software defined networking', *Journal of Network and Computer Applications*, Vol. 187, p.103108.

Alhaj, T.A., Siraj, M.M., Zainal, A., Elshoush, H.T. and Elhaj, F. (2016) 'Feature selection using information gain for improved structural-based alert correlation', *PLoS ONE*, Vol. 11, p.e0166017.

Almaraz-Rivera, J.G., Perez-Diaz, J.A., Cantoral-Ceballos, J.A., Botero, J.F. and Trejo, L.A. (2022) 'Toward the protection of IoT networks: Introducing the LATAM-DDoS-IoT dataset', *IEEE Access*, Vol. 10, pp.106909–106920.

Banitalebi Dehkordi, A. and Soltanaghaei, M. (2020) 'A novel distributed denial of service (DDoS) detection method in software defined networks', *IEEE Transactions on Industry Applications*, pp.1–1.

Chaabouni, N., Mosbah, M., Zemmari, A., Sauvignac, C. and Faruki, P. (2019) 'Network intrusion detection for IoT security based on learning techniques', *IEEE Communications Surveys and Tutorials*, Vol. 21, No. 3, pp.2671–2701.

Chandrashekar, G. and Sahin, F. (2014) 'A survey on feature selection methods', *Computers and Electrical Engineering*, Vol. 40, No. 1, pp.16–28.

Chauhan, P. and Atulkar, M. (2023a) 'A framework for DDoS attack detection in SDN-based IoT using hybrid classifier', *Lecture Notes in Electrical Engineering*, Vol. 946, pp.889–900.

Chauhan, P. and Atulkar, M. (2023b) 'An efficient centralized DDoS attack detection approach for Software Defined Internet of Things', *Journal of Supercomputing*, Vol. 79, No. 9, pp.10386–10422.

Chen, Z., Jiang, F., Cheng, Y., Gu, X., Liu, W. and Peng, J. (2018) 'XGBoost Classifier for DDoS Attack Detection and Analysis in SDN-Based Cloud', *Proceedings - 2018 IEEE International Conference on Big Data and Smart Computing, BigComp 2018*, Shanghai, China, pp.251–256.

Chouhan, R.K., Atulkar, M. and Nagwani, N.K. (2019) 'Performance Comparison of Ryu and Floodlight Controllers in Different SDN Topologies', *1st International Conference on Advanced Technologies in Intelligent Control, Environment, Computing and Communication Engineering, ICATIECE 2019*, IEEE, Bangalore, India, pp.188–191.

Chouhan, R.K., Atulkar, M. and Nagwani, N.K. (2023) 'A framework to detect DDoS attack in Ryu controller based software defined networks using feature extraction and classification', *Applied Intelligence*, Vol. 53, No. 4, pp.4268–4288.

Das, H., Naik, B. and Behera, H.S. (2022) 'A Jaya algorithm based wrapper method for optimal feature selection in supervised classification', *Journal of King Saud University – Computer and Information Sciences*, Vol. 34, No. 6, pp.3851–3863.

Elhag, S., Fernández, A., Altalhi, A., Alshomrani, S. and Herrera, F. (2019) 'A multi-objective evolutionary fuzzy system to obtain a broad and accurate set of solutions in intrusion detection systems', *Soft Computing*, Vol. 23, No. 4, pp.1321–1336.

Friha, O., Ferrag, M.A., Shu, L., Maglaras, L., Choo, K. K.R. and Nafaa, M. (2022) 'FELIDS: Federated learning-based intrusion detection system for agricultural Internet of Things', *Journal of Parallel and Distributed Computing*, Vol. 165, pp.17–31.

Fu, Y., Yan, Z., Cao, J., Koné, O. and Cao, X. (2017) 'An automata based intrusion detection method for Internet of Things', *Mobile Information Systems*, Vol. 2017, pp.1–13.

Haider, S., Akhunzada, A., Mustafa, I., Patel, T.B., Fernandez, A., Choo, K. K.R. and Iqbal, J. (2020) 'A Deep CNN Ensemble Framework for Efficient DDoS Attack Detection in Software Defined Networks', *IEEE Access*, Vol. 8, pp.53972–53983.

Kabacinski, W. and Abdulsahib, M. (2020) 'Wide-sense nonblocking converting-space-converting switching node architecture under XsVarSWITCH control algorithm', *IEEE/ACM Transactions on Networking*, Vol. 28, No. 4, pp.1550–1561.

Kalkan, K., Altay, L., Gür, G. and Alagöz, F. (2018) 'JESS: Joint entropy-based DDoS defense scheme in SDN', *IEEE Journal on Selected Areas in Communications*, Vol. 36, No. 10, pp.2358–2372.

Karan, B.V., Narayan, D.G. and Hiremath, P.S. (2018) 'Detection of DDoS attacks in software defined networks', *Proceedings 2018 3rd International Conference on Computational Systems and Information Technology for Sustainable Solutions, CSITSS 2018*, IEEE, Bengaluru, India, pp.265–270.

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q. and Liu, T.Y. (2017) 'LightGBM: A highly efficient gradient boosting decision tree', *Advances in Neural Information Processing Systems*, December, pp.3147–3155.

Khammassi, C. and Krichen, S. (2017) 'A GA-LR wrapper approach for feature selection in network intrusion detection', *Computers and Security*, Vol. 70, pp.255–277.

Khanday, S.A., Fatima, H. and Rakesh, N. (2023) 'Implementation of intrusion detection model for DDoS attacks in lightweight IoT networks', *Expert Systems with Applications*, Vol. 215, p.119330.

Khonde, S.R. and Ulagamuthalvi, V. (2020) 'Ensemble and feature selection-based intrusion detection system for multi-attack environment', *2020 5th International Conference on Computing, Communication and Security (ICCCS)*, IEEE, Patna, India, pp.1–8.

Khraisat, A., Gondal, I., Vamplew, P. and Kamruzzaman, J. (2019) 'Survey of intrusion detection systems: techniques, datasets and challenges', *Cybersecurity*.

Koroniotis, N., Moustafa, N., Sitnikova, E. and Turnbull, B. (2019) 'Towards the development of realistic botnet dataset in the Internet of Things for network forensic analytics: Bot-IoT dataset', *Future Generation Computer Systems*, Vol. 100, pp.779–796.

Liu, Y., Zhao, B., Zhao, P., Fan, P. and Liu, H. (2019) 'A survey: Typical security issues of software-defined networking', *China Communications*, Vol. 16, No. 7, pp.13–31.

Mishra, S. (2023) 'Cyber defence using attack graphs prediction and visualisation', *International Journal of Communication Networks and Distributed Systems*, Vol. 29, No. 3, pp.268–289.

Niyaz, Q., Sun, W. and Javaid, A.Y. (2017) 'A deep learning based DDoS detection system in software-defined networking (SDN)', *ICST Transactions on Security and Safety*, Vol. 4, No. 12, p.153515.

Polat, H., Polat, O. and Cetin, A. (2020) 'Detecting DDoS attacks in software-defined networks through feature selection methods and machine learning models', *Sustainability (Switzerland)*, Vol. 12, No. 3, p.1035.

Priyadarsini, P.I. (2021) 'ABC-BSRF: Artificial bee colony and borderline-SMOTE RF algorithm for intrusion detection system on data imbalanced problem', *Lecture Notes on Data Engineering and Communications Technologies*, Vol.56, pp.15–29.

Ravi, N. and Shalinie, S.M. (2020) 'Learning-driven detection and mitigation of DDoS attack in IoT via SDN-cloud architecture', *IEEE Internet of Things Journal*, Vol. 7, No. 4, pp.3559–3570.

Roesch, M. (1999) 'Snort – Lightweight intrusion detection for networks', *Proceedings of the 13th Conference on Systems Administration, LISA 1999*, Seattle, Washington, pp.229–238.

Saba, T., Rehman, A., Sadad, T., Kolivand, H. and Bahaj, S.A. (2022) 'Anomaly-based intrusion detection system for IoT networks through deep learning model', *Computers and Electrical Engineering*, Vol. 99, p.107810.

Saeed, A., Ahmadinia, A., Javed, A. and Larijani, H. (2016) 'Intelligent intrusion detection in low-power IoTs', *ACM Transactions on Internet Technology*, Vol. 16, No. 4, pp.1–25.

Sarhan, M., Layeghy, S., Moustafa, N. and Portmann, M. (2021) 'NetFlow datasets for machine learning-based network intrusion detection systems', *Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering, LNICST*, Vol. 371, LNICST, pp.117–135.

Sarica, A.K. and Angin, P. (2020) 'Explainable security in SDN-based IoT networks', *Sensors (Switzerland)*, Vol. 20, No. 24, pp.1–30.

Singh, A., Parizi, R.M., Zhang, Q., Choo, K. K.R. and Dehghantanha, A. (2020) 'Blockchain smart contracts formalization: Approaches and challenges to address vulnerabilities', *Computers and Security*, Vol. 88, p.101654.

Tan, L., Pan, Y., Wu, J., Zhou, J., Jiang, H. and Deng, Y. (2020) 'A new framework for DDoS attack detection and defense in SDN environment', *IEEE Access*, Vol. 8, pp.161908–161919.

Taylor, P.J., Dargahi, T., Dehghantanha, A., Parizi, R.M. and Choo, K.K.R. (2020) 'A systematic literature review of blockchain cyber security', *Digital Communications and Networks*, Vol. 6, No. 2, pp.147–156.

Venkata Rao, R. (2016) 'Jaya: A simple and new optimization algorithm for solving constrained and unconstrained optimization problems', *International Journal of Industrial Engineering Computations*, Vol. 7, No. 1, pp.19–34.

Venkatesh, B. and Anuradha, J. (2019) 'A review of Feature Selection and its methods', *Cybernetics and Information Technologies*, Vol. 19, No. 1, pp.3–26.

Wagner, D. and Soto, P. (2002) 'Mimicry attacks on host-based intrusion detection systems', *Proceedings of the ACM Conference on Computer and Communications Security*, ACM, New York, NY, USA, pp.255–264.

Wang, Y., Hu, T., Tang, G., Xie, J. and Lu, J. (2019) 'SGS: Safe-guard scheme for protecting control plane against DDoS attacks in software-defined networking', *IEEE Access*, Vol. 7, pp.34699–34710.

Wang, M., Lu, Y. and Qin, J. (2020a) 'A dynamic MLP-based DDoS attack detection method using feature selection and feedback', *Computers and Security*, Vol. 88, p.101645.

Wang, Z., Cao, C. and Zhu, Y. (2020b) 'Entropy and confidence-based undersampling boosting random forests for imbalanced Problems', *IEEE Transactions on Neural Networks and Learning Systems*, Vol. 31, No. 12, pp.5178–5191.

Ye, J., Cheng, X., Zhu, J., Feng, L. and Song, L. (2018) 'A DDoS attack detection method based on SVM in software defined network', *Security and Communication Networks*, Vol. 2018, pp.1–8.

Yu, S., Zhang, J., Liu, J., Zhang, X., Li, Y. and Xu, T. (2021) 'A cooperative DDoS attack detection scheme based on entropy and ensemble learning in SDN', *Eurasip Journal on Wireless Communications and Networking*, Vol. 2021, No. 1, p.90.