International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

OML-SDN: Detection of DDoS attacks in SDN using Optimized Machine Learning Methods

Konda Srikar Goud*[1], Srinivasa Rao Giduturi[2]

Submitted: 06/05/2023 **Revised**: 16/07/2023 **Accepted**: 07/08/2023

Abstract: Software-Defined Networks (SDN) is a new technology that allows for future networks' dynamic and efficient design. It redefines the term "network" by allowing network components to be programmed. As a result, network operators can design and control the entire network using the centralized, programmable console architecture. Furthermore, SDN enables network engineers to monitor and control their networks centrally, detecting malicious traffic and link failures. Despite the network's resilience and global visibility, SDN's control plane remains vulnerable to a wide range of security threats, including Distributed Denial of Service (DDoS) attacks, which can render the entire network inaccessible. This study proposes a machine learning-based framework for detecting attack traffic in a centralized SDN environment to address these shortcomings. Kernel Principal Component Analysis (KPCA) is used in this study to reduce the dimensionality of the feature space and Particle Swarm Optimization (PSO) to optimize various SVM parameters. A simplified kernel (s-RBF) was introduced to reduce noise caused by feature differences and increases reliability. We used a weighted support vector machine (WSVM) with PSO. The proposed KPCA-WSVM-PSO model, when compared to other classifiers, achieves the highest attack detection rate, according to the experimental data. We can implement the proposed framework into the SDN control plane to reduce the attacks.

Keywords: *DDoS, Feature Selection, PSO, RBF Kernel, SDN, SVM*

1. Introduction

Software Defined Networks (SDN) is a rapidly emerging model that overcomes the limitations of traditional architectures to comply with exponential growth in volumes of data and technical advances. SDN is a new technology that separates the control and data plane operations. The controller in the control plane manages the control plane operations. The infrastructure layer consists of dump devices and is used to redirect packets. The control plane provides an overall network view, enabling rapid management, forwarding rule generation, and configuration. The controller is the brain of the network and the central entity of the entire network, and all switches in the network follow the controller's decisions. FloodLight, Ryu, Pox, Open Daylight, and Nox are some SDN controllers that provide a set of APIs for developing applications. SDN is widely used in data centers and telecommunications to cater to the demands of next-gen networks. Figure 1 depicts the SDN framework.

However, despite the benefits of SDN described above, the centralization of the SDN architecture raises several security concerns. However, fundamental security issues remain a concern. One such concern is DDoS attacks. This attack will result in the SDN controller becoming isolated from the rest of the network, resulting in the SDN losing its central control. As a result, the attacks can compromise the primary motivation behind SDN, i.e., central authority.

Therefore, DDoS detection and protection techniques are essential because they can better adapt to data centers and cloud technologies, help networks grow in the future, and keep them secure.

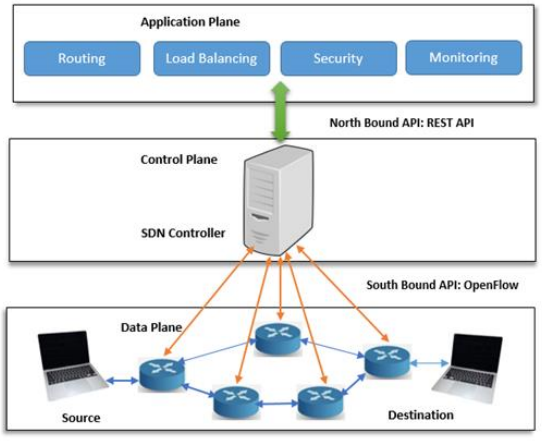

Fig 1. Architecture of SDN

The flood of packets targets the network during an attack. The switch will not find corresponding flow inputs with the forged source and destination IP addresses and will consider these mismatched flow the newest [1]. Subsequently, the device either forwards an incoming message to the controller or forwards it. The controller in SDN determines

[1] *Department of CSE, GITAM School of Technology, GITAM University, Andhra Pradesh, INDIA*
ORCID ID : 0000-0002-8830-0889
[2] *Department of CSE, GITAM School of Technology, GITAM University, Andhra Pradesh, INDIA*
** Corresponding Author Email: kondasrikargoud@gmail.com*

the forwarding path of these packets. Many DDoS attack streams are hidden in legitimate traffic, continuously using and depleting controller resources. Finally, the controller cannot process newly arriving flows, causing the controller to be disabled and the SDN architecture to be lost. Unfortunately, even with an extra controller, the same challenges remain.

As SDN is a new network paradigm, there are limited DDoS attack identification and mitigation solutions. However, the research community is implementing solutions by extending the existing network architecture features without considering the attributes of DDoS attack traffic and SDN benefits. They are still using the old techniques on the SDN controller, which makes the control plane work harder.

Authors in [2] [3] [4] [5] [6] [7] have done a lot of research on analyzing the traffic patterns in an SDN paradigm under a DDoS attack. Still, the majority of these works include the use of unrealistic network topology. Some other authors used conventional datasets to detect the attack on the SDN paradigm. Some other authors simulated malicious traffic patterns but kept the dataset private from researchers. Traditional methods are inapplicable in SDN due to architectural differences among these networks. This paper works on a novel dataset containing essential attack detection features. The conventional datasets, such as NSL-KDD and KDD-cup99, need to be updated; those features do not apply to the SDN paradigm. The author [8] used a Mininet emulator to create the SDN traffic dataset.

The primary contribution of this work is to design an efficient DDoS defensive framework so that network administrators can incorporate it in the SDN controller to identify the attack patterns. The proposed framework integrates WSVM with KPCA and PSO. First, the KPCA extracts the optimal features and the attack detection using the WSVM classifier, where PSO is used to optimize the various SVM parameters. Then, we simulate the experiment on the mininet emulator and deploy the proposed framework on the POX controller. Finally, we compared the performance of our proposed model with recent baseline classifiers and it was observed from the results that our method surpasses the cutting edge methods.

Section 2 discusses the literature review of various methods for detecting DDoS attacks. In Section 3, we discuss multiple models along with the proposed model. In section 4, we evaluate the performance of the proposed work and various classifiers. Finally, in section 5, we discuss the conclusion of the research work.

2. Literature Review

In recent times, many researchers have used ML techniques to mitigate multiple attacks on SDN. This section discusses research on various DDoS detection techniques based on ML and DL techniques. Statistical computations are essential to deal with a wide range of threats. ML algorithms have enabled IDS systems with the capability of making relevant predictions. The authors in [9] [10] [11] proposed various techniques for detecting DDoS attacks, but the major drawback is that they have not used the SDN dataset. Instead, they used the conventional dataset, made for traditional networks and available to the public. It has features that don't apply to the SDN. The authors in [12] [13] developed various machine learning models to investigate DDoS attacks. Still, the major drawback is that they used only two features, which need to be revised to explain the accuracy attained. In our work, we have employed 23 features.

The authors in [4] investigated various Deep Learning methods for detecting DDoS attacks, but the limitation is that they considered 67 features. To improve the accuracy, we need to reduce the features. Therefore, in our work, we have employed 23 features to improve accuracy. The author of [5] worked on an unrealistic dataset because, in the real world, attack traffic would coexist with legitimate traffic. The authors in [6] and [7] created an attack dataset in the SDN environment but did not make it publicly accessible. The authors in [14, 15] introduced a novel architectural model for identifying and limiting low-rate DDoS (LR-DDoS) attacks in SDN. Where the controller is deployed with the IDS and IPS modules to detect attacks using various trained ML and DL algorithms. Here the author used the DoS dataset from the CIC-DdoS2019 dataset. The experiment results revealed that the MLP algorithm performs better (95% accuracy). In the SDN environment, the author in [16] proposed a novel approach combined with an SVM algorithm for classifying attack traffic from benign traffic and KPCA and Genetic algorithms (GA) to determine a selection strategy to enhance its performance. The dataset was used, including benign and attack traffic. Hence, the experimental results showed the accuracy was 98.9%.

The author [17] employed ML models to detect an attack in the SDN paradigm. They used the Scapy tool and the OpenFlow switch to generate the attack traffic pattern to collect the statistics. After the feature engineering step, they trained the linear and polynomial SVM algorithms to classify the traffic. According to the findings, the algorithm produces higher accuracy.

The security architecture proposed in [18] can detect attacks in an SDN paradigm. Adaptive learning models are used to classify the traffic, using a cross-validation approach to ensure that the classification findings are accurate. Despite the optimistic results, they test the security model on a wide range of datasets collected from real-world circumstances. In the context of SDN, detecting DDoS attacks requires a new security model established by the author in [19]. For the model to work successfully, it needs two components

based on machine learning algorithms. First, the K-Means algorithm selects the most correlated characteristics for the data pre-processing, while the KNN algorithm classifies the traffic. Clustering algorithms are the names given to both of these approaches. Their method is more accurate and has a higher recall rate than entropy-based techniques and distributed Self-Organizing Maps (SOM).

The author in [20] proposed a strategy with entropy and DL models. They used a twofold detection technique to detect the attack. Initially, they used entropy detection to identify possibly harmful communication and the CNN model on the second level to identify attack traffic. Finally, they put the approach to the test by using DNN, DT, and SVM algorithms. In terms of accuracy and precision, the CNN model produces better results than the other algorithms.

The author in [21] presented an ensemble method for detecting DDoS attacks by employing various ML models in the context of the SDN to identify suspicious traffic. In terms of accuracy, the ensemble of SVM and SOM algorithms was significantly higher when compared with any other algorithms, with 98.12%. In their proposal, the authors of [22] presented a practical DDoS attack detection framework. On two fronts, they safeguarded the system. They began by using Snort as a tool to detect attacks. They then used a DNN and SVM classifier to categorize the attacks. Regarding categorization, the studies show that DNN is superior to the other approaches. On the other hand, the SVM's accuracy is far higher, at 92.30%.

As the authors [23] demonstrated in their study, DL models effectively detect and categorize DDoS attacks. The DNN model was used to analyze the CICDDoS2019 dataset and effectively classified attack traffic. According to their findings, the DNN model performed well in detecting and categorizing intrusions. However, depending on the dataset used, the findings may differ. As a result, they were able to improve their efforts by working with a variety of datasets.

Most authors used hybrid machine learning models to discover breaches during their analysis. For example, the authors in [24] proposed a DDoS prevention system to detect attack flows based on the SDN architecture. They employed the KNN and SOM algorithms to construct their hybrid strategy. In addition, they used flow statistics generated from SDN switches to identify whether the traffic was benign or malicious. To reduce the consequences of various DDoS attacks in vehicular ad hoc networks, the author [25] worked on a hybrid solution that included neural networks and DT (VANET) principles. The proposed hybrid technique outperforms the separate neural network and DT models. In [26], the author suggested using a hybrid model to deal with DDoS attacks.

Several researchers presented various DDoS detection approaches based on ML for cloud computing and IoT networks. Identifying these attacks with a high degree of precision is a crucial challenge for solutions based on ML. Authors in [27] have researched to detect DDoS attacks against the IoT. The approaches employed time and packet-based sampling techniques to sample the incoming traffic streaming into the SDN data plane. They want to reduce the processing cost of the IDS and DNN models while simultaneously raising the classification performance by implementing these varied sampling methodologies. According to the findings, the model they presented had a higher overall detection rate.

The authors in [28] proposed a security strategy for detecting and mitigating distributed denial service threats in IoT networks. They used a technique called LEDEM, which uses the ML model to detect malicious traffic. The central controller, which in turn controls a variety of specialized controllers, is in charge of LEDEM. They've used a range of security measures for IoT environments, which they divide into categories based on how they operate: mobile IoT and fixed IoT. They used the data to test and assess their security system. Using ensemble techniques, the author in [29] investigated the feasibility of web-hell penetration in an IoT environment. The researchers used principal component analysis with RF and extremely randomized Trees (ET) to choose the essential traits (ET). While RF and ET are helpful in light Internet of Things scenarios, the voting method is more useful in heavy Internet of Things environments.

The authors in [30] suggested an ML-based framework to secure cloud computing facilities. They created a framework named SaE-ELM as an approach for Intrusion Detection Systems. They compared the accuracy of classification attained by their practice to that achieved by widely used ML models such as ANN, DT, and SVM on four different datasets. Even though the model they built takes a little longer to test and train than the SaE-ELM model, the results are reasonably satisfactory. Although SDN provides new capabilities to IDSs through its central management and programmable structure, the quality of training datasets is critical to the detection systems' performance.

The datasets used by many researchers in their work need to be updated, which is the biggest issue. Furthermore, the need for up-to-date datasets is growing as attack characteristics change. For example, they use the datasets LITNET-2020 [31] and the datasets from Boazici University [32] to detect DDoS attacks.

We'll review a few cutting-edge strategies for identifying and combating DDOS attacks. There are various strategies for detecting attacks, some based on machine learning and others on statistical approaches. For example, some researchers [12] are attempting to evaluate DDoS attacks by limiting the number of requests a single user may make.

They examined the switch's two characteristics: the rate of incoming traffic concerning time; and the higher and poorer limits logged for a user utilizing the system. The algorithm uses these two characteristics to determine and classify the traffic. After assessing a representative user behavior pattern, the minimum and maximum values are determined.

The author in [33] uses machine learning as part of their technique for detecting link flooding attacks. The flooding attacks on the Burst Header Packet (BHP) served as the motivation for this collection. They used only fourteen of the twenty-two features in the dataset. According to the trials' findings, the MLP achieves the highest outcomes. The suggested method begins with the Mininet network, which transports the traffic. The process then continues to the next stage. They use Python Application Programming Interface to extract Open flow statistics (API). Pre-processing data is the first step in machine learning algorithms. Next, they train the classifier with the pre-processed information. Finally, the classifier sorts the incoming traffic into groups.

The authors in [3] used various ML algorithms to explain how to identify DDoS attacks. However, it only used one function at any one time, i.e., the flow counter. As a result, the overall number of flows and the quantity of RAM consumed immediately increased when the attack started. During the investigation, they determined that ordinary traffic takes 1.07 seconds for a packet to reach its intended destination. On the other hand, after launching the attack, the timing takes 1.28 seconds longer.

The author in [34] concentrates on recognizing buffer saturation attacks by limiting network connections to hosts that are likely to be blacklisted to prevent these attacks. Compared to other methods considered to be state-of-the-art, Line Switch's implementation as a control plane module not only defends the controller from being attacked but also reduces the time overhead by 30%.

The feature selection approach is crucial in identifying the significant attributes from the high volumes of data. This method eliminates irrelevant attributes and extracts the relevant attributes. Filter-based and wrapper-based techniques are the two categories under which the feature selection methods are organized [35]. While the filter-based strategy retrieves the subset of attributes without consideration of the classification algorithm, the wrapper-based technique depends on the classification algorithm. Most researchers used machine learning techniques and various feature selection approach combinations to create DDoS detection approaches. The author in [36] uses SVM for the classification and ranking approach as their feature selection. The proposed methods for the selection of the correlated features in [37] [38] [39] [40] include a genetic algorithm and decision tree. For further improvements, the author in [41] proposed effective approaches that include consistency-based filtering, INTERACT, and correlation-based feature selection. In [7], they used the BIRCH hierarchical clustering technique to identify the pertinent features, and in [6], bagging with REPTree is trained with the specified attributes. These methods rely on wrappers and only function with the chosen classification techniques.

3. Proposed Methodology

3.1. Kernel Principal Component Analysis

Feature selection plays a crucial role in achieving optimal results. In the proposed model, the s-RBF as a kernel function in SVM, an intern, increases the training time significantly. To reduce the training time, we use KPCA, which converts high-dimensional feature space to low-dimensional feature space. KPCA is a nonlinear analysis technique that is merely a PCA performed using a kernel function. For example, let us consider a1, a2, a3,….., an are 'n' training samples. If the training samples are transformed into the feature set through a nonlinear function θ, we may use PCA to analyze the data (f). The feature set appropriate correlation matrix can be calculated using, , $\sum_{\emptyset} = \frac{1}{n}\sum_{i=1}^{n} \emptyset(a_i)\emptyset(a_i)^T$. The eigen-vectors of $\sum \emptyset$, as a set, must be included in the feature extractors of the feature set, as is easily demonstrated. We can recover the original data with a minimal mean-square error by employing these feature extractors and extracting features from the actual samples. We can convert the high-dimensional feature set into the low-dimensional feature set by using the following equation.

$$Yj = \left[\frac{\sum_{j=1}^{N} \propto_j^1 k(x_j,x)}{\sqrt{\gamma_1^{\alpha}}} \quad \frac{\sum_{j=1}^{N} \propto_j^2 k(x_j,x)}{\sqrt{\gamma_2^{\alpha}}} \quad \ldots \quad \frac{\sum_{j=1}^{N} \propto_j^m k(x_j,x)}{\sqrt{\gamma_m^{\alpha}}} \right]^T \qquad (1)$$

Where, α1, α2, α3... αm, are the values corresponding to the m's largest eigen values $(\gamma_1^{\alpha}, \gamma_1^{\alpha}, \gamma_1^{\alpha}, .., \gamma_m^{\alpha})$. where $\propto_j^1$ indicates the jth transformed feature αi.

3.2. Support Vector Machine

SVM [38] is helpful for various purposes, including regression and classification applications. However, its primary purpose is to generate the best fit line or decision boundary that can divide an n-dimensional space into classes. This allows us to place any new data points in the appropriate category. The points (a.k.a. Support vectors) that are geographically closest to the decision plane and that contribute to the process of determining the margin of the decision plane. It is preferable to have a decision plane with a high margin of error rather than a small margin. The optimal decision plane, also known as a hyper plane, is the one that separates the various classes the most effectively. It is effective for classification issues involving many features, and we select it as the solution to our problem. However, when applied to our dataset, it does not yield adequate results because the characteristics are highly correlated,

which makes it impossible to locate a decision boundary using such features.

3.3. Weighted-SVM

The main principle behind w-SVM is to give weights to various data features in the training set. As a result, w-SVM can learn the decision surface in accordance with the relative weights assigned to each feature. We used the results and the point bi-serial co-relation between each attribute to construct their weights. Additionally, we consider various SVM kernels for the classification task with different train and test data split ratios.

Let 'D' consist of the original dataset's properties as a matrix with n rows and p columns, where

$$D = \begin{pmatrix} D_{11} & D_{12} \ldots\ldots D_{1p} \\ D_{21} & D_{22} \quad D_{2p} \\ & \vdots \\ D_{n1} & D_{n1} \quad D_{np} \end{pmatrix} \quad (2)$$

Let the relative importance of each characteristic be represented by a diagonal matrix, φ_{qXq} , in the sense that

$$\varphi = \begin{pmatrix} \varphi_{11} & 0 & 0 \\ 0 & \varphi_{22} & 0 \\ & \vdots & \\ 0 & 0 & \varphi_{np} \end{pmatrix} \quad (3)$$

With $tr(\varphi) = 1$. The formula for assigning separate weights is as:

$$\varphi_{ij} = \frac{|\gamma_{x_j y_i}|}{\sum_{j=1}^{p} |\gamma_{x_j y_i}|} \qquad for\ j= 1, 2,..,p,\ i =1 \qquad (4)$$

Point bi-serial correlation between the ith feature Xi and Yi reply in binary form is denoted as $\varphi_{x_i y_i}$. For each Xi, $\varphi_{x_i y_i}$ is calculated as:

$$\varphi_{x_i y_i} = \frac{D'_{+1} - D'_{-1}}{S_X} \sqrt{\frac{n_{p+1}\, p-1}{n-1}} \quad (5)$$

Where D'_{+1}, D'_{-1} represents the means of the continuous variable (D). p+1, p-1 represents the point of proportion in group (+1,-1), Py = $\frac{n_y}{n}$, y = +1, -1, and

$$Sx = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(D_i - D')^2} \quad (6)$$

Standard deviation of the ith feature D_i, represented by a sample. Let K be a new feature set generated by combining D and φ.

$$K = D\,\varphi \quad (7)$$

Due to this, the new data matrix K is represented as

$$Z = \begin{pmatrix} K_{11} & K_{12} \ldots\ldots K_{1p} \\ K_{21} & K_{22} \quad K_{2p} \\ & \vdots \\ K_{n1} & K_{n1} \quad K_{np} \end{pmatrix} \quad (8)$$

By using the formula zij = Xij φ_{ij}, where i = 1, 2,..., n and j = 1, 2,..., p. So, we can calculate Z as (Z1, Z2,..., Zp). If we assume that there are n samples and that each sample has K features and that each point belongs to the one or the other groups, then we can write yi = ±1. As a result, the shape of the training feature set is represented as "zi yi," where i might be 1, 2, …, n. The data point zi is the one nearest to the hyper plane, and the weight vector w is orthogonal to the hyper plane if and only if $w' z = 0$. This allows us to express the hyper plane's equation as follows, for any bias b:

$$w' z + b = 0 \quad (9)$$

To normalize w using minimum zi, it is necessary to ensure that

$$|w' z_i + b| = 1 \quad (10)$$

In practice, SVM is implemented by selecting w and b so that the training data (in this case, zi and yi) can be characterized by the equation:

$$y_i(z_i w+b)-1 \geq 0 \quad (11)$$

The goal of the SVM is, to increase the gap between the $y_i = 1$ and $y_i = +1$ classes of samples. As a result, the Quadratic Programming (QP) problem formulation and optimization in (11) adhere to the same pattern as observed in [3].

$$min_{\alpha} \left(\frac{1}{2} \sum_{i=1}^{n} \sum_{j-1}^{n} \alpha_i \alpha_j y_i y_j z_i z_j - \sum_{i=1}^{n} \alpha_i \right) \quad (12)$$

In which the Lagrange multiplier αi ≥0, with w = $\sum_{z_i \in sv} \alpha_i y_i z_i$ are used. In this research we have conducted 5 fold CV to render the classifier more stable on both datasets. The stratified approach helps in splitting each class of the dataset in to equal proportions. Several performance metrics were used to assess how well the proposed technique predicted outcomes.

3.4. Particle Swarm Optimization

Kennedy and Eberhart [42], Inspired by the social behavior of groups of individuals, such as the birds flocking or fish school together, propose the PSO model. In Particle Swarm Optimization (PSO), a swarm of particles is used to find the best possible solution and then move to that location. In every cycle, each particle will move forward in the direction that improves its position relative to other particles and the entire universe. The individual particle's motion can be described as:

$$V_i^{t+1} = W.V_i^t + C_1U_1^t(P_{b1}^t - P_i^t) + C_2U_2^t(g_b^t - P_i^t) \quad (13)$$

$$P_i^{t+1} = P_i^t + \alpha v_i^{t+1} \quad (14)$$

Where, t is the iteration number, C_1 and C_2 are the learning factors, and U_1 and U_2 are the normal-distributed positive random values between 0 and 1. Constraint factor 'α' representing a variable with the potential to influence the velocity value. Constant 'w' stands for the inertial weight coefficient. A particle's velocity is represented by a value 'p_i' The best possible position, p_b, is represented by particle 'i', while the best possible position, g_b, is represented by any of the other particles in the swarm.

3.5. The Simplified Radial Basis Function (s-RBF) kernel

The Radial Basis Function (RBF) kernel is commonly used in SVMs for various classification problems. It is known for its effectiveness in many cases. However, it can produce a large number of support vectors when applied to datasets with attributes that vary significantly. These support vectors can increase the training time and potentially degrade the model's performance.

In networking scenarios, network flow data often includes attributes related to different protocols, which can vary significantly. This variation in attribute sets can make it challenging to use a standard RBF kernel effectively.

To address these challenges, an s-RBF kernel has been developed as follows.

$$K(x_i, x_j) = exp\left[\frac{-\left|\frac{x_{i-m_v}}{m_t} - \frac{x_{j-m_v}}{m_t}\right|^2}{\sigma^2}\right] \quad (15)$$

Where, $K(x_i, x_j)$ represents the similarity or kernel value between data points x_i and x_j.

$$mv_i = \frac{1}{n}\sum_{i=1}^{n} P_{ij} \quad (16)$$

$$ms_i = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}\left(P_{ij} - mv_i\right)^2} \quad (17)$$

The above formulas compute the mean value (mv_i) and standard deviation (ms_i) of specific features across training samples. "n" is the number of samples, and "P_{ij}" represents attributes. These calculations help understand feature distribution and variability, and the "s-RBF" kernel is mentioned, indicating its potential use in machine learning.

The s-RBF kernel is an improved kernel function used to measure the similarity between data points while accounting for attribute differences across samples. When using SVMs, selecting the right parameters, such as C (which controls the trade-off between maximizing margin and minimizing error) and σ (the width of the radial basis function), is critical for model performance. To find the optimal values for C and σ, various optimization approaches can be employed, including Genetic Algorithms (GA), Simulated Annealing (SA), Particle Swarm Optimization (PSO), and grid search combined with cross-validation. These methods automate the search for the best parameter values, improving the SVM model's suitability and accuracy for a given task. In this study, we have employed PSO for optimization.

3.6. Proposed framework

Here we'll discuss the proposed KPCA-WSVM-PSO framework. The primary goal of this framework is to increase the model's accuracy. (a) By identifying the optimal features using the KPCA method and (b) determining the best feature weights γ and C values via PSO. The initial step is to pre-process the "SDN DDoS attack" dataset by the KPCA-based feature selection. It eliminates irrelevant and noisy attributes and extracts the relevant features. The first 10 features with the highest scores are selected, and created a new subset. The KPCA is used to filter out many features that are not very significant, hence reducing the computational load on the SVM classifier. The kernel parameters calculate the feature weights using a PSO-based technique. Feature weighting is employed to estimate each feature's weight based on its presence in the training set. Without feature weighting, optimizing the values of C and γ becomes essential. If we examine n features, then n + 2 decision variables are required for feature weighting. All n variables can take on values between 0 and 1, with the sum always 1. Figure 2 provides a graphical representation of this solution. We use this to represent particles, and PSO was involved in finding their optimum parameter values.

Moreover, we develop a threshold function denoted by $U_\delta(.)$ to eliminate noisy features which are redundant and have less impact on output. Instead, we include those features which have a higher potential for classification. In reality, the threshold function acts as a selector to eliminate the redundant features in the final stage. This function has a domain that consists of a set of feature weights and a range of adjusted feature weights.

$$U_\delta(a_i) = \begin{cases} 0 & if\ a_i \leq \delta, \\ a_i & if\ a_i > \delta, \end{cases} \quad (18)$$

Where, $0 \leq \delta \leq 1$ and a_i represents the importance of the ith feature.

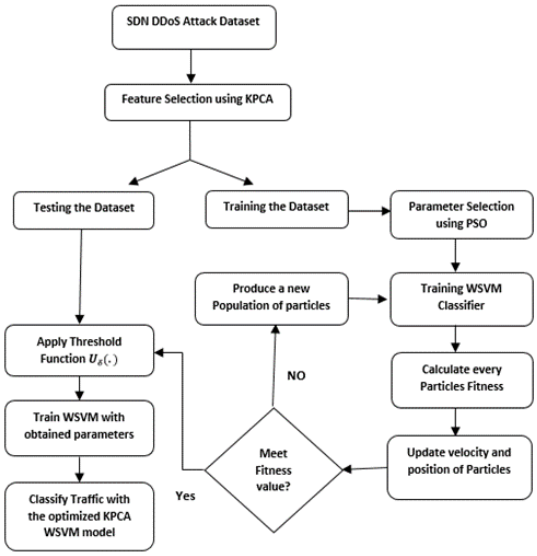

Fig 2. The proposed framework

Therefore, the weighted matrix, $\alpha = (\alpha_1, \alpha_2, \alpha_3 \ldots, \alpha_d)$ can be obtained by using the normal form in the following way:

$$\alpha_k = \frac{U_\delta(a_k)}{\sum_{i=1}^{n} U_\delta(a_i)} \text{ , where } k \text{ value is } 1,2,3, \ldots, n. \quad (19)$$

Hence, we train the dataset in the following way.

i. We should use the KPCA strategy to predict the top features. Then, in the later phase, we used the selected features, where the PSO was used to find the optimal values for the weights assigned to the features and the kernel parameters.

ii. Use the cross-validation technique while splitting the dataset into train and test sets.

iii. We must individually adjust the PSO's parameters for each training set. Start by randomly generating the particle positions and velocities and then set the inertia weight and maximum number of repetitions in the learning parameters.

iv. Train Weighted SVM following the particle values.

v. Determine each particle's fitness function using the formula correct_classified/total_samples), where total_samples stands for the training samples and correct_classified stands for the successfully classified samples.

vi. Maintaining each particle's velocity and position using (13).

vii. Until it reaches the necessary generation count, we must generate a new swarm of particles before moving on to the next step (iv).

viii. Determine the optimal global position, Pgbest, to use as a guide for calculating the feature weights and kernel parameters. Then, the threshold function U_δ filters out duplicate genes.

ix. Use the obtained parameters to train the WSVM classifier.

x. We should use the most effective method to classify the attack traffic.

4. Simulation and Results Discussion

In this section, we evaluate the proposed methodology's results with those obtained using the SVM, KPCA-SVM, and KPCA-PSO-SVM classifiers. For the experiment, we considered the following datasets and simulation environment.

4.1. Dataset Description

The "SDN DDOS attack Dataset" [43], developed in the SDN paradigm, was used in our work, which is open to researchers. It includes 24 features and a total of 1.04 lakh network traffic records, including TCP, ICMP, and UDP traffic, labeled with both the ordinary and the attack traffic class labels. In addition to attributes that define the source and the target system, the dataset provides statistical information such as "byte count, duration in seconds, packet rate, and packets per flow." The below figure represents the features in the dataset.

S. No	Feature Name	S. No	Feature Name
1	dt	13	Byteperflow
2	switch	14	Pktrate
3	Src	15	Pairflow
4	Dst	16	Protocol
5	pktcount	17	port_no
6	Bytecount	18	tx_bytes
7	Dur	19	rx_bytes
8	dur_nsec	20	tx_kbps
9	tot_dur	21	rx_kbps
10	Flows	22	tot_kbps
11	Packetins	23	label
12	Pktperflow		

Fig. 3. Features in the SDN DDoS attack Dataset

S. No	Feature Name	S. No	Feature Name
1	duration	22	is_guest_login
2	protocol_type	23	count
3	service	24	srv_count
4	Flag	25	serror_rate
5	src_bytes	26	srv_serror_rate
6	dst_bytes	27	rerror_rate
7	Land	28	srv_rerror_rate
8	wrong_fragment	29	same_srv_rate
9	Urgent	30	diff_srv_rate
10	Hot	31	srv_diff_host_rate
11	num_failed_logins	32	dst_host_count
12	logged_in	33	dst_host_srv_count
13	num_compromised	34	dst_host_same_srv_rate
14	root_shell	35	dst_host_diff_srv_rate
15	su_attempted	36	dst_host_same_src_port_rate
16	num_root	37	dst_host_srv_diff_host_rate
17	num_file_creations	38	dst_host_serror_rate
18	num_shells	39	dst_host_srv_serror_rate
19	num_access_files	40	dst_host_rerror_rate
20	num_outbound_cmds	41	dst_host_srv_rerror_rate
21	is_host_login		

Fig. 4. Features in the NSL-KDD Dataset

To verify the efficiency of our approach, we use a different dataset, i.e., the NSL-KDD dataset. In the simulation, we used a total of 41 features, which included 1.08 lakh records. The dataset consists of Probe, DoS, R2L, U2R, etc. Since there are no duplicates in either dataset, the machine learning classifiers will not get biased toward more common occurrences. Therefore, we evaluate the performance of the proposed technique using two different data sets.

4.2. Simulation Environment

Before beginning any simulation work for an SDN network, we must select a controller. In the experiment, we will use a POX-based controller. It's well-known for being a responsive and flexible controller. Furthermore, Mininet is a software-defined networking (SDN) tool that may emulate conventional networks in a computer. Therefore, we may use it to simulate network architecture of any size, and we can apply the resulting code to real-world networks. For this reason, we are considering using the Mininet 2.0.0 emulator.

We deployed the proposed framework on an Intel core i7 processor with 16 GB of RAM. Mininet version 2.0.0, compatible with OpenFlow version 1.3, has been installed on the Virtual Box. Mininet was used to create a topology that includes 15 switches and 64 computers. As part of the experiment, a single host will employ IP spoofing to attack the other hosts with the IP address 10.0.0.1.

4.3. Performance Metrics

The performance evaluation phase is crucial while developing a robust ML model. Metrics used to measure the efficiency of a model are known as performance metrics. We can use these criteria to determine how well our model fits the data. We can improve the model's performance by adjusting these hyper-parameters. Performance metrics help evaluate how well a particular ML model generalizes to new data it has never seen before, which is a crucial goal of every ML model. In this study, we used the metrics below to measure ML models' performance while detecting DDoS attacks in SDN.

Models	Accuracy (%)	Precision (%)	Recall (%)
KPCA-PSO-WSVM	98.835	98.542	99.137
KPCA-PSO-SVM	98.014	97.127	98.955
PCA-PSO-SVM	97.306	96.162	98.546
PSO-SVM	95.601	94.145	97.250
SVM	91.435	89.247	94.223

4.3.1. Accuracy

Accuracy is one of the most simple and crucial classification metrics. The proportion of successful predictions with the total number of predictions determines it.

$$\text{Accuracy} = \frac{Tp+Tn}{Tp+Tn+Fp+Fn} \quad (20)$$

4.3.2. Precision

Precision is employed to address the drawbacks of accuracy. We can measure precision as the proportion of True positives with the sum of expected correct predictions. (i.e., true positives and false positives). The precision is inversely proportional to the false positives. The number of false positives goes down with the increase in precision.

$$\text{Precision} = \frac{Tp}{Tp+Fp} \quad (21)$$

4.3.3. Recall or Sensitivity

It is similar to precision, which determines how many false positives were incorrectly detected. We measure recall as the proportion of true positives with the total number of predictions that are correctly predicted positive and falsely predicted negatives (i.e., true positives and false negatives).

$$\text{Recall} = \frac{Tp}{Tp+Fn} \quad (22)$$

4.3.4. F- Score

F-score is used to assess the classifier by comparing the proportion of correct predictions to the total number of possible accurate predictions. We can obtain F- Score by combining the values of Precision and Recall. It is a metric that incorporates Recall and accuracy into a single score. As a result, we compute the F1 Score as the harmonic mean of both accuracy and Recall, with both contributing equally to the final Score.

$$\text{F Score} = 2 * \frac{Precision * Recall}{Precision + Recall} \quad (23)$$

4.4. Results Discussion

In this experiment, we compared the results of our proposed model with the SVM, PCA-SVM, KPCA-SVM, SVM-PSO, and KPCA-PSO-SVM classifiers in terms of various evaluation metrics. We evaluate the classifiers mentioned above using two popular publicly available datasets, the "SDN DDoS attack dataset" and the "NSL KDD" dataset. We assess the accuracy of these classifiers using the "Leave-One-Out" cross-validation. We used a single instance from the original dataset for testing and the remaining samples for training. We repeated this several times to guarantee that we used each instance as test data at least once. Besides, we repeated the testing ten times on the dataset to make it more realistic. Finally, we evaluated the performance of the classifiers by averaging the results from these ten tests and the number of selected features in each trial.

Moreover, we have dropped the features with PSO weight less than or equal to the threshold value (δ). To find the optimal threshold value, we began with 0.2, increased it by 0.1, and recorded the results. For both datasets, we found the threshold value ranges from 0.3 to 0.5. The table 1 &2 below illustrates the experimental results.

Table 1. Performance of various classifiers on SDN DDoS Attack Dataset

Table 2. Performance of various classifiers on NSL KDD Dataset

Models	Accuracy (%)	Precision (%)	Recall (%)
KPCA-PSO-WSVM	98.287	98.713	97.720
KPCA-PSO-SVM	95.037	91.798	93.390
PCA-PSO-SVM	94.783	90.904	92.649
PSO-SVM	94.317	89.124	91.403
SVM	88.082	87998	87.167

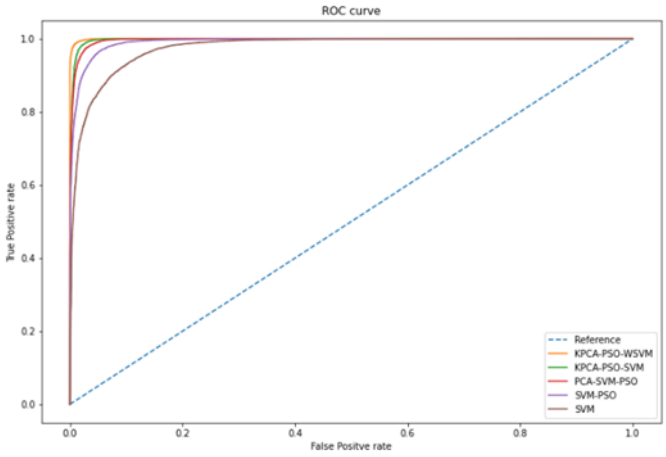

Fig. 5. Comparison of ROC Curve for Various methods

The results show that the single SVM classifier achieves the least accuracy and time consuming on both datasets as it relies on trial judges. Moreover, the accuracy achieved by the KPCA-SVM classifier is better than a single SVM. At the same time, PCA is a powerful tool to extract more relevant features and eliminates redundant and noisy features. Furthermore, using a kernel function with PCA allows for identifying more primary components than is possible with traditional PCA, leading to better overall performance. Finally, the above results show that feature selection plays a prominent role in increasing classification accuracy.

Furthermore, the results of SVM-PSO state that the PSO plays an essential role in selecting features that significantly help to predict malicious traffic. The selection of kernel parameters has a significant impact on classification accuracy. The results of KPCA-PSO-SVM state that employing feature selection by KPCA and determining kernel parameters by PSO increases the SVM accuracy. Finally, the proposed model KPCA-PSO-WSVM achieves the highest accuracy in detecting the attack traffic. The above results further demonstrate that the proposed approach provides higher performance when compared to PSO-SVM. In addition to this, employing the threshold function U_δ (.) in averaging the selected features is very successful in effectively reducing the total number of selected features.

In the table below, we compare our proposed work with the existing research on DDoS attack detection using ML models. Researchers suggested various techniques for detecting DDoS attacks, but the major drawback is that they have not used the SDN dataset. Instead, they used the conventional dataset, made for traditional networks and available to the public. It has features that don't apply to the SDN.

The datasets used in the recent studies are CIC-DoS, KDD Cup'99, UNB-ISX, CAIDA 2016, and CICIDS2017. The fact is that there is a need to update these datasets with the

latest attack patterns and which is the biggest issue. The demand for up-to-date datasets is growing as attack characteristics change. Our work used an SDN dataset simulated on an SDN environment with the latest attack traffic patterns. The table 2 below presents the results of research carried out by other researchers in classifying DDoS attacks.

Table 3. Comparison of Related Studies with the Proposed System

Method	Dataset	Accuracy (%)
Author in [15] designed a model, with six different ML classifiers to detect LR-DDoS attacks in SDN paradigm.	CIC-DoS 2017	95.12
Author in [16] proposed a model with KPCA-GA-SVM to detect DDoS attack in SDN environment.	NSL-KDD	98.24
Author in [17] designed a model with linear and polynomial SVM.	KDD Cup'99	97.2
Author in [21] proposed an ensemble model with SVM and SOM.	NSL-KDD	98.12
Author in [23] proposed Deep learning model on CICDDOS2019 dataset.	CIC-DDOS2019	97.52
The authors in [30] suggested an ML-based framework named SaE-ELM using ANN, DT, and SVM.	NSL-KDD, and CIC-IDS 2017	95
Proposed model, KPCA-PSO-WSVM.	SDN DDoS Attack, NSL-KDD	98.835

The results show that ML models effectively identify DDoS attack traffic. Based on that, the proposed framework outperforms the state-of-the-art classifiers in terms of accuracy. Therefore we conclude that the KPCA-PSO-WSVM framework effectively addresses the feature selection and DDoS attack classification problems.

5. Conclusion

In this research work, we designed an efficient KPCA-PSO-based Weighted Support Vector Machine framework to classify the traffic as normal and attack in the SDN environment. This novel framework involves two stages. First, we employ the KPCA technique to retrieve the most critical features from a feature space. The kernel function in PCA can reduce more principal components than PCA, leading to better performance. Then, we train the PSO-WSVM classifier with these features. PSO eliminates noisy and redundant features, calculates the importance score, and assigns weights to different features. Moreover, s-RBF is used as a kernel in SVM as they perform better in classifying complicated datasets effectively by creating more complex decision boundaries. We have conducted an experiment on two different datasets for the classification of DDoS attacks, where the proposed KPCA-PSO-WSVM framework outperforms all other previously reported results with 98.853% accuracy. Furthermore, we can deploy the proposed technique on the SDN controller as it utilizes low computational resources in identifying malicious traffic patterns. In future research, we need to increase new attack patterns and analyze various deep-learning models to classify traffic patterns accurately on large datasets.

Conflicts of interest

The authors declare no conflicts of interest.

References

[1] Cui, Yunhe, Qing Qian, Chun Guo, Guowei Shen, Youliang Tian, Huanlai Xing, and Lianshan Yan. "Towards DDoS detection mechanisms in software-defined networking."Journal of Network and Computer Applications 190 (2021): 103156.

[2] Palmieri, Francesco. "Network anomaly detection based on logistic regression of nonlinear chaotic invariants."Journal of Network and Computer Applications 148 (2019): 102460.

[3] da Silva, Anderson Santos, Juliano Araujo Wickboldt, Lisandro Zambenedetti Granville, and Alberto Schaeffer-Filho. "ATLANTIC: A framework for anomaly traffic detection, classification, and mitigation in SDN." In NOMS 2016-2016 IEEE/IFIP Network Operations and Management Symposium, pp. 27-35. IEEE, 2016.

[4] Niyaz, Quamar, Weiqing Sun, and Ahmad Y. Javaid. "A deep learning based DDoS detection system in software-defined networking (SDN)."arXiv preprint arXiv:1611.07400 (2016).

[5] Santos, Reneilson, Danilo Souza, Walter Santo, Admilson Ribeiro, and Edward Moreno. "Machine learning algorithms to detect DDoS attacks in SDN."Concurrency and Computation: Practice and Experience 32, no. 16 (2020): e5402.

[6] Myint Oo, Myo, Sinchai Kamolphiwong, Thossaporn Kamolphiwong, and Sangsuree Vasupongayya. "Advanced support vector machine-(ASVM-) based

detection for distributed denial of service (DDoS) attack on software defined networking (SDN)."Journal of Computer Networks and Communications 2019 (2019).

[7] Cui, Yunhe, Lianshan Yan, Saifei Li, Huanlai Xing, Wei Pan, Jian Zhu, and Xiaoyang Zheng. "SD-Anti-DDoS: Fast and efficient DDoS defense in software-defined networks."Journal of Network and Computer Applications 68 (2016): 65-79.

[8] Octopress, Mininet emulation Software, 2018. http://www.mininet.org/. (Accessed 19 July 2008).

[9] Goud, Konda Srikar, and Srinivasa Rao Gidituri. "Security Challenges and Related Solutions in Software Defined Networks: A Survey."

[10] Wang, Bing, Yao Zheng, Wenjing Lou, and Y. Thomas Hou. "DDoS attack protection in the era of cloud computing and software-defined networking."Computer Networks 81 (2015): 308-319.

[11] Latah, Majd, and Levent Toker. "Towards an efficient anomaly-based intrusion detection for software-defined networks."IET networks 7, no. 6 (2018): 453-459.

[12] Buragohain, Chaitanya, and Nabajyoti Medhi. "FlowTrApp: An SDN based architecture for DDoS attack detection and mitigation in data centers." In 2016 3rd International Conference on Signal Processing and Integrated Networks (SPIN), pp. 519-524. IEEE, 2016.

[13] Xie, Junfeng, F. Richard Yu, Tao Huang, Renchao Xie, Jiang Liu, Chenmeng Wang, and Yunjie Liu. "A survey of machine learning techniques applied to software defined networking (SDN): Research issues and challenges."IEEE Communications Surveys & Tutorials 21, no. 1 (2018): 393-430.

[14] Shin, Seungwon, and Guofei Gu. "Attacking software-defined networks: A first feasibility study." In Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking, pp. 165-166. 2013.

[15] Perez-Diaz, Jesus Arturo, Ismael Amezcua Valdovinos, Kim-Kwang Raymond Choo, and Dakai Zhu. "A flexible SDN-based architecture for identifying and mitigating low-rate DDoS attacks using machine learning."IEEE Access 8 (2020): 155859-155872.

[16] Sahoo, Kshira Sagar, Bata Krishna Tripathy, Kshirasagar Naik, Somula Ramasubbareddy, Balamurugan Balusamy, Manju Khari, and Daniel Burgos. "An evolutionary SVM model for DDOS attack detection in software defined networks."IEEE Access 8 (2020): 132502-132513.

[17] Kyaw, Aye Thandar, May Zin Oo, and Chit Su Khin. "Machine-Learning Based DDOS Attack Classifier in Software Defined Network." In 2020 17th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON), pp. 431-434. IEEE, 2020.

[18] Janarthanam, S., N. Prakash, and M. Shanthakumar. "Adaptive learning method for DDoS attacks on software defined network function virtualization."EAI Endorsed Transactions on Cloud Systems 6, no. 18 (2020).

[19] Tan, Liang, Yue Pan, Jing Wu, Jianguo Zhou, Hao Jiang, and Yuchuan Deng. "A new framework for DDoS attack detection and defense in SDN environment."IEEE Access 8 (2020): 161908-161919.

[20] Wang, Lu, and Ying Liu. "A DDoS attack detection method based on information entropy and deep learning in SDN." In 2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), vol. 1, pp. 1084-1088. IEEE, 2020.

[21] Deepa, V., K. Muthamil Sudar, and P. Deepalakshmi. "Design of ensemble learning methods for DDoS detection in SDN environment." In 2019 International Conference on Vision Towards Emerging Trends in Communication and Networking (ViTECoN), pp. 1-6. IEEE, 2019.

[22] Karan, B.V.; Narayan, D.G.; Hiremath, P.S. Detection of DDoS Attacks in Software Defined Networks. In Proceedings of the 2018 3rd International Conference on Computational Systems and Information Technology for Sustainable Solutions (CSITSS), Bengaluru, India, 20–22 December 2018; pp. 265–270.

[23] Cil, Abdullah Emir, Kazim Yildiz, and Ali Buldu. "Detection of DDoS attacks with feed forward based deep neural network model."Expert Systems with Applications 169 (2021): 114520.

[24] Nam, Tran Manh, Phan Hai Phong, Tran Dinh Khoa, Truong Thu Huong, Pham Ngoc Nam, Nguyen Huu Thanh, Luong Xuan Thang, Pham Anh Tuan, and Vu Duy Loi. "Self-organizing map-based approaches in DDoS flooding detection using SDN." In 2018 International Conference on Information Networking (ICOIN), pp. 249-254. IEEE, 2018.

[25] Adhikary, Kaushik, Shashi Bhushan, Sunil Kumar, and Kamlesh Dutta. "Hybrid algorithm to detect DDoS attacks in VANETs."Wireless Personal Communications 114, no. 4 (2020): 3613-3634.

[26] Hosseini, Soodeh, and Mehrdad Azizi. "The hybrid technique for DDoS detection with supervised learning algorithms."Computer Networks 158 (2019): 35-45.

[27] Ujjan, Raja Majid Ali, Zeeshan Pervez, Keshav Dahal, Ali Kashif Bashir, Rao Mumtaz, and Jonathan González. "Towards sFlow and adaptive polling sampling for deep learning based DDoS detection in SDN."Future Generation Computer Systems 111 (2020): 763-779.

[28] Ravi, Nagarathna, and S. Mercy Shalinie. "Learning-driven detection and mitigation of DDoS attack in IoT via SDN-cloud architecture."IEEE Internet of Things Journal 7, no. 4 (2020): 3559-3570.

[29] Yong, Binbin, Wei Wei, Kuan-Ching Li, Jun Shen, Qingguo Zhou, Marcin Wozniak, Dawid Połap, and Robertas Damaševičius. "Ensemble machine learning approaches for webshell detection in Internet of things environments."Transactions on Emerging Telecommunications Technologies (2020): e4085.

[30] Kushwah, Gopal Singh, and Virender Ranga. "Optimized extreme learning machine for detecting DDoS attacks in cloud computing."Computers & Security 105 (2021): 102260.

[31] Damasevicius, Robertas, Algimantas Venckauskas, Sarunas Grigaliunas, Jevgenijus Toldinas, Nerijus Morkevicius, Tautvydas Aleliunas, and Paulius Smuikys. "LITNET-2020: An annotated real-world network flow dataset for network intrusion detection."Electronics 9, no. 5 (2020): 800.

[32] Erhan, Derya, and Emin Anarım. "Boğaziçi University distributed denial of service dataset."Data in brief 32 (2020).

[33] Rasool, Raihan Ur, Usman Ashraf, Khandakar Ahmed, Hua Wang, Wajid Rafique, and Zahid Anwar. "Cyberpulse: a machine learning based link flooding attack mitigation system for software defined networks."IEEE Access 7 (2019): 34885.

[34] Ambrosin, Moreno, Mauro Conti, Fabio De Gaspari, and Radha Poovendran. "LineSwitch: Tackling control plane saturation attacks in software-defined networking."IEEE/ACM Transactions on Networking 25, no. 2 (2016): 1206-1219.

[35] P. Illavarason and B. Kamachi Sundaram, "A Study of Intrusion Detection System using Machine Learning Classification Algorithm based on different feature selection approach," 2019 Third International conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), 2019, pp. 295-299.

[36] Tamilarasan, Ashwin, Srinivas Mukkamala, Andrew H. Sung, and Krishna Yendrapalli. "Feature ranking and selection for intrusion detection using artificial neural networks and statistical methods." In The 2006 IEEE International Joint Conference on Neural Network Proceedings, pp. 4754-4761. IEEE, 2006.

[37] Stein, Gary, Bing Chen, Annie S. Wu, and Kien A. Hua. "Decision tree classifier for network intrusion detection with GA-based feature selection." In Proceedings of the 43rd annual Southeast regional conference-Volume 2, pp. 136-141. 2005.

[38] Kuang, Fangjun, Weihong Xu, and Siyang Zhang. "A novel hybrid KPCA and SVM with GA model for intrusion detection." Applied Soft Computing 18 (2014): 178-184.

[39] Li, Yinhui, Jingbo Xia, Silan Zhang, Jiakai Yan, Xiaochuan Ai, and Kuobin Dai. "An efficient intrusion detection system based on support vector machines and gradually feature removal method." Expert systems with applications 39, no. 1 (2012): 424-430.

[40] Pascoal, Cláudia, M. Rosário Oliveira, António Pacheco, and Rui Valadas. "Theoretical evaluation of feature selection methods based on mutual information." Neurocomputing 226 (2017): 168-181.

[41] Koc, Levent, Thomas A. Mazzuchi, and Shahram Sarkani. "A network intrusion detection system based on a Hidden Naïve Bayes multiclass classifier." Expert Systems with Applications 39, no. 18 (2012): 13492-13500.

[42] Kennedy, James, and Russell Eberhart. "Particle swarm optimization." In Proceedings of ICNN'95-international conference on neural networks, vol. 4, pp. 1942-1948. IEEE, 1995.

[43] Ahuja, Nisha, Gaurav Singal, and Debajyoti Mukhopadhyay. "DDOS attack SDN dataset."Mendeley Data 1 (2020).

[44] Yang, L. ., & Daimin, G. . (2023). Children's Perspective on Digital Picure Book: A Brief Analysis . International Journal on Recent and Innovation Trends in Computing and Communication, 11(3), 166–177. https://doi.org/10.17762/ijritcc.v11i3.6336

[45] Mohammad Hassan, Machine Learning Techniques for Credit Scoring in Financial Institutions , Machine Learning Applications Conference Proceedings, Vol 3 2023.

[46] Raghavendra, S., Dhabliya, D., Mondal, D., Omarov, B., Sankaran, K.S., Dhablia, A., Chaudhury, S., Shabaz, M. Retracted: Development of intrusion detection system using machine learning for the analytics of Internet of Things enabled enterprises (2023) IET Communications, 17 (13), pp. 1619-1625.