IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Ingénierie des Systèmes d'Information
Vol. 30, No. 4, April, 2025, pp. 1015-1025
Journal homepage: http://iieta.org/journals/isi

Enhancing Intrusion Detection System for Software-Defined Networks Based on Machine Learning

Huda Abdulrazzaq Wahib*, Mahmood Zaki Abdullah, Ahmad Saeed Mohammad

Computer Engineering Department, College of Engineering, Mustansiriyah University, Baghdad 10052, Iraq

Corresponding Author Email: Huda_alshemary82@uomustansiriyah.edu.iq

https://doi.org/10.18280/isi.300418

Received: 20 January 2025
Revised: 18 April 2025
Accepted: 28 April 2025
Available online: 30 April 2025

Keywords:
intrusion detection system (IDS), software-defined network (SDN), Feature Selection (FS), Artificial Intelligence (AI), Machine Learning (ML)

ABSTRACT

The rapid expansion of the internet has increased network size and complexity, necessitating dynamic management strategies. Traditional networks struggle with scalability and monitoring, prompting the adoption of software-defined networks (SDNs), which offer programmability and flexibility by decoupling control and data planes. However, this centralized architecture has introduced new security challenges. Machine learning (ML)-based intrusion detection systems (IDSs) have emerged as effective solutions. This paper explores the integration of ML-powered IDS in SDN environments, evaluating classifiers like Decision Tree (DT) and Random Forest (RF) using metrics such as accuracy, precision, recall, and F1-score. Results show DT and RF achieve 99.99% classification accuracy, highlighting their potential for enhancing SDN security. The study emphasizes that combining feature selection with robust classifiers significantly improves threat detection, enabling targeted defense mechanisms and improving SDN resilience against cyberattacks.

1. INTRODUCTION

An intrusion detection system (IDS) is a technology that relies on monitoring network and packet traffic in real time to determine whether a packet present on the network is malicious or unwanted, while also identifying the general behavior of the network and any abnormal behavior [1]. Big data, strong computer resources, and network growth raise the requirement for the necessary tasks that must be done simultaneously in real time. As a result, IDS should monitor with caution, accuracy, and precision qualities that have not been present in prior techniques [2]. However, it is very remarkable how quickly the machine. The accuracy of learning algorithms has increased. Its introduction is based on the increasing demand for improved performance in many kinds of networks [3]. However, the software-defined network (SDN) implementation of the network-based IDS has opened up a new channel for its adoption due to the increasing diversity and magnitude of security threats in modern networks [4]. The exponential growth in network data volume and linked devices is accompanied by inherent security risks. The expansion of some technologies such as artificial intelligence (AI) tools, the Internet of Things (IoT), and quantum computing [5] leads to an increased amount of danger and attack, thereby complicating making network security challenging to implement and necessitating a new paradigm. The demand for sophisticated, flexible, and resilient security implementation has increased because of several assaults [6]. A novel technique that has been researched in recent years, SDN can be defined as a networking architecture that splits it into a control plane (decision-making) and a data plane (packet forwarding) in the network. As seen in Figure 1, the network management tasks are shifted from the various networking devices to a centralized controller, making it simpler to monitor and configure the network overall [7]. It's crucial to remember that SDN itself adds additional software components, including the controller and related software stack, even if it can aid in lowering the software complexity of network administration [8]. When implementing SDN, consideration should be given to the complexity of these components, which must be maintained and controlled. But all things considered, SDN's centralization, abstraction, programmability, and automation features help to streamline network administration and lower software complexity [9]. However, SDN is now open to assaults due to centralization, insecure controller communication, and improper authorization and authentication [10]. Data analytics has grown in popularity and usage across a wide range of application were used; this is due to the recent massive rise in computer power. The shortcomings of conventional IDS have drawn increased attention to the utilization of Machine Learning (ML) for advanced combat assaults and enhanced security [11]. The use of machine learning algorithms for malware detection [12], network intrusion detection [13], and botnet attack detection [14] has been investigated by the research community. By learning from experience, machine learning (ML) techniques might help the detecting system become more autonomous. Traditional networks have mostly used these methods to categorize harmful traffic and network assaults [15]. They have demonstrated significant promise in the categorization of network traffic and are frequently used for classification and prediction tasks [16]. The main advantage of using ML techniques in SDN is their capacity to impact network-wide security standards, as opposed to more localized policy implementation in traditional networks [17].

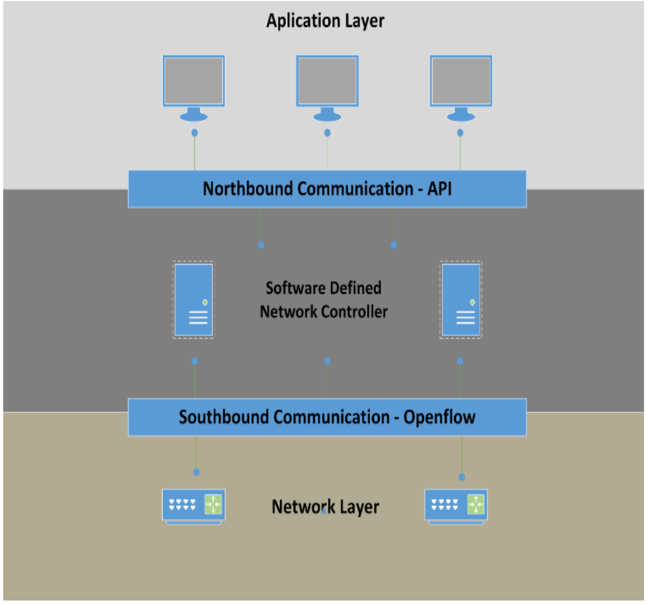

Aplication Layer
Northbound Communication - API
Software Defined
Network Controller
Southbound Communication - Openflow
Network Layer

Figure 1. SDN structure [7]

Three types of machine learning algorithms are frequently distinguished: supervised, unsupervised, and semi-supervised. In IDS, supervised machine learning techniques perform better than unsupervised and reinforced learning, claims [18]. Furthermore, the performance of machine learning techniques is discovered to be influenced by the types of data and learning approaches [19]. This work attempts to examine the potential of ML approaches in offering dependable security protection for SDN, considering previous research that uses a variety of ML algorithms and compares the accuracy and performance of various supervised algorithms. This research specifically examines five machine-learning techniques: Naive Bayes (NB), K-Nearest Neighbor (KNN), Support Vector Machine (SVM), Random Forest (RF), and Decision Tree (DT).

The primary contributions of this paper are as follows:

- The paper proposes an in-depth analysis, and comparative study of five machine learning algorithms, NB, KNN, SVM, RF, and DT, applied to intrusion detection within SDNs.
- It discusses how each algorithm varies in terms of accuracy, precision, recall, and F-score, outlining the most appropriate techniques that shall serve in enhancing the detection capabilities in the SDN environment.
- Because the study focuses on SDNs, it addresses challenges in dynamic and scalable network architectures, hence assuring that the proposed techniques will be optimized for threat detection.
- The contribution of this research study leads to the development of a robust and adaptable IDS framework that will use machine learning and can be integrated into SDN infrastructures to enhance network security.
- One significant contribution of using feature selection techniques in machine learning for IDS within SDN is the enhancement of detection accuracy. By identifying and retaining only the most relevant features from a potentially high-dimensional dataset, feature selection minimizes noise and reduces the risk of overfitting.

The remainder of the paper is structured as follows: The second section of the text is devoted to an examination of the extant literature and related research. The methodological description along with the implementation of machine learning models coupled with the used dataset has been described in Section 3. Section 4 presents the results and a comparison of the results, while Section 5 presents a discussion of the findings.

1.1 Research questions

RQ1. How is the creation of SND facilitated by flow-based network intrusion detection?

RQ2. How may machine learning be used to increase the accuracy of intrusion detection in SDN architectures with limited raw features?

RQ3. How can the system's throughput and latency performance be assessed?

RQ4. How can the feature selection techniques enhance the performance of ML algorithms?

2. RELATED WORKS

The recent research emphasis on cybersecurity and network defense has resulted in the extensive employment of the UNSW-NB15 dataset in the development of IDS that employ machine learning techniques. This study includes articles published between 2019 and 2024 that review various ML-based approaches and assess their effectiveness in enhancing intrusion detection.

1. KNN-Based Intrusion Detection Models

Nikhitha and Jabbar [20] proposed an IDS model using the KNN classifier. KNN, a supervised learning algorithm, demonstrated strong classification and accuracy performance, achieving a 99.96% accuracy on the ISCX dataset.

2. Comparative Evaluation of Multiple ML Models

Hasan et al. [21] compared multiple machine learning models, including Logistic Regression, SVM, DT, RF, and ANN for IoT security. DT, RF, and ANN achieved high test accuracy rates of 99.4%.

3. Integration of ML with SDN for High Detection Accuracy

Ibrahim and Bhaya [22] presented an SDN-enabled IDS architecture using GridSearch and SVM. The system, trained on UNSW-NB15 and NSL-KDD, achieved over 99.8% detection accuracy.

4. Hybrid IDS Using SVM and Neuro-Fuzzy Systems

Mehmood et al. [23] proposed a hybrid IDS approach using feature elimination and various SVMs for detection, with ANFIS for classification. Their model achieved 99.3% accuracy and low mean square error.

5. RF-Based Signature IDS

Zeleke et al. [24] developed a centralized signature-based IDS using RF on the CICIDS2017 dataset, achieving 99.968% accuracy with only 12 features.

6. Tree-Based ML Techniques for SDN Security

Alzahrani and Alenazi [25] applied DT, RF, and XGBoost to detect malicious activities in SDN using the NSL-KDD dataset. Their multi-class classifier achieved 95.95% accuracy using only 5 features.

7. Feature Selection Using Genetic Algorithm (GA)

Saba et al. [26] enhanced IDS performance by using a Genetic Algorithm for feature selection, then applied DT, Ensemble Classifier, and SVM on NSL-KDD, achieving 99.8% accuracy.

8. Selective Logging and IP Traceback in IDS

Hadem et al. [27] proposed an SVM-based IDS integrated with selective logging for IP traceback. Their system achieved 95.98% accuracy using the full NSL-KDD dataset and 87.74% on selected subfeatures.

9. DoS and Port Scanning Detection with ANOVA and ML

AlMasri et al. [28] used ANOVA for feature selection and applied several ML models to detect DoS and Probe attacks. The NB model achieved 86.9% and 93.5% accuracy, respectively.

10. Hybrid Feature Selection with LightGBM

Logeswari et al. [29] introduced HFS-LGBM for SDN IDS. Combining CFS and RF-RFE for feature selection, and LightGBM for classification, the model achieved 98.72% accuracy.

11. ML-Based IDS for IoT-SDN Integration

Alshammari and Alserhani [30] developed an ML-based IDS using SVM, KNN, LR, RF, and DT on the ToN-IoT dataset. RF outperformed others with a 99% accuracy rate.

12. Preprocessing and Feature Scaling in ML IDS

Raju and Suma [31] emphasized preprocessing and evaluation. Their DT classifier achieved 99.17% accuracy, followed by RF and KNN with 99.11% and 98.22%.

13. ML for DDoS Detection in SDN-IoT

Bhayo et al. [32] applied NB, DT, and SVM in SDN-WISE IoT for DDoS detection. Accuracy rates were 97.4%, 96.1%, and 98.1%, respectively, with significant memory optimization.

14. Intrusion Detection in IoT Using Dual Datasets

Kumari et al. [33] compared ML classifiers on two IoT datasets, achieving 99.11% and 99.99% accuracy for the IoT Network intrusion and IoT_23 datasets, respectively.

15. Multi-Attack Detection in SDN-IoT Networks

Ferrão et al. [34] proposed MAIDS using XGBoost and RF on NSL-KDD and CICIDS2017 datasets, achieving average accuracies of 99.88% and 99.89%.

16. Improving IDS with Feature Balancing and Optimization

Hacilar et al. [35] examined feature imbalance, binary classification, and optimization techniques such as SMOTE, ROS, and ADASYN with XGBoost. Impressive accuracy, detection, and low false alarm rates were achieved across UNSW-NB15, AWID, and InSDN datasets.

3. METHODOLOGY

This methodology, as shown in Figure 2 starts on the UNSW-NB15 dataset, which is one of the modern benchmark datasets in recent times for benchmarking IDS in cybersecurity research, and training the model with the dataset has comprehensiveness with a wide range of attacks with scenarios and relevance to modern network traffic patterns compared with other datasets.

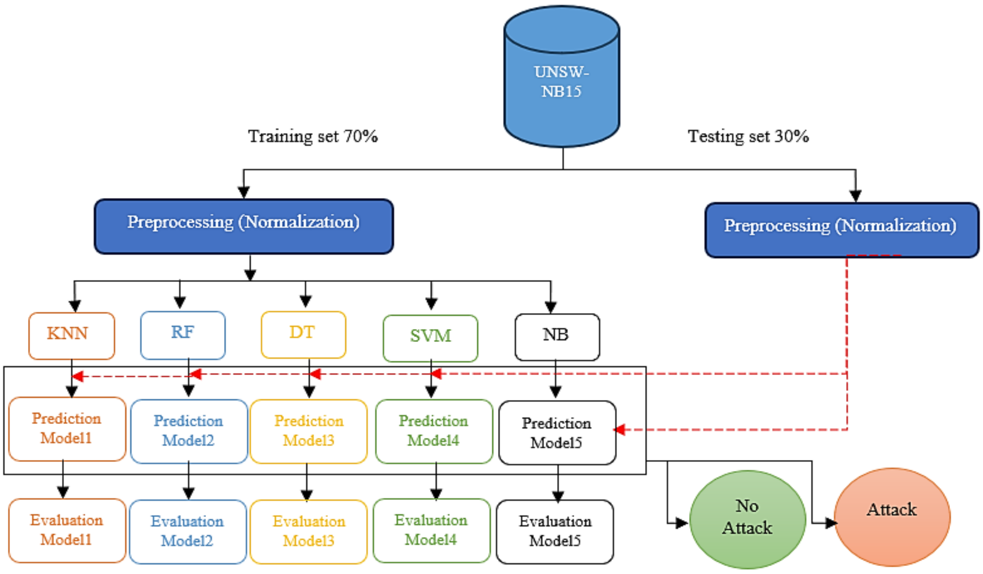


Figure 2. The proposed IDS ML-based methodologies architecture

The preprocessing starts with data division; dividing the dataset into training and testing, usually in 70-30 proportions helps analyze the generalized performance of the machine learning model, while the data pre-processing step involves preparations of the dataset, making it ready for analysis. It includes normalization to enhance the performance of distance-based models like KNN. This step is, therefore, crucial for noise reduction and ensuring the integrity of the data for training the model. Classification is done after preprocessing with five machine learning techniques usually supervised: NB, KNN, SVM, RF, and DT. Each model will be trained based on pre-processed data to identify and classify network intrusions. In this regard, each classifier's performances are measured based on accuracy, precision, recall, and F1-score so that no comparison is left concerning the efficacies brought in by these algorithms in finding cybersecurity threats using the UNSW-NB15 dataset.

3.1 UNSW-NB15 dataset description

The UNSW-NB15 dataset had evolved by the Australian Center for Cyber Security (ACCS) in partnership with other academics from across the world. The IXIA Perfect Storm program was utilized to generate a diverse array of both standard and atypical contemporary network traffic. The IXIA tool proactively gathers and compiles information systems security vulnerabilities and exposures that are known to the public. It serves as a useful resource for learning about contemporary public attacks.

The UNSW-NB15 dataset embeds many contemporary low-key assaults in an attempt to replicate contemporary network settings. The dataset's 10 different traffic types are normal, fuzzing, analysis, backdoor, DOS, exploits, generic, reconnaissance, and worms [36].

3.2 Data preprocessing

Data can contain different types of data, whether images, audio files, video clips, structured and unstructured tables, etc. The free text, video, or image must be converted into 1s and 0s since a machine cannot comprehend it in its original form. Therefore, the utilization of raw data directly input into a machine learning model is not a viable approach to achieve the desired results [37]. The first stage of machine learning is called data preprocessing, during which the input is changed or encoded so that the computer can process or read it more rapidly. Stated differently, it might also mean that the model method can quickly assess the data's characteristics. The most significant and influencing factor in a supervised machine learning algorithm's ability to generalize is data pre-processing [38]. About the input space dimension, the amount of training data increases exponentially. Pre-processing is an important process for model construction since the model can account for a range of 50% to 80% of the whole classification process, according to estimates. Furthermore, enhancing the quality of the data is also necessary for amended performance of the model [39].

Data Splitting: is an important technique used for removing or minimizing bias in training data that have used in machine learning models. The researchers invariably execute some procedure to circumvent the production of overfitting machine learning algorithms that could demonstrate substandard performance on authentic test data. Data scientists and analysts typically divide the datasets into several distinct subsets, which they then use to train various parameters [40]. One of the very common methodologies in machine learning, when it comes to splitting data for model performance assessment, is this k-fold cross-validation approach. Unlike splitting the data into a fixed training and testing set, this approach splits the data into "K" roughly equal-sized subsets (or folds) [41]. The remaining "K-1" folds are used as the training set, and the model is trained and validated "K" times, using a different fold as the validation set each time. This gives the performance evaluation greater robustness by guaranteeing that every data point is used for both training and validation. K-fold cross-validation reduces variance associated with a single estimate, hence giving a better estimate of how well the model will generalize to unseen data. It also makes the outcome of the evaluation less sensitive to the way the data has been initially split. Common choices are 5-fold or 10-fold cross-validation, but the number of folds can vary depending on the size of the dataset and the computational resources. This technique is particularly useful when working with limited data, as it maximizes the use of available information [42].

Data Normalization: Certain features in the "UNSW-NB15" dataset may have lower values than others, while other features may have comparatively higher values. Furthermore, because the classification algorithm may be biased in favor of characteristics with larger values, out-of-range values may yield inaccurate findings. To prevent the outweighing issue, which would prefer features with higher values over those with lower ones, data standardization is crucial. There are several normalization methods, including traditional scalar approaches and min-max. As stated in Eq. (1), the min-max approach is used in this work to scale the feature values between zero and one.

$$Z = \frac{x-\min(x)}{\max(x)-\min(x)} \tag{1}$$

where, X stands for the feature value and Z for the normalized value. The maximum and minimum feature amounts are denoted by $\max(x)$ and $\min(x)$ respectively [43]. Normalization in this work involves scaling certain features of the dataset within a common range, usually between 0 and 1 or -1 and 1. And this method was chosen because it preserves the distribution of the data and ensures that all features have the same range, making it easier for the model to learn from the data. This helps in reducing different magnitudes and units across features, making it impossible for one feature to dominate others in the learning process. Convergence happens more efficiently and faster; thus, with normalized data, algorithms make better models and yield accurate results. Besides, it stabilizes gradient descent and other methods of optimization. In general, normalization improves the robustness of machine learning models.

3.3 Features selection stage

The field of machine learning relies primarily on feature selection and extraction from the database that have been used. Furthermore, it is worth noting that incorporating a large number of features from the database can lead to increased model complexity and increased training time. Therefore, selecting relevant features is essential for the effectiveness of machine learning approaches. Therefore, an effective feature selection strategy must be followed carefully to address this problem [44]. There is an important thing to get successful implementation of machine learning algorithms: a judicious selection of relevant features selection is of paramount importance. It has been posited that the omission of certain elements, which are of the utmost importance, may result in a consequential deficiency in the accuracy of the results. A multitude of issues must be addressed. A variety of feature selection techniques are applied in the field of machine learning. These consist of recursive feature elimination, chi-square, and backward feature selection [45]. These methods are applied according to correlation, dimensionality, and datasets. In our model, we employed a variety of methods, including recursive feature elimination (RFE), backward elimination, and forward feature selection, to identify the most salient features. These techniques were selected because they yielded higher accuracies compared to other methods.

(1) Recursive Feature Elimination (RFE) "1st technique"

RFE is a wrapper feature selection method that recursively eliminates the least important features based on a predefined model performance metric. In each iteration, the model is retrained with the remaining features until the optimum subset of features is achieved. This method is particularly efficient in feature dimensionality reduction while preserving important features [46]. RFE improves model interpretability and reduces computational complexity by focusing on the most impactful attributes. This has extensive use both in linear and nonlinear models.

(2) Backward Elimination "2nd technique"

Backward selection starts by considering all the available features and iteratively removes the least relevant among them [47]. After the model retraining in each step, one can check the removal effect of any particular feature. This process is continued until an optimal subset of features is obtained that balances performance with simplicity. Backward selection is computational for large data but gives more elaborative insight into the relevance of the features. This reduces overfitting by eliminating either irrelevant or redundant features, hence increasing the efficiency of the model.

(3) Forward Feature Selection “3rd technique”

Forward selection is the process where an empty set of features is considered at the start, and one feature is progressively added based on its contribution to model improvement. At each step, the feature whose addition increases any performance metric, including accuracy and AUC, is included in the set of selected features. This is computationally much easier than backward elimination and is therefore suitable for high-dimensional data. Forward selection guarantees that only the most relevant features are used, which enhances model accuracy and reduces the possibility of overfitting. It is useful when the computational resources are limited [48].

3.4 Machine learning-based intrusion detection

An AI technology and subset of artificial intelligent that called machine learning (ML) investigates different approaches to learning from and forecasting data. It uses features that correspond to an object’s features to find and learn data patterns. Supervised learning and unsupervised learning are the two main subcategories of ML [49]. Supervised learning requires labeled data with relevant information, but classification is a common problem. Manual tagging is costly and time-consuming, making it difficult to obtain enough annotated data. Unsupervised learning is simpler to implement but can extract relevant feature information from unlabeled data. Despite this, supervised learning techniques often outperform unsupervised learning in detection [50]. The intrusion detection with five machine learning techniques involves the training of NB, KNN, SVM, RF, and DT algorithms on how to recognize malicious activities within the traffic as shown in Figure 3.

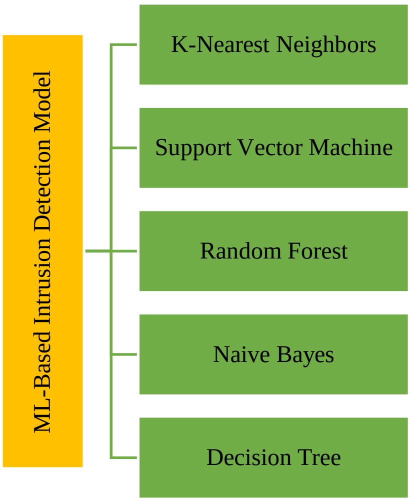


Figure 3. ML-based intrusion detection model

Each method processes the input features extracted from the dataset to classify the traffic as normal or suspicious. NB is based on probabilistic inference using feature distributions, while KNN decides on the class of a data point based on the analysis of its closest neighbors in the feature space. SVM develops hyperplanes to separate classes, trying to provide a maximal margin between the instances that belong to normal and attack classes. RF combines several DT for robust classification by aggregating outputs of trees, while DT develops a flowchart-like model composition based on decision rules derived from the dataset itself. By using these algorithms, it enables the detection of both known and newly emerging threats through learning complex patterns within network flows.

KNN: The KNN method uses the nearest means to divide a dataset into sets that are either malicious or non-malicious. It takes prior training to make accurate predictions. To determine the weight of each attribute, we first determined the probability of each of the K neighbors to be classified using the KNN. The dataset, or new instance (X), is provided, and the Euclidean distance is calculated in terms of locating the associated class variable as indicated by equation in order to determine the output variable.

$$Euclidean_{i,j} = \sqrt{\sum_{k=1}^{n}(x_{ik} - y_{ik})} \tag{2}$$

where, (x_i) represents a new instance, (y_j) represents an old instance [51].

SVM: it is a machine learning tool that has been used for different models to get classification and regression. An SVM operates by using critical data points known as support vectors to determine the optimal hyperplane to optimize the margin between classes. It can handle both linearly and non-linearly separable data thanks to the kernel approach, which comprises the linear, polynomial, and radial basis functions and sigmoid kernels. It works especially well for classification issues that are binary or even multi-class. The equation for a SVM classifier can be represented as Eq. (3):

$$f(x) = W^T x + b \tag{3}$$

where, $f(x)$ is the decision function indicating the label of the class of the input, w defines the orientation of the hyperplane because it is the weight vector, x represents an input feature vector, and b is the bias term, shifting the hyperplane away from the origin [52, 53].

RF: is a technique for ensemble learning that uses numerous decision trees to generate predictions. Also, it can demonstrate how to enhance the classification and regression tasks by training multiple decision trees on various subsamples of the data set. After that, the predictions of these distinct trees are then integrated to enhance accuracy and prevent overfitting. The equation of RF is based on an aggregation of many decision trees’ predictions as Eq. (4). The final prediction $\check{y}$ for a given input x can be expressed as:

$$\check{y} = \frac{1}{N}\sum_{i=1}^{N} T_i x \tag{4}$$

The forest has a total of N decision trees, while $T_i x$ indicates the forecast for the input x that the i-th decision tree made [54].

NB: is predicated on the idea that features are naïvely and independently unconnected to one another. Based on observed feature values, it computes the posterior probability of classes using the Bayes theorem. Gaussian, Multinomial, and

Bernoulli Naive Bayes algorithms are available, depending on the kind of distribution that is assumed for the features. Naive Bayes has gained popularity for a variety of applications due to its ease of use and effectiveness in training and prediction tasks [55]. The equation for a Nb is shown in Eq. (5).

$$p(c|x) = \frac{p(X|C)p(c)}{p(x)} \tag{5}$$

$p(c)$ is the posterior probability of class (c, target) (x, characteristics) given the predictor, $p(x)$ is the prior probability of the predictor, $p(c)$ is the historical probability of a specific class, and $p(c|x)$ is the probability of a predictor in a specific class [56].

DT: is an approach to prediction tasks that divide the predictor space into easily analysed segments. Regression and classification of real-world scenarios are two more uses for it. Additionally, the machine learning algorithm to decision tree structures makes decisions based on the feature values. In contrast, the root of the tree is located at the very top. In addition to gradually evolving the decision tree, the branches are constructed using objective rules derived from the features of the dataset [57].

To create a decision tree, follow these steps [58]:

1. Divide the entire dataset into training and test sets.
2. Use the training set as an input to the tree's root.
3. Use information theory to find the root, as demonstrated in (2).
4. Follow the prone procedure.
5. Repeat steps 1 through 4 until all nodes have become leaf nodes.

As show in the Eqs. (6) and (7) the entropy equation.

$$Entropy\ E(H) = \sum_{k=1}^{d} -P_j \log_2 P_j \tag{6}$$

$$information\ gain = Entropy\ (before) - \sum_{k=1}^{d} Entropy(j\backslash k, later) \tag{7}$$

To achieve this, the algorithm progressively separates the database into several subgroups based on entropy or information amount, and it continues to do so until a halt condition is satisfied [59].

3.5 Model evaluation

Several evaluation methodologies are chosen to gauge the effectiveness of the employed ML techniques to give a thorough explanation of the outcomes of ML-based IDS. Specifically, the confusion matrix, displayed in Table 1, is used to evaluate the performance of the detection rate using precision, recall, F-measure, and accuracy metrics, as detailed below [60, 61].

Table 1. Typical structure of confusion matrix [61]

	Predicted as 'Normal'	Predicted as 'Attack'
Actual Normal Class	TP	FP
Actual Attack Class	FN	TN

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

Accuracy: According to Eq. (8), accuracy is defined as the ratio of the model's accurate data to the whole data, as shown below.

$$\text{Accuracy} = \frac{\text{TP+TN}}{\text{TP+TN+FP+FN}} \tag{8}$$

Precision: The percentage of actual cases among all positive examples found by the model (TP) is what defines precision, which is described in Eq. (9). Stated differently, it indicates the proportion of classified assault incidents that are truly classed as an attack.

$$\text{Precision} = \frac{\text{TP}}{\text{TP+FP}} \tag{9}$$

Recall: is defined as the proportion of attack traffic instances overall, as provided by Eq. (10) to the number of attacks that the model classified as attacks. It shows the proportion of real cases that were disclosed to all true instances.

$$\text{Recall} = \frac{\text{TP}}{\text{TP+FN}} \tag{10}$$

F1-Score: This metric provides a harmonic average measurement of an estimator's sensitivity and precision and is defined in Eq. (11).

$$\text{F} - \text{score} = 2 * \frac{\text{precision}*\text{recall}}{\text{precision+recall}} \tag{11}$$

4. RESULTS EVALUATION

This section evaluates the suggested model using the chosen machine-learning techniques.

4.1 Results of ML techniques without FS

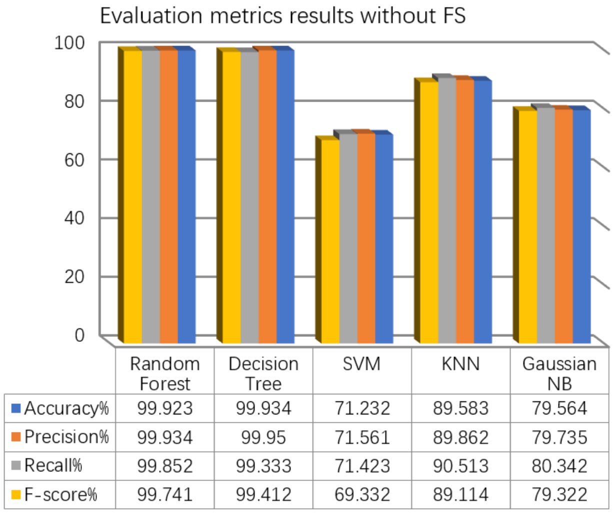
Table 2 and Figure 4 depict a model performance comparison, without feature selection. It reflects that all scores of the RF stand higher than any other model, meaning that it's the best performing one; SVM stands a little after RF while KNN and DT performed averagely and Gaussian NB poorly. This suggests that RF and SVM are more robust, whatever the data condition without feature selection, whereas Gaussian NB fails to compete for the best results.

4.2 Results of ML techniques with RFE-20 FS

As shown in Figure 5 Comparatively, RF, DT, SVM, and KNN had approximately equal performances for F1-score, Accuracy, Recall, and Precision, showing how robust and reliable they can be. On the other hand, NB showed the lowest performances regarding all metrics, suggesting that it is unsuitable in this context. Regarding time efficiency, NB was the fastest, while SVM was the most computationally complex and thus took the longest. RF and KNN show a perfect balance between high performance and time efficiency, thus making them the best choices; SVM presents the best accuracy but requires more time to process. Generally, RF is the most balanced and reliable model. As shown in Figure 6 models like DT and NB are very time-efficient, with very low processing time. RF and KNN fall in the middle in terms of efficiency. Although SVM may be superior in accuracy or other metrics, this result shows its high computational cost; hence, DT and NB are more suitable for applications that require speed.

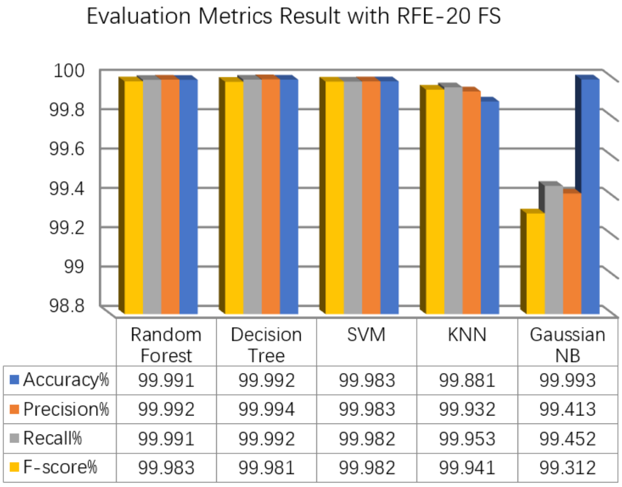
Table 2. Results of ML without FS

	Accuracy%	Precision%	Recall%	F-score%
NB	79.564	79.735	80.342	79.322
KNN	89.583	89.862	90.513	89.114
SVM	71.232	71.561	71.423	69.332
DT	99.934	99.950	99.333	99.412
RF	99.923	99.934	99.852	99.741



	Random Forest	Decision Tree	SVM	KNN	Gaussian NB
🟦 Accuracy%	99.923	99.934	71.232	89.583	79.564
🟧 Precision%	99.934	99.95	71.561	89.862	79.735
⬜ Recall%	99.852	99.333	71.423	90.513	80.342
🟨 F-score%	99.741	99.412	69.332	89.114	79.322

Figure 4. Evaluation metrics results without FS



	Random Forest	Decision Tree	SVM	KNN	Gaussian NB
🟦 Accuracy%	99.991	99.992	99.983	99.881	99.993
🟧 Precision%	99.992	99.994	99.983	99.932	99.413
⬜ Recall%	99.991	99.992	99.982	99.953	99.452
🟨 F-score%	99.983	99.981	99.982	99.941	99.312

Figure 5. Evaluation metrics result with RFE-20 FS

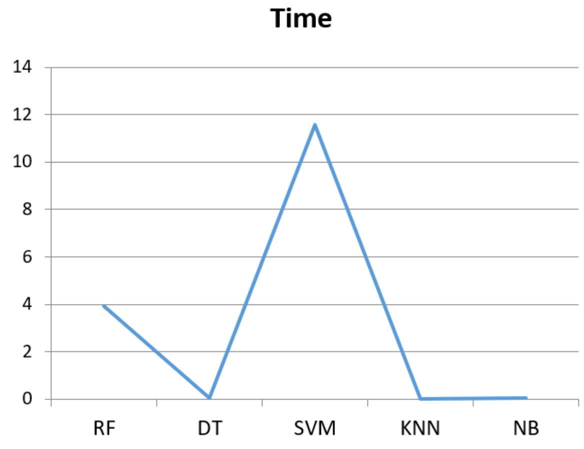
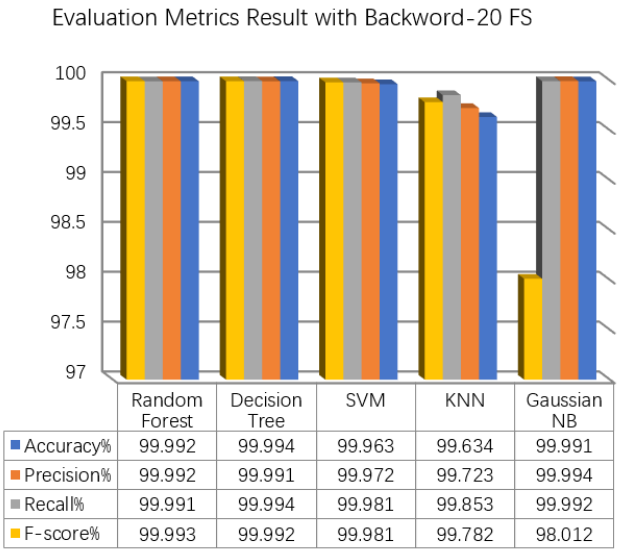


Figure 6. Time result with RFE-20 FS

4.3 Results of ML techniques with backword-20 FS

As shown in Figure 7, RF, DT, SVM, and KNN are very consistent in performing well on Accuracy, scoring almost full marks, which means NB is far from the rest and thus is weaker in predictive capability. Precision: RF, DT, SVM, and KNN again show high values, proving to be reliable in the low rate of false positives, while NB has a significant drop in precision. Recall is high for RF, DT, SVM, and KNN, meaning that they can detect the relevant cases very well, but NB, with its lower recall, it tends to miss positive instances. F1 scores show that RF, DT, SVM, and KNN have a very balanced performance due to the high values of precision and recall; in the case of NB, the low F1 score outlines an inconsistent performance. Time-wise, NB is the fastest, thus very computationally efficient and light; also, KNN processes rather fast. RF and DT are moderately fast and balance time with performance, while for SVM, computational complexity is highest: it took the longest running time among all. The best balance among all metrics due to its high accuracy, precision, and recall, though it is reasonably time-efficient, is RF. Other competitive alternatives are DT and KNN, which have a slight increase in computation. The SVM has high precision and recall; however, its extended runtime decreases its efficiency.



	Random Forest	Decision Tree	SVM	KNN	Gaussian NB
🟦 Accuracy%	99.992	99.994	99.963	99.634	99.991
🟧 Precision%	99.992	99.991	99.972	99.723	99.994
⬜ Recall%	99.991	99.994	99.981	99.853	99.992
🟨 F-score%	99.993	99.992	99.981	99.782	98.012

Figure 7. Evaluation metrics results with backword-20 FS

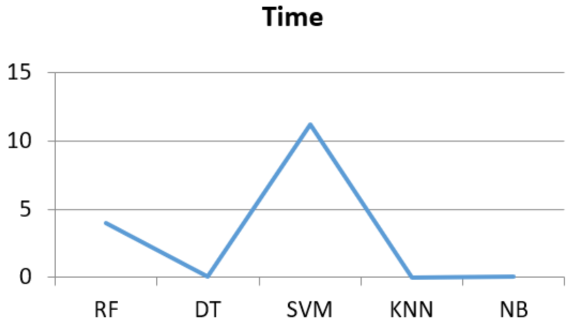


Figure 8. Time result with backword-20 FS

On the contrary, NB is suitable for when the priority is on speed but sacrifices predictive quality and is hence the weakest performer among the tested models. This analysis confirms that RF is the most robust model operating at both high performance and efficient computation. In Figure 8, DT and NB present the lowest time, which is very efficient. RF and KNN also have very low time, a little higher than those of DT and NB. Indeed, this reflects that while powerful, SVM is not very suitable for applications where computational efficiency is crucial, while DT, NB, and other light models are more suitable for such applications.

4.4 Results of ML techniques with forword-20 FS

Figures 9 and 10 illustrate the performance metrics and time of several machine learning models, including KNN, RF, DT, SVM, and NB, which are guided by forward feature selection.

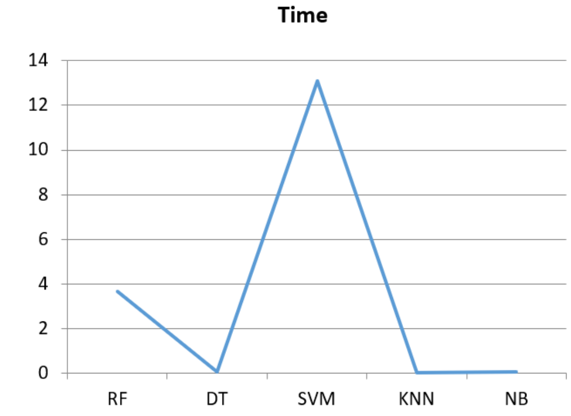
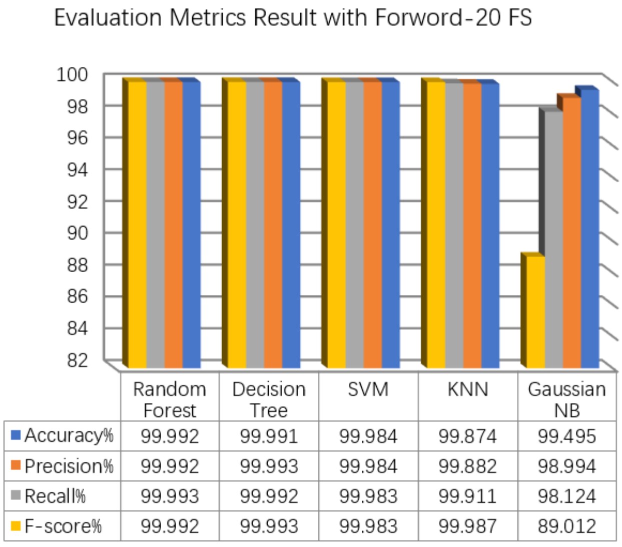


Figure 9. Time result with forword-20 FS



	Random Forest	Decision Tree	SVM	KNN	Gaussian NB
🟦 Accuracy%	99.992	99.991	99.984	99.874	99.495
🟧 Precision%	99.992	99.993	99.984	99.882	98.994
⬜ Recall%	99.993	99.992	99.983	99.911	98.124
🟨 F-score%	99.992	99.993	99.983	99.987	89.012

Figure 10. F-score result with forword-20 FS

In the accuracy chart, RF and DT had perfect scores of 1.0, with SVM and KNN each scoring 0.99, whereas NB trailed with a score of 0.97, showing much weaker predictive capabilities. It gets precision where RF, DT, SVM, and KNN get the perfect score of 1.0, making them reliable models with fewer numbers of false positives, whereas NB falls with 0.98. It is the same as that of accuracy; all these, RF, DT, SVM, and KNN, stand at 1.0 score, whereas NB stands as poorest with a value of 0.98, once again the worst result of missed relevant cases in Recall. The F1-score is a balancing measure between precision and recall; in its case, RF, DT, SVM, and KNN performed perfectly, with NB somewhat lagging. In terms of Time, the fastest will be NB-0.04 sec, followed by KNN-0.09 sec and DT-0.06 sec, while RF performed slightly longer, 3.67 seconds, and SVM had the largest running time because of its computational complexity at 13.09 seconds. Overall, RF and DT are the best-performing models, as all metrics are perfect and have relatively reasonable computation times. Strong performances are also realized by KNN and SVM, though the latter is costlier in terms of time. Meanwhile, NB was efficient in computation time but always had the worst values for all metrics, hence being unreliable for this task.

4.5 Comparison results with related studies

Table 3 compares the accuracy of several studies and models elucidates the effectiveness of machine learning algorithms in intrusion detection for SDN networks. Most of the referenced models show very impressive performances, with accuracy rates mostly falling within the range of 95% to almost 99.99%. Models in papers [20, 24, 26, 34] achieve accuracy above 99%, some going as high as 99.99%, which testifies to a good capability of threats being detected. Among these, the KNN model performs best at 98.7%, comparable to most of the existing related works. SVM fares equally well at 99.9%, which is very close to that reported by the highest-scoring referenced studies. RF and DT models achieve a perfect accuracy of 99.99%, reflecting their extraordinary capability to classify network traffic without error within the context of SDN. While NB reaches a somewhat lower 97.2%, it is competitive. These minor differences among the models probably reflect varying strengths in modeling complex traffic patterns that may depend on the features of the dataset or algorithm structure. In an overview, these results tend to bring out the potential of advanced machine learning techniques, especially RF and DT for realizing highly accurate intrusion detection, hence being among the top options in robust security frameworks of SDNs.

Table 3. Accuracy comparison with related studies

Ref.	Accuracy%
[20]	99.96
[21]	99.4
[22]	99.8
[23]	99.3
[24]	99.968
[25]	95.95
[26]	99.8
[27]	95.98
[28]	86.9
[29]	98.72
[30]	99%
[31]	99.17
[32]	97.4
[33]	99.11
[34]	99.88
[35]	99.9
The proposed model with KNN without FS	89.583
The proposed model with SVM without FS	71.232
The proposed model with RF without FS	99.923
The proposed model with NB without FS	79.564
The proposed model with DT without FS	99.934
The proposed model with KNN with RFE-20 FS	99.881
The proposed model with SVM with RFE-20 FS	99.983
The proposed model with RF with RFE-20 FS	99.991
The proposed model with NB with RFE-20 FS	99.993
The proposed model with DT with RFE-20 FS	99.992
The proposed model with KNN with Backward-20 FS	99.634

The proposed model with SVM with Backward-20 FS	99.963
The proposed model with RF with Backward-20 FS	99.992
The proposed model with NB with Backward-20 FS	99.991
The proposed model with DT with Backward-20 FS	99.994
The proposed model with KNN with Forword-20 FS	99.874
The proposed model with SVM with Forword-20 FS	99.984
The proposed model with RF with Forword-20 FS	99.992
The proposed model with NB with Forword-20 FS	99.495
The proposed model with DT with Forword-20 FS	99.991

5. CONCLUSIONS

This study presents the efficiency of NB, KNN, SVM, RF, and DT models by applying them to the problem of network intrusion detection based on the UNSW-NB15 dataset. Data preprocessing normalized the data to improve the performance of distance-based models, such as KNN, by reducing noise to maintain the integrity of data. Then, three feature selection techniques were used to enhance the performance of ML classifiers. Later, each of these classifiers was trained to detect various types of network intrusions. Performance metrics used here are accuracy, precision, recall, and F1-score, which detail the comparative performance of the models.

As illustrated in the previous section, the summary for the best results of feature selection that appear and are highlighted for two mechanisms is that RF and DT offer the best balance of high performance across different metrics such as accuracy, precision, recall, and F1-score, while maintaining reasonable computation times that achieve 3.67 seconds for RF and 0.06 seconds for DT. On the other hand, the KNN algorithm performs well but is slightly slower. In contrast, the fastest algorithm was NB, which reached 0.04 seconds but showed the weakest predictive capabilities overall. While SVM achieved high accuracy, it had the slowest performance, with 13.09 seconds, due to its computational complexity making it less suitable for time-sensitive applications.

Notably, some models in certain cases had perfect results with 100% metrics of accuracy, precision, and recall, showcasing a somewhat misleadingly perfect classifier with no misclassifications. The good performance has demonstrated the capability of some machine learning algorithms to perform intrusion detection with very high accuracy. The results therefore give an overview of the strengths of each model and guide on how best to choose the most effective techniques for cybersecurity applications based on the UNSW-NB15 dataset.

For Limitation and future work our models were evaluated by using a single dataset (UNSW-NB15), which may not apply to represent the variety and complexity of real-world network traffic and evolving cyber threats.

For future work, we must apply our model to have extended the evaluation to multiple benchmark datasets to improve reliability and robustness. Furthermore, using different deep learning approaches, such as CNNs or LSTMs, and hybrid models combining ML with deep learning to improve IDS. Finally, concatenate between real-time IDS simulation environments to assess latency and resource efficiency.

REFERENCES

[1] Das, S., Gangwani, P., Upadhyay, H. (2023). Integration of machine learning with cybersecurity: Applications and challenges. Artificial Intelligence in Cyber Security: Theories and Applications, 67-81. https://doi.org/10.1007/978-3-031-28581-3_7

[2] Abiodun, O.I., Jantan, A., Omolara, A.E., Dada, K.V., Umar, A.M., Linus, O.U., Arshad, H., Kazaure, A.A., Gana, U., Kiru, M.U. (2019). Comprehensive review of artificial neural network applications to pattern recognition. IEEE Access, 7: 158820-158846. https://doi.org/10.1109/ACCESS.2019.2945545

[3] Kattenborn, T., Leitloff, J., Schiefer, F., Hinz, S. (2021). Review on Convolutional Neural Networks (CNN) in vegetation remote sensing. ISPRS Journal of Photogrammetry and Remote Sensing, 173: 24-49. https://doi.org/10.1016/j.isprsjprs.2020.12.010

[4] Sherstinsky, A. (2020). Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. Physica D: Nonlinear Phenomena, 404: 132306. https://doi.org/10.1016/j.physd.2019.132306

[5] Pinaya, W.H.L., Vieira, S., Garcia-Dias, R., Mechelli, A. (2020). Autoencoders. In Machine Learning, pp. 193-208. Academic Press. https://doi.org/10.1016/B978-0-12-815739-8.00011-0

[6] Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C., Sun, M. (2020). Graph neural networks: A review of methods and applications. AI Open, 1: 57-81. https://doi.org/10.1016/j.aiopen.2021.01.001

[7] Brendel, W., Rauber, J., Bethge, M. (2017). Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. arXiv preprint arXiv:1712.04248. https://doi.org/10.48550/arXiv.1712.04248

[8] Anthi, E., Williams, L., Javed, A., Burnap, P. (2021). Hardening machine learning denial of service (DoS) defences against adversarial attacks in IoT smart home networks. Computers & Security, 108: 102352. https://doi.org/10.1016/j.cose.2021.102352

[9] Mohammadi, M., Rashid, T.A., Karim, S.H.T., Aldalwie, A.H.M., Tho, Q.T., Bidaki, M., Rahmani, A.M., Hosseinzadeh, M. (2021). A comprehensive survey and taxonomy of the SVM-based intrusion detection systems. Journal of Network and Computer Applications, 178: 102983. https://doi.org/10.1016/j.jnca.2021.102983

[10] Ying, X. (2019). An overview of overfitting and its solutions. Journal of Physics: Conference Series, 1168: 022022. https://doi.org/10.1088/1742-6596/1168/2/022022

[11] Kamel, H., Abdullah, M.Z. (2022). Distributed denial of service attacks detection for software defined networks based on evolutionary decision tree model. Bulletin of Electrical Engineering and Informatics, 11(4): 2322-2330. https://doi.org/10.11591/eei.v11i4.3835

[12] Vaidhya, M. (2022). Sentiment analysis of different E-Commerce platform reviews using machine learning algorithm. Ph.D. Thesis, IOE Pulchowk Campus, Patan, Nepal.

[13] Nagelli, A., Saleena, B. (2023). A comparative review of sentimental analysis using machine learning and deep

learning approaches. Journal of Information & Knowledge Management, 22(3): 2350003. https://doi.org/10.1142/S021964922350003X

[14] Kamel, H., Abdullah, M.Z. (2022). A new approach of extremely randomized trees for attacks detection in software defined network. Indonesian Journal of Electrical Engineering and Computer Science, 28(3): 1613-1620. https://doi.org/10.11591/ijeecs.v28.i3.pp1613-1620

[15] Tahseen, T., Kabir, M.M.J. (2022). A comparative study of deep learning neural networks in sentiment classification from texts. Machine Learning and Autonomous Systems: Proceedings of ICMLAS 2021, pp. 289-305. Singapore: Springer Nature Singapore. https://doi.org/10.1007/978-981-16-7996-4_20

[16] Alzahrani, M.E., Aldhyani, T.H., Alsubari, S.N., Althobaiti, M.M., Fahad, A. (2022). Developing an intelligent system with deep learning algorithms for sentiment analysis of E-commerce product reviews. Computational Intelligence and Neuroscience, 2022(1): 3840071. https://doi.org/10.1155/2022/3840071

[17] Jain, S., Roy, P.K. (2024). E-commerce review sentiment score prediction considering misspelled words: A deep learning approach. Electronic Commerce Research, 24(3): 1737-1761. https://doi.org/10.1007/s10660-022-09582-4

[18] Alsubari, S.N., Deshmukh, S.N., Aldhyani, T.H., Al Nefaie, A.H., Alrasheedi, M. (2023). Rule-based classifiers for identifying fake reviews in e-commerce: A deep learning system. In Fuzzy, Rough and Intuitionistic Fuzzy Set Approaches for Data Handling: Theory and Applications, Singapore: Springer Nature Singapore, pp. 257-276. https://doi.org/10.1007/978-981-19-8566-9_14

[19] Da'u, A., Salim, N. (2020). Recommendation system based on deep learning methods: A systematic review and new directions. Artificial Intelligence Review, 53(4): 2709-2748. https://doi.org/10.1007/s10462-019-09744-1

[20] Nikhitha, M., Jabbar, M.A. (2019). K nearest neighbor based model for intrusion detection system. International Journal of Recent Technology and Engineering (IJRTE), 8(2): 2258-2262.

[21] Hasan, M., Islam, M.M., Zarif, M.I.I., Hashem, M.M.A. (2019). Attack and anomaly detection in IoT sensors in IoT sites using machine learning approaches. Internet of Things, 7: 100059. https://doi.org/10.1016/j.iot.2019.100059

[22] Ibrahim, O.J., Bhaya, W.S. (2021). Intrusion detection system for cloud based software-defined networks. Journal of Physics: Conference Series, 1804(1): 012007. https://doi.org/10.1088/1742-6596/1804/1/012007

[23] Mehmood, M., Javed, T., Nebhen, J., Abbas, S., Abid, R., Bojja, G.R., Rizwan, M. (2022). A hybrid approach for network intrusion detection. CMC-Computer, Materials & Continua, 70(1): 91-107. https://doi.org/10.32604/cmc.2022.019127

[24] Zeleke, E.M., Melaku, H.M., Mengistu, F.G. (2021). Efficient intrusion detection system for SDN orchestrated Internet of Things. Journal of Computer Networks and Communications, 2021(1): 5593214. https://doi.org/10.1155/2021/5593214

[25] Alzahrani, A.O., Alenazi, M.J. (2021). Designing a network intrusion detection system based on machine learning for software defined networks. Future Internet, 13(5): 111. https://doi.org/10.3390/fi13050111

[26] Saba, T., Sadad, T., Rehman, A., Mehmood, Z., Javaid, Q. (2021). Intrusion detection system through advance machine learning for the internet of things networks. IT Professional, 23(2): 58-64. https://doi.org/10.1109/MITP.2020.2992710

[27] Hadem, P., Saikia, D.K., Moulik, S. (2021). An SDN-based intrusion detection system using SVM with selective logging for IP traceback. Computer Networks, 191: 108015. https://doi.org/10.1016/j.comnet.2021.108015

[28] AlMasri, T., Snober, M.A., Al-Haija, Q.A. (2022). IDPS-SDN-ML: An intrusion detection and prevention system using software-defined networks and machine learning. In 2022 1st International Conference on Smart Technology, Applied Informatics, and Engineering (APICS), Surakarta, Indonesia, pp. 133-137. https://doi.org/10.1109/APICS56469.2022.9918804

[29] Logeswari, G., Bose, S., Anitha, T.J.I.A. (2023). An intrusion detection system for SDN using machine learning. Intelligent Automation & Soft Computing, 35(1): 867-880. https://doi.org/10.32604/iasc.2023.026769

[30] Alshammari, T.M., Alserhani, F.M. (2022). Scalable and robust intrusion detection system to secure the IoT environments using software defined networks (SDN) enabled architecture. International Journal of Computer Networks and Applications (IJCNA), 9(6): 678-688. https://doi.org/10.22247/ijcna/2022/217701

[31] Raju, V.S.A., Suma, B. (2023). Network intrusion detection for IoT-botnet attacks using ML algorithms. In 2023 7th International Conference on Computation System and Information Technology for Sustainable Solutions (CSITSS), Bangalore, India, pp. 1-6. https://doi.org/10.1109/CSITSS60515.2023.10334188

[32] Bhayo, J., Shah, S.A., Hameed, S., Ahmed, A., Nasir, J., Draheim, D. (2023). Towards a machine learning-based framework for DDOS attack detection in software-defined IoT (SD-IoT) networks. Engineering Applications of Artificial Intelligence, 123: 106432. https://doi.org/10.1016/j.engappai.2023.106432

[33] Kumari, P., Mangat, V., Singh, A. (2023). Comparative analysis of state-of-the-art attack detection models. In 2023 14th International Conference on Computing Communication and Networking Technologies (ICCCNT), Delhi, India, pp. 1-7. https://doi.org/10.1109/ICCCNT56998.2023.10306428

[34] Ferrão, T., Manene, F., Ajibesin, A.A. (2023). Multi-attack intrusion detection system for software-defined internet of things network. Computers, Materials & Continua, 75(3): 4985-5007. https://doi.org/10.32604/cmc.2023.038276

[35] Hacilar, H., Aydin, Z., Güngör, V.Ç. (2024). Network intrusion detection based on machine learning strategies: Performance comparisons on imbalanced wired, wireless, and software-defined networking (SDN) network traffics. Turkish Journal of Electrical Engineering and Computer Sciences, 32(4): 623-640. https://doi.org/10.55730/1300-0632.4091

[36] Meftah, S., Rachidi, T., Assem, N. (2019). Network based intrusion detection using the UNSW-NB15 dataset. International Journal of Computing and Digital Systems, 8(5): 478-487. https://doi.org/10.12785/ijcds/080505

[37] Khosla, C., Saini, B.S. (2020). Enhancing performance of deep learning models with different data augmentation

techniques: A survey. In 2020 International Conference on Intelligent Engineering and Management (ICIEM), London, UK, pp. 79-85. https://doi.org/10.1109/ICIEM48762.2020.9160048

[38] Çetin, V., Yıldız, O. (2022). A comprehensive review on data preprocessing techniques in data analysis. Pamukkale Üniversitesi Mühendislik Bilimleri Dergisi, 28(2): 299-312.

[39] Amato, A., Di Lecce, V. (2023). Data preprocessing impact on machine learning algorithm performance. Open Computer Science, 13(1): 20220278. https://doi.org/10.1515/comp-2022-0278

[40] Velliangiri, S., Alagumuthukrishnan, S.J.P.C.S. (2019). A review of dimensionality reduction techniques for efficient computation. Procedia Computer Science, 165: 104-111. https://doi.org/10.1016/j.procs.2020.01.079

[41] Xu, Y., Goodacre, R. (2018). On splitting training and validation set: A comparative study of cross-validation, bootstrap and systematic sampling for estimating the generalization performance of supervised learning. Journal of Analysis and Testing, 2(3): 249-262. https://doi.org/10.1007/s41664-018-0068-2

[42] Kahloot, K.M., Ekler, P. (2021). Algorithmic splitting: A method for dataset preparation. IEEE Access, 9: 125229-125237. https://doi.org/10.1109/ACCESS.2021.3110745

[43] Chen, P., Li, F., Wu, C. (2021). Research on intrusion detection method based on Pearson correlation coefficient feature selection algorithm. Journal of Physics: Conference Series, 1757(1): 012054. https://doi.org/10.1088/1742-6596/1757/1/012054

[44] Dhal, P., Azad, C. (2022). A comprehensive survey on feature selection in the various fields of machine learning. Applied Intelligence, 52(4): 4543-4581. https://doi.org/10.1007/s10489-021-02550-9

[45] Ozkan-Okay, M., Samet, R., Aslan, Ö., Kosunalp, S., Iliev, T., Stoyanov, I. (2023). A novel feature selection approach to classify intrusion attacks in network communications. Applied Sciences, 13(19): 11067. https://doi.org/10.3390/app131911067

[46] Awad, M., Fraihat, S. (2023). Recursive feature elimination with cross-validation with decision tree: Feature selection method for machine learning-based intrusion detection systems. Journal of Sensor and Actuator Networks, 12(5): 67. https://doi.org/10.3390/jsan12050067

[47] Farahdiba, S., Kartini, D., Nugroho, R.A., Herteno, R., Saragih, T.H. (2023). Backward elimination for feature selection on breast cancer classification using logistic regression and support vector machine algorithms. Indonesian Journal of Computing and Cybernetics Systems, 17(4): 429-440. https://doi.org/10.22146/ijccs.88926

[48] Kamalov, F., Elnaffar, S., Cherukuri, A., Jonnalagadda, A. (2024). Forward feature selection: Empirical analysis. Journal of Intelligent Systems and Internet of Things, 11(1): 44-54. https://doi.org/10.54216/JISIoT.110105

[49] Abdulameer, M.H., Abdullah, M.Z., Jassim, A.K., Al Khalidy, M.M.M. (2024). A hybrid for analyzing text streaming using data mining and machine learning techniques. Journal of Engineering and Sustainable Development, 28(5): 675-680. https://doi.org/10.31272/jeasd.28.5.13

[50] Sarker, I.H. (2021). Deep learning: A comprehensive overview on techniques, taxonomy, applications and research directions. SN Computer Science, 2(6): 1-20. https://doi.org/10.1007/s42979-021-00815-1

[51] Isnain, A.R., Supriyanto, J., Kharisma, M.P. (2021). Implementation of K-Nearest Neighbor (K-NN) algorithm for public sentiment analysis of online learning. Indonesian Journal of Computing and Cybernetics Systems, 15(2): 121-130. https://doi.org/10.22146/ijccs.65176

[52] Wendong, Y., Zhengzheng, L., Bo, J. (2017). A multi-factor analysis model of quantitative investment based on GA and SVM. In 2017 2nd International Conference on Image, Vision and Computing (ICIVC), Chengdu, China, pp. 1152-1155. https://doi.org/10.1109/ICIVC.2017.7984734

[53] Dai, H. (2018). Research on SVM improved algorithm for large data classification. In 2018 IEEE 3rd International Conference on Big Data Analysis (ICBDA), Shanghai, China, pp. 181-185. https://doi.org/10.1109/ICBDA.2018.8367673

[54] Speiser, J.L., Miller, M.E., Tooze, J., Ip, E. (2019). A comparison of random forest variable selection methods for classification prediction modeling. Expert Systems with Applications, 134: 93-101. https://doi.org/10.1016/j.eswa.2019.05.028

[55] Chen, S., Webb, G. I., Liu, L., Ma, X. (2020). A novel selective Naïve Bayes algorithm. Knowledge-Based Systems, 192: 105361. https://doi.org/10.1016/j.knosys.2019.105361

[56] Xu, F., Pan, Z., Xia, R. (2020). E-commerce product review sentiment classification based on a Naïve Bayes continuous learning framework. Information Processing & Management, 57(5): 102221. https://doi.org/10.1016/j.ipm.2020.102221

[57] Saad, M.M., Jamil, N., Hamzah, R. (2018). Evaluation of support vector machine and decision tree for emotion recognition of Malay folklores. Bulletin of Electrical Engineering and Informatics, 7(3): 479-486. https://doi.org/10.11591/eei.v7i3.1279

[58] Cömert, Z. (2020). Fusing fine-tuned deep features for recognizing different tympanic membranes. Biocybernetics and Biomedical Engineering, 40(1): 40-51. https://doi.org/10.1016/j.bbe.2019.11.001

[59] Charbuty, B., Abdulazeez, A. (2021). Classification based on decision tree algorithm for machine learning. Journal of Applied Science and Technology Trends, 2(1): 20-28. https://doi.org/10.38094/jastt20165

[60] Yalman, Y., Uyanık, T., Atlı, İ., Tan, A., Bayındır, K.Ç., Karal, Ö., Golestan, S., Guerrero, J.M. (2022). Prediction of voltage sag relative location with data-driven algorithms in distribution grid. Energies, 15(18): 6641. https://doi.org/10.3390/en15186641

[61] Rainio, O., Teuho, J., Klén, R. (2024). Evaluation metrics and statistical tests for machine learning. Scientific Reports, 14(1): 6086. https://doi.org/10.1038/s41598-024-56706-x