Received: 26 April 2024 | Revised: 12 June 2024 | Accepted: 25 June 2024 | Published online: 4 July 2024

REVIEW

Journal of Computational and Cognitive Engineering
2025, Vol. 4(2) 108–120
DOI: 10.47852/bonviewJCCE42023249

BON VIEW PUBLISHING

A Systematic Analysis and Review on Intrusion Detection Systems Using Machine Learning and Deep Learning Algorithms

Sneha Leela Jacob[1] and **Parveen Sultana Habibullah**[1,*]

[1] *School of Computer Science and Engineering, Vellore Institute of Technology, India*

Abstract: An intrusion detection system (IDS) is crucial for defending computer networks and systems from cyberattacks, unauthorized entry, and harmful activities. Machine learning (ML) and deep learning (DL) based IDS is a security solution that uses sophisticated algorithms to automatically identify and predict malicious activity occurring within a computer network. It improves the network's security and identification of threats using various algorithms to monitor network traffic patterns, spot anomalies, and discern between normal and abnormal behavior. This study reviews 58 papers on the topic of IDS that implement various ML and DL techniques. The most commonly used techniques are the support vector machine (SVM), decision tree (DT), random forest (RF), K nearest neighbors, gradient boosting (GB), Naïve-Bayes (NB), multilayer perceptron (MLP), artificial neural network, recurrent neural network, and convolutional neural network (CNN). These techniques are tested on four different datasets: KDD Cup, NSL-KDD, UNSW-NB15, and Kyoto. The experimentation showed that, among ML algorithms, the DT classifier has the best average training time of 2.8s, the best average testing time of 0.08s, but achieved an average accuracy of 97.46% across all datasets. On the other hand, the NB classifier is easier to implement but took an average of 4.40s in training time, 1.28s in testing time, and has the least average accuracy of 74.22% across all datasets. The more sophisticated techniques such as SVM, MLP, GB, and CNN are time consuming with CNN taking the highest time in three out of four datasets. The RF algorithm achieved an average accuracy of 99.51%, the highest level of accuracy among all algorithms. In this way, a comprehensive analysis of the strengths and weaknesses of various ML and DL algorithms for IDS is presented.

Keywords: intrusion detection system, machine learning, deep learning, network security

1. Introduction

The proliferation of digital devices, instances of data breaches, advanced network attacks, and the ubiquitous accessibility of networking technology have heightened the demand for network security. Generally, keys are substituted by two-factor authentication, and devices are equipped with disc encryption. However, the growing use of networking tools can lead to concerns regarding their hardware and software maintenance and their proper use [1]. Intrusion detection system (IDS) is deployed to oversee and assess network activities, identify malevolent behavior, and safeguard networks and computer systems [2].

Figure 1 shows the different types of IDS, such as host IDS, application IDS, and network IDS. A network IDS monitors network traffic to detect and identify any unauthorized access or intrusion attempts. A host IDS is used to monitor a single host, such as a server or PC. Application IDS monitors a limited number of acknowledged apps [3]. IDS employs signature-based, anomaly-based, and hybrid-based techniques to detect and classify intrusions. Signature-based detection identifies known threats by matching their signatures. Anomaly-based detection matches unexpected user behavior to preestablished patterns. Anomaly-based detection is efficient in countering unfamiliar or zero-day threats, but it is prone to greater rates of false positives. A hybrid-based approach integrates numerous strategies to maximize benefits and address limitations associated with depending solely on a single methodology [4].

The general workflow diagram of ML algorithms is given in Figure 2. It commences with datasets, which are acquired by data retrieval from many sources. Subsequently, these data are gathered and subjected to preprocessing to eliminate any impurities and arrange it in a suitable shape for subsequent utilization. Feature extraction is conducted to find the most pertinent information from the data, while feature scaling and selection are carried out to standardize the data and choose the most important features. In the modeling step, a machine learning (ML) model is trained using the chosen features. Once the model has been trained, model evaluation is conducted to gauge the model's performance on data that it has not been previously exposed to. Once the model's performance meets the desired standards, it is subsequently implemented for practical use in real-world applications. Ultimately, monitoring is conducted to guarantee that the model maintains its high performance as fresh data are received.

Automatic learning streamlines the process of selecting, combining, and configuring machine learning models. Implementing automated IDS processes enhances usability and precision by

***Corresponding author:** Parveen Sultana Habibullah, School of Computer Science and Engineering, Vellore Institute of Technology, India. Email: hparveensultana@vit.ac.in

© The Author(s) 2024. Published by BON VIEW PUBLISHING PTE. LTD. This is an open access article under the CC BY License (https://creativecommons.org/licenses/by/4.0/).

Figure 1
Different types of IDS

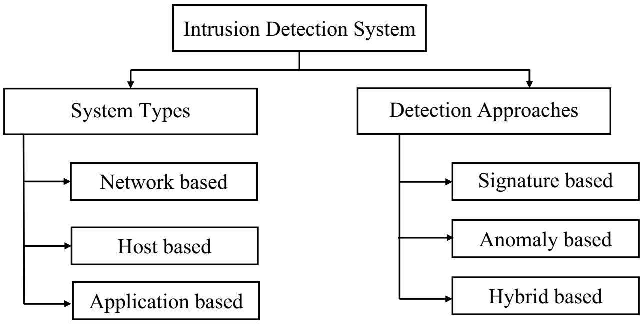
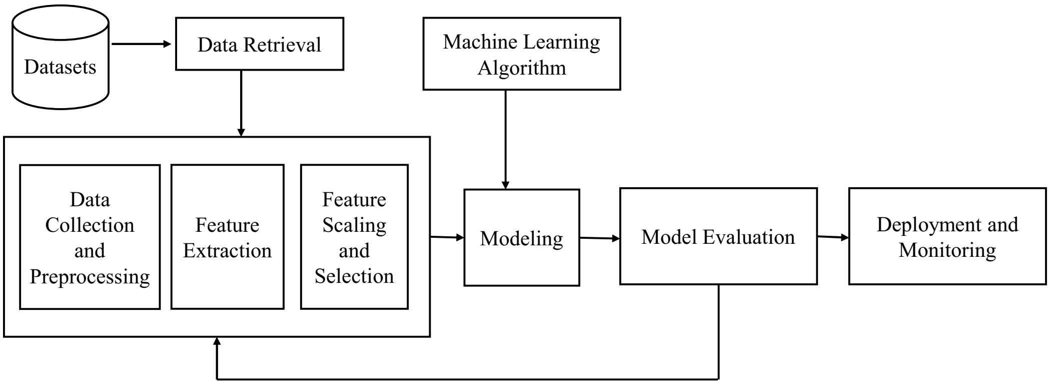


Figure 2
General workflow diagram of ML algorithms


Datasets
Data Retrieval
Machine Learning Algorithm
Data Collection and Preprocessing
Feature Extraction
Feature Scaling and Selection
Modeling
Model Evaluation
Deployment and Monitoring

leveraging ML algorithms to recognize potential risks, identify anomalous behavior, safeguard cloud data, anticipate potential threats, and identify malware [5].

Traditional approaches are constrained in their ability to reliably forecast results for systems that are becoming more intricate and advanced. These methods depend on observational and physical frameworks that necessitate substantial previous knowledge for extracting deteriorating properties. The limitations of traditional methods encompass the requirement for preexisting information, the intricacy in selecting appropriate model functions, and the difficulties in handling complicated systems and noisy data. In addition, conventional techniques, such as rule-based systems, have constraints in handling intricate real-world network data. ML methods, including deep learning, surpass these limitations by effectively dealing with intricate patterns and data with a large number of dimensions [6].

The growing dependence on the internet subjects to various disadvantages. Data breaches undermine the security of personal information, resulting in violations of privacy and damage to one's reputation. Cyberattacks subject individuals to the risks of online fraud and unpleasant content. The proliferation of internet usage in smart cities, driverless automobiles, health monitoring, and phone banking leads to an escalation in hazards [7]. IDS reduces risks by identifying and avoiding illicit use and harmful activity.

The main contributions of this review paper are as follows:

1) The study reviews 58 research articles on the topic of IDS using ML and deep learning (DL) algorithms.
2) The study presents the topic of classifying and detecting intrusions, including data collection, data preprocessing, feature selection and extraction, classification, and efficiency measurement of an IDS. The study presents a detailed review and analysis of ML and DL algorithms for IDS and categorization of labeled datasets with each algorithm's advantages and limitations.
3) This study focuses on 10 distinct algorithms to identify malware across 4 distinct network datasets.
4) The results of training and testing of the chosen 10 algorithms using the 4 datasets are demonstrated.
5) A comprehensive comparison of the chosen ML- and DL-based IDS methods is presented.

2. Literature Review

To gather up the relevant literature, the following inclusion–exclusion criteria are applied.

1) The *year* of the publication is considered. Papers between 2018 and 2024 are included in the study. Papers before 2018 are excluded.
2) The *relevance* of the publications is considered. The included papers should be directly related to IDS and contribute to the understanding of the topic. The excluded papers do not contribute to the understanding of IDS.
3) The *methodology* of the publication is considered. The included papers should use ML algorithms for intrusion detection. The excluded papers do not use ML algorithms for intrusion detection.
4) The *quality* of the publication is considered. The included papers are well-written, with a clear methodology and significant results. The excluded papers are poorly written, lack a clear methodology, or do not present significant results.
5) The *novelty* of the publication is considered. The included papers present new findings or insights that advance the field. The excluded papers do not present new findings or insights.

6) The *credibility* of the publication is considered. The included papers are published in a reputable journal or conference.
7) The *recency* of the publication is considered. Recent papers are included as they contain the most up-to-date research. The excluded papers are those whose findings are outdated.
8) The *datasets* of the publication are considered. The included papers use the following benchmarked networking datasets: KDD Cup, NSL KDD, UNSW-NBB15, and Kyoto. The excluded papers either did not use the considered benchmarked datasets or were not among the chosen 58 papers.

Othman et al. [8] developed the Spark-Chi-SVM model for intrusion detection using SVM classifier on Apache Spark Big Data. They used ChiSq Selector for feature selection and KDD 99 for training and evaluation. The model exhibited strong performance, demonstrated faster training time, and proved beneficial for handling large datasets, as evidenced by an analysis among the Chi-SVM and Chi-Logistic regression classifiers. A limitation of this work is the lack of a multiclass model that can recognize different types of attacks.

Kasongo [2] suggested an IDS framework that integrates recurrent neural network (RNN) techniques with an XGBoost-based feature selection methodology. The study assessed the performance of the classifier by utilizing the UNSW-NB15 and NSL-KDD datasets and compared the performance of long short-term memory (LSTM), Simple ENN, and GRU RNNs. The XGBoost-LSTM binary classification technique demonstrated good performance. A limitation of this work is that it fails to examine the minority classes and the utilization of hybrid methodologies.

Al-Bakaa and Al-Musawi [9] used recurrence qualification analysis to develop an IDS based on anomaly detection. Three feature selection methods are used in this paper: filter-based methods, wrapper methods, and embedded methods. Filter-based methods use statistical tests like chi-square, f-regression scoring function, information gain, and MRMR algorithm to score features based on their correlation with the target variable. Wrapper methods use techniques like forward selection ranking, backward elimination ranking, genetic algorithm, and recursive feature elimination for feature selection. Embedded methods use hybrid methods like Lasso and Elastic Net which combine filter and wrapper methods to achieve good results with low computational cost. The feature selection process in Al-Bakaa and Al-Musawi [9] was done by initially identifying 19 features using decision trees (DT) and random forest (RF) classifiers which achieved good accuracy and F-score rates. Further, the f-regression scoring function was applied that resulted in the selection of the seven relevant features that played a significant role in the detection process. The effectiveness of the system was assessed using the UNSW-NB15 dataset, which demonstrated its ability to accurately identify various forms of attacks.

Thakkar and Lohiya [1] presented an overview of ML-based intrusion detection. The methods to find and categorize incursions using ML techniques were discussed. Data preprocessing that includes cleaning raw network data to improve classifier performance by removing redundancy caused by alterations in packet-level features during network communication was performed. Data normalization or standardization was employed to reduce computing complexity and enable expedited calculation while preserving the significance of features. Feature selection techniques like filter, wrapper, and embedded methods were discussed, each serving different purposes in optimizing the feature set for classification. Feature extraction techniques such as principle component analysis (PCA) were employed to transform raw data into understandable inputs for learning and classification. Additionally, an in-depth review of contemporary ML methods for intrusion detection and classification was presented. Several challenges for ML-based IDS were discussed. Future research directions for improving the efficacy and effectiveness of IDS were discussed.

Sarkar et al. [10] examined two prominent intrusion detection (ID) datasets: KDD Cup 99 and NSL-KDD. The NSL-KDD dataset was preferred over KDD Cup 99 dataset because it addressed the problems of duplicate connections and inaccuracies, which diminished the meaningfulness of the data. To tackle the issue of imbalanced classes in NSL-KDD, a data augmentation technique was employed. This involved generating specialized datasets for each class, with a precise oversampling and undersampling ratio of 1:10. The data augmentation procedure utilized distinct strategies for each class. These techniques included AllKNN SVM SMOTE for the normal class, Repeated Edited Nearest Neighbours ADASYN for the DoS class, One-Sided Selection SMOTEENN for the probe class, Random Under Sample Borderline 1 SMOTE for the R2L class, and Cluster Centroids SMOTE Tomek for the U2R class. A customized classifier model was created using a multilayer perceptron (MLP) with one hidden layer consisting of 128 neurons. The rectified linear unit (ReLU) activation function and Adam optimizer were utilized to forecast the quality of each approach for every class. Their IDS employed a cascaded structure of MLP to improve accuracy, achieving a false positive rate of 1.95% and an 89% detection rate. The approach additionally enhanced the weights, resulting in a precision rate of 87.63% and a false-positive rate of 1.68%.

Debicha et al. [11] presented a framework to bypass NIDS protection in botnet assaults. An authentic adversarial algorithm capable of generating genuine traffic without the need for specific techniques was shown. The architecture incorporated a responsive defence approach influenced by hostile detection, which preserved the original functionality of NIDS. The defence system was adaptable, employing bagging ensemble and machine learning techniques, and incorporated a contextual discounting mechanism. Data preprocessing was done by transforming the categorical data into a binary format that could be used by the model using data filtering and "one-hot encoding." The dataset's inconsistencies were resolved by assigning a value of 0 to "RatioOutIn" instead of using a pseudo-infinite value when "OutBytes" and "InBytes" were both 0. The attacker could modify a feature, and in response, the dependent features were changed using the project function to ensure that semantic restrictions were preserved. The suggested defence mechanism utilized a traffic-based technique, which was considered more practical compared to a feature-based approach. This is because it relied on black-box knowledge rather than white-box knowledge. The approach demonstrated an efficacy in detecting malevolent botnet traffic. More research was called for to further investigate its effectiveness in alternate scenarios.

Jmila and Khedher [12] examined the use of shallow classifiers in MLh-based IDS. The classifiers' resistance to adversarial attacks was evaluated using seven established approaches. One ot encoding was utilized to convert categorical attributes such as protocol, service, and status into a one-hot numeric array, which was then used to prepare the data for classifiers. Data normalization was done to achieve more uniform values, which in turn improved the performance of classifiers by improving the quality of the data that was input to the classifier. This, in turn, enhanced the classifiers' capacity to learn and generate precise predictions in the task of network intrusion detection. In addition, a Gaussian data augmentation defense approach was utilized to assess its effect on the resilience of the classifier. The paper used two benchmarking datasets, namely the UNSW-NB15 and the NSL-KDD, to conduct comprehensive testing under different conditions.

Chen et al. [13] proposed a highly efficient network-based anomaly detection (NBAD) method that utilised LSTM and deep belief networks. The detection of NBAD was improved by tackling challenges associated with high-dimensional and large-scale data, which are reduced efficiency and increased computational workload. Feature extraction methods played a critical role in eliminating duplicate features and dimensionality to enhance the efficiency and accuracy of NBAD. Traditional techniques such as Principal Component Analysis (PCA) and genetic algorithms were employed for feature extraction. However, their effectiveness in terms of accuracy and efficiency was not found to be always optimal. The paper suggested using a Deep Belief Network (DBN) to extract nonlinear features and reduce data dimensionality automatically, while still maintaining accuracy. This was then followed by a light-structure long short-term memory (LSTM) network for classification, resulting in both high accuracy and efficiency. The research also discussed the utilization of Edited Nearest Neighbors (ENN) to clean the dataset. Additionally, a two-layer NBAD method was presented, which combined a Sequence Forward Selection (SFS) algorithm with a decision tree (DT) model for both feature selection and classification. This approach yielded good precision rates. The deep belief network employed dimensionality reduction and feature extraction techniques on high-dimensional network behavior data, that led to reduced processing costs. The LSTM network demonstrated superior accuracy and efficiency when compared with existing approaches.

Douiba et al. [14] presented an efficient intrusion detection model for ensuring the security of IoT systems. The data pre-processing in the paper involved two main steps. First, all inconsistent values such as real and NaN values were identified and removed to ensure data quality. Second, the feature vector and target label were defined and prepared using a Catboost encoder which encoded categorical values to reduce overfitting and normalization problems. Feature selection was employed to build an optimized IDS for IoT security. The paper validated their optimized IDS based on the Catboost classifier combining gradient boosting (GB) and DT algorithms. The feature engineering process was employed to reduce gradient estimation bias and to improve generalization capability, which enhanced the model's accuracy and performance. Categorical features were handled using the Catboost encoder, which encoded these features by greedily using the target statistics on the whole dataset to prevent overfitting and target leakage issues. They employed target metrics and GB techniques to effectively manage enormous volumes of data and to effectively tackle the problem of class imbalance. The model was augmented with GPU and was suggested to provide efficient IDS for IoT networks, demonstrating the resilience of Catboost as a ML technique.

Ali Khan et al. [15] enhanced the efficiency of intrusion detection in a network setting by utilizing the UNSW_NB15 and CICIDS2017 datasets. The missing attribute values in the data were filled, and the data consistency were ensured through the use of regression analysis during the data preprocessing stage. Feature selection was employed to choose relevant features in the dataset. A correlation index was utilized to identify and choose significant features based on their relevance to the target class. Pearson correlation coefficient was utilized to determine the relationship between attributes and to choose the most advantageous ones for optimizing model performance. By conducting correlation analysis, the feature space was reduced from 75 to 15, focusing on the most promising features. A resilient ensemble model was created using AutoML that surpassed conventional models in its ability to detect network intrusions. AutoML was employed to identify the most suitable models.

3. Intrusion Detection Using Machine Learning

This section explains the basic steps in ML-based IDS.

3.1. Data preprocessing

This module provides the data from the networks for the IDS to analysis. A network-based IDS is responsible for the acquisition and iteration of data packets, whereas a host-based IDS concentrates on the retrieval of information related to disc usage and system activities [16]. Steps in ML- and DL-based IDS are depicted in Figure 3.

Data processing is essential for optimizing the training methods of ML models. Public datasets can be accessed for research purposes, and the data can be transformed into appropriate formats for machine learning ML and DL algorithms [17]. Data preparation is essential for ensuring the dependability, accuracy, and resilience of enterprise systems. The precision of real-world data is compromised by the presence of varied persons and software systems.

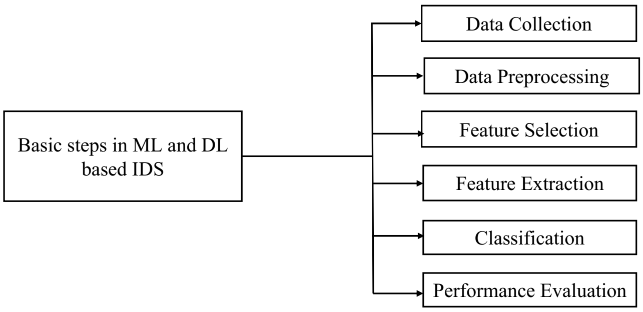
3.1.1. Data profiling

Data profiling refers to the systematic examination, analysis, and evaluation of data in order to gather statistical information pertaining to its quality. The process commences with an examination of the available data and its inherent attributes. Data scientists engage in the process of identifying datasets that are relevant to the specific problem under consideration. They carefully examine the significant properties of these datasets and formulate a hypothesis regarding the features that may have relevance for the specified techniques or ML activity. In addition, connections are established between the data sources and the corresponding business concepts, while also deliberating on the potential utilization of preprocessing libraries. It encompasses a spectrum of techniques, ranging from basic summaries to intricate statistical analyses. This process is crucial for gaining a comprehensive knowledge of the data [18].

3.1.2. Data cleansing

Data cleansing includes the removal of erroneous data, completion of missing data, or any other necessary steps to ensure the suitability of raw data for feature engineering. It is the systematic procedure of identifying and rectifying erroneous or corrupt data inside extensive databases utilized for big data analytics. Data contamination can arise from a multitude of factors, including noise, missing values, discrepancies, and other quality-related concerns. Data cleansing is a process that seeks to improve the overall dependability and precision of the data prior to conducting analytical operations. The study by Hosseinzadeh et al. [19] provides a comprehensive analysis of various methods and approaches used for data cleansing for various big data analysis.

Figure 3
Steps in ML and DL based IDS



3.1.3. Data reduction

Raw datasets often consist of redundant or extraneous data that may result from various methods of defining events or data that is not pertinent to a certain ML, artificial intelligence (AI), or analytics application. Data reduction approaches aim to minimize the duplication present in the initial dataset, allowing for more efficient storage of vast volumes of data in a reduced format. Data reduction strategies offer other benefits, such as enhancing data efficiency. Once data reduction is accomplished, the resultant data become more accessible for AI approaches to utilize in many ways, such as advanced data analytics apps that can significantly simplify decision-making duties [20]. The process of data reduction includes the application of several methods, such as PCA to convert the original raw data into a lower dimensional data for more concise and manageable representation that is better suited for specific purposes or applications.

3.1.4. Data transformation

Data transformations are often utilized methods that have a wide variety of applications in the quantitative analysis of data. They involve applying a mathematical adjustment to the value of a variable [21]. Data specialists engage in the deliberation of how different aspects of the data ought to be structured in order to optimize coherence and alignment with the intended objective. This may involve tasks such as organizing unorganized data, integrating relevant variables when appropriate, or determining significant ranges to prioritize.

3.1.5. Data enrichment

During this stage, data specialists utilize several feature engineering frameworks to implement the necessary changes to the data. The desired outcome is the arrangement of a dataset in a manner that effectively balances the training duration of a novel model with the necessary computational resources [22].

3.2. Feature selection

Feature selection is a crucial process in ML that decreases the number of input variables in an algorithm by keeping just the pertinent data and discarding unnecessary information. It entails the automatic selection of relevant features for a particular model, according to the issue being addressed. This strategy aids in reducing noise throughout the dataset and minimizing the size of the input data [23]. Feature selection is an empirical technique that chooses a specified subset of features from an initial set of attributes [24]. Different feature selection methods in ML.

3.2.1. Filter methods

Filter approaches employ feature ranking to assess feature selection, eliminating those below a certain threshold and retaining those above it [25]. These strategies are frequently employed throughout preprocessing and are highly effective in reducing duplicated, linked, and redundant characteristics. Nevertheless, they fail to tackle the issue of multicollinearity. Feature selection is commonly evaluated on an individual basis, which can be beneficial when features function independently. These techniques are commonly employed in the preprocessing phase and are not classified as ML algorithms.

3.2.2. Wrapper methods

Wrapper approaches, also referred to as greedy algorithms, train algorithms in an iterative manner by using a subset of features. This method, which requires a lot of computational resources, aims to minimize the length of the path to find a suitable collection of features. However, it does not fully make use of simulations involving subsets of features, which has a detrimental influence on both feature selection and prediction accuracy [26]. Wrapper strategies produce optimal feature sets for training models, enhancing accuracy, but they necessitate greater computer resources compared to filter methods.

3.3. Feature extraction

It is a procedure that transforms raw data into numerical characteristics, facilitating convenient manipulation while preserving crucial information. The process entails lowering the dimensions in every dataset to retrieve pertinent information [27]. Manual feature extraction entails the identification and description of pertinent qualities, whereas automated feature extraction employs techniques or deep neural networks to autonomously extract attributes from signals or images, hence eliminating human involvement.

Feature selection involves the process of selecting a subset of the original features in the data, taking into account their relevance and contribution to the predictive model. Conversely, feature extraction refers to the process of generating novel features from the initial data. This is typically achieved by applying transformations or combining the original features to capture crucial information in a space with fewer dimensions.

3.4. Train-test split

A ML-based IDS operates in two stages: training and testing. The training step enhances the quality of the system by effectively training it, while the testing phase assesses the performance of the fundamental approach. A validation set refines the parameters of the selected classifier. The success rate is contingent upon the quantity of instructions and testing. Generally, the preprocessed data are divided into either 80-20 train-test split or 90-10 train-test split [28, 29]. Testing data are utilized to assess the learning of a new model. ML-based IDS algorithms are capable of performing both binary and multiclass categorization [1, 30].

3.5. Algorithms for IDS

This subsection presents the commonly used ML algorithms in IDS setting.

3.5.1. Support vector machine (SVM)

Support vector machine (SVM) is a supervised learning algorithm that classifies target objects into both linear and nonlinear categories. It employs sophisticated computation to transform low-dimensional spaces into high-dimensional spaces, detecting patterns, and anomalies. SVM is employed in diverse domains such as attack categorization, processing images, and text categorization. The technique minimizes risk by precisely identifying the support vectors located on a hyperplane. Additionally, it effectively lowers generalization error by strategically increasing the distance between these support vectors. The SVM algorithm is highly efficient when dealing with data that is easily distinguishable and can even handle nonlinear data by converting it into a feature space with larger dimensions [1, 31, 32].

1) Advantages

SVM is a robust ML method capable of performing nonlinear classification and linear separation of data. It achieves this by constructing a higher dimensional space using the categories included in the training dataset [33]. It exhibits noise resistance, can process both structured and unstructured data, and possesses robust generalization skills.

2) Disadvantages

The computation time for training data in SVMs is lengthy, parameter selection is highly sensitive, understanding is complicated by small model calibrations, convex quadratic programming is numerically costly, performance is influenced by kernel selection, and memory consumption is substantial.

3.5.2. Random forest (RF)

Random forest (RF) is a supervised ML technique that use ensemble learning to construct DT for both regression and classification tasks. The system employs a voting process to generate forecasts, whereby the model's forecast is determined by the largest number of votes received. RF is employed for the purpose of feature selection, classification, and meta-estimation in order to mitigate over-fitting and enhance performance. The model employs the bagging technique to train a collection of random DT, leading to a varied ensemble [34].

1) Advantages

RF is adept at rapidly managing huge datasets that contain a multitude of characteristics. It effectively mitigates the problem of DT overfitting, leading to improved accuracy in classification and regression tasks. Additionally, RF helps prevent data overflow.

2) Disadvantages

The utilization of numerous trees has a detrimental effect on the model's performance, as it consumes substantial resources and processing power, hence posing a challenge for real-time classification prediction [32].

3.5.3. K-nearest neighbor (KNN)

K-nearest neighbor (KNN) is a classifier that is focused on a specific area and estimates a function while delaying calculations until it is evaluated [35]. It is a supervised classification technique that assigns data to a certain class based on its closest neighboring data point. The method determines categories by evaluating the numerical value of k and predicts the types of data samples based on their reliability and proximity to the closest neighbor. KNN is utilized in IDS to identify the most suitable class for a given data point by calculating the minimal distance between the data points and matching them with the associated class. It is applicable for both regression and classification tasks.

1) Advantages

KNN modeling is characterized by its cost-effectiveness, efficiency, and simplicity. It can instantly adapt to random input without the need for a training phase and can effectively handle enormous datasets.

2) Disadvantages

Examining the model presents difficulties stemming from a scarcity of training data, the presence of noise, the need for computationally intensive processes, high costs, the handling of huge datasets, and the curse of dimensionality. Accurate data scaling is essential.

3.5.4. Decision tree (DT)

DT is a type of ML algorithm that is used to analyze data and make predictions in both classification and regression tasks. It is commonly employed in a wide range of applications. The system employs a hierarchical arrangement of nodes and edges, which symbolize categorization clusters and the process of decision-making grounded in factual information. DT is capable of processing both categorical and numerical data. However, its performance may be impacted by imbalanced data. This approach utilizes recursion and greediness to design a robust tree-based structure for data samples.

1) Advantages

The DT classifier produces precise and easily understandable outcomes by estimating probabilities and considering costs. It improves accuracy when combined with common approaches and scenarios, hence boosting stakeholder comprehension [36].

2) Disadvantages

The stability of the DT structure can be compromised by factors such as changes in the data, the level of complexity, and the duration of the training process. Inadequate for scenarios involving limited data, regression analysis, continuous value projections, or combating prepared attacks.

3.5.5. Naïve Bayes (NB)

Naïve Bayes (NB) is a ML classifier that utilizes Bayes' theorem and probabilistic learning to do classification. The process involves classifying unidentifiable data into distinct categories using the probabilities of unique features. The NB algorithm is capable of handling both multiclass and binary classification tasks and is particularly effective when dealing with categorical data. It uses a small amount of training data to estimate performance parameters and calculates the probability of each class before and after the data is observed [37].

1) Advantages

The NB algorithm provides rapid processing time, adaptability, and classification speed, successfully addressing binary and multiclass prediction issues. It also requires less training data by assuming feature independence.

2) Disadvantages

The inference of subclass features by NB is impeded by an assumption of independent characteristics, rendering its application challenging in real-world scenarios and big datasets.

3.5.6. Gradient boosting (GB)

One particularly well-liked boosting algorithm is GB. By minimizing the loss function, each successive model in the training process transforms multiple weak learners into strong learners. The way that this kind of algorithm works is by sequentially adding predictors to an ensemble. To fix the predecessor which it had in the sequence, each prediction that is added to the ensemble is used. Although this method seeks to suit the current data rather than altering the sample values at each iteration. Due to its effectiveness with complicated datasets, it is starting to gain popularity [38].

1) Advantages

GB algorithms train faster, especially on bigger datasets. It is more accurate compared to other nodes generally. Most of them include support for handling categorical features and missing nodes by default.

2) Disadvantages

GB algorithm's resulting models may be difficult to understand and computationally expensive and time consuming to train. They are prone to overfitting.

3.5.7. Multilayer perceptron (MLP)

Multilayer perception (MLP) is a type of artificial neural network (ANN) that consists of multiple interconnected layers. It is trained using the backpropagation algorithm, which involves adjusting the weights of the network based on the error between the predicted and actual outputs. The system comprises input, output, and hidden layers, through which data flows in a unidirectional manner. MLPs can solve issues that cannot be separated linearly and are commonly employed for tasks such as pattern classification, prediction, and approximation. They bear resemblance to a feed-forward network [39].

1) Advantages

MLP can be used to solve sophisticated nonlinear issues. After training, it provides speedy predictions and performs well with massive input data. Even with less data, the same accuracy ratio can be attained.

2) Disadvantages

The influence of independent variables on dependent variables is ambiguous, and computations can be laborious. The quality of training data has a direct impact on the performance of the model. When models are not correct, they have issues in generalizing the learned patterns [40].

3.5.8. Artificial neural network (ANN)

Neural networks are categorization algorithms that imitate the structural organization of the human brain, comprising an input layer, output layer, and hidden layers. These networks have links and operate in a manner analogous to synapses in the human brain. ANNs produce input data by applying nonlinear functions, which establish connections between nodes. The design has interconnected layers, wherein the input and output layers generate output. During the learning process, the weights of neurons and edges undergo changes, initially assigned random weights to each node [41].

1) Advantages

ANNs are capable of analyzing intricate, nonlinear connections. They have the ability to adapt to unexpected input, continue functioning even in the presence of component failures, and make judgments without the need for additional programming.

2) Disadvantages

ANN involves using a large amount of data and optimizing the design and activation function to make the most efficient use of the CPU. Performance is influenced by factors such as layer size and node count, which necessitate a substantial amount of processing time.

3.5.9. Recurrent neural networks (RNN)

Recurrent neural network (RNN) is a powerful DL method designed specifically for handling sequential input. It achieves this by utilizing recurrent layers and memory cells. The system is composed of three layers: input, recurrent, and output. The input layers transform the sensor output into a vector, whereas the recurrent layers offer feedback. The model incorporates input and output layers that are fully connected, perhaps incorporating a SoftMax layer [42].

1) Advantages

RNNs are capable of effectively processing input of any length and time-series predictors. They are able to retain all information throughout time and assign new weights to each time step.

2) Disadvantages

Recurrent computing and training of RNN models can be time intensive, particularly when dealing with extended sequences, which might pose difficulties for certain activation functions. Additionally, RNNs are prone to problems such as exploding gradients and vanishing gradients.

3.5.10. Convolutional neural networks (CNN)

Convolutional neural networks (CNN) is a neural network model that incorporates convolutional, grouping, and fully connected layers. The convolutional layer is responsible for extracting features like edges, textures, and shapes from the input image. On the other hand, pooling techniques are used to decrease the size of the feature map and spatial dimensions. The practice of pooling output in fully linked layers is utilized for picture prediction or classification [43].

1) Advantages

CNN extracts feature automatically. It is extremely accurate in classifying and recognizing images. It has high capacity to work with huge datasets.

2) Disadvantages

It requires a lot of labeled data. It has interpretability issues. It has limited efficiency with respect to sequential data. It takes more time for training. It requires high computational requirements.

3.6. Datasets

Within the realm of IDS, datasets are of utmost importance for the purpose of training and accessing ML models. IDS datasets are utilized for the purpose of training ML algorithms. By being exposed to labeled network traffic data, which includes both benign and malicious instances, the models develop the capability to detect patterns associated with different types of attacks.

3.6.1. KDD 99

The dataset commonly employed for evaluating detection algorithms is KDD 99. This dataset was generated using data obtained from DARPA's 1998 IDS Evaluation Program. The KDD training set comprises around 4.9 million individual connection vectors, each containing 41 distinct characteristics. These vectors are classed either as normal or attack, with a specific focus on one assault. The simulated attacks can be categorized into four types: Denial of Service (DoS), Remote to Local (R2L), User to Root (U2R), and Probing attack [44].

1) Denial of service

During this attack, the target device's memory becomes overwhelmed and occupied, preventing it from responding after allowing the request. Disabling the machine is the most effective method to combat this type of attack.

2) Remote to local

This attack occurs when a hacker who does not have physical access to the target computer is able to send packets from the computer to the system and is then able to take advantage of security flaws in the system to get access to the target machine.

3) User to root

In this attack, a cybercriminal with specific access to a device tries to get control of the router by taking advantage of system flaws. This can be accomplished using a variety of techniques, which includes sniffing, phishing, or social engineering.

4) Probe attack

This attack attempts to gain data via computer networks with the clear intention of evading security safeguards on the system. The KDD dataset classifies attributes into three categories: basic, traffic, and content. Essential characteristics encompass elements that can be removed from an IP/TCP connection, leading to a complete detection delay. Traffic attributes are divided into two groups: same hosts as well as same service attributes. These attributes are determined using a certain window interval. These features are designed to be time based in order to counteract slow-moving probing attacks that monitor ports at longer intervals. U2R and R2L attacks, albeit infrequent, do not often follow a sequential intrusion pattern. Therefore, it is necessary to examine the data section for any abnormal behavior in order to detect these attacks [45].

3.6.2. NSL- KDD

The dataset consists of 125,973 samples from the entire KDD data collection, which is categorized into four groups: KDDTrain+, KDDTest+, KDDTrain-21, and KDDTest-21 [46]. The dataset comprises 43 attributes, which encompass traffic input flow, Label, and Score. The dataset has four categories of KDD: Probe assaults, DoS, U2R, and R2L.

3.6.3. UNSW- NB15

The UNSW-NB15 dataset, created in 2015, is a network intrusion dataset that consists of 42 features. These attributes include nine different types of assaults, such as DoS, analysis, fuzzers, reconnaissance, worms, and shellcode. The dataset comprises 2,540,044 genuine and abnormal network occurrences. The dataset is partitioned into testing and training sets, with distinct records for each type of assault [47].

3.6.4. Kyoto dataset

The Kyoto dataset, derived from real-time network traffic data spanning from 2006 to 2009, encompasses cases from 2006 to 2015 and comprises 14 statistical variables [48]. The system exhibits a high level of precision and employs several techniques, such as honeypots, web crawlers, email sensors, and darknet sensors. The dataset has three indicators: a value of 1 represents no assault, −1 represents known attacks, and there are 10 additional features.

3.7. Performance measures for an IDS

Enhancing computer security depends on the intrusion detection system's effectiveness. It educates users of the IDS's advantages and disadvantages while giving service creators the information and conclusions they need to improve their IDS. The effectiveness of an IDS is evaluated by its capacity to operate precisely and categorize occurrences as attacks or regular behavior using its predictive skills. The true positive (TP), true negative (TN), false positive (FP), and false negative (FN) components of a confusion matrix are used to produce commonly used performance metrics and are shown in Table 1.

Table 1
Confusion matrix

		Predicted	
		0	1
Actual	0	TN	FP
	1	FN	TP

Following are the performance metrics that were determined using the confusion matrix:

True positive (TP): Both actual class and predicted class of data point is 1.

False positive (FP): Actual class is 0 and predicted class is 1.

True negative (TN): Both actual class and predicted class is 0.

False negative (FN): Actual class is 1 and predicted class is 0.

The most commonly used performance measures are accuracy [30], precision, recall [49], F1-Score [50], sensitivity, specificity, false-positive rate (FPR), and false-negative rate (FNR) [51].

3.7.1. Accuracy

Percentage of accurate predictions made by the model. Defined as the ratio of all accurate predictions to all predictions for a given dataset. Accuracy is given by Equation (1).

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

3.7.2. Precision

Precision refers to the percentage of properly diagnosed positive samples compared to the total number of positive samples. Precision is given by equation.

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

3.7.3. Recall

Also called true positive rate (TPR). The TPR is calculated by dividing the number of accurately predicted positive cases (TP) by the total number of actual positive instances. Recall is given by Equation (3).

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

3.7.4. F1 score

The F1 score incorporates both precision and recall evaluating the effectiveness of an algorithm. F1 score is given by Equation (4).

$$F1\,Score = \frac{2 \ast Precision \ast Recall}{Precision + Recall} \tag{4}$$

3.7.5. False positive rate

FPR is typically employed and calculated as the ratio of misclassified predictions (FP) to the total of FP and TN. FPR is given by Equation (5).

$$FPR = \frac{FP}{FP + TN} \tag{5}$$

3.7.6. True negative rate

True negative rate (TNR) is also called model's specificity and is determined as the ratio of samples that are correctly classified (TN) to the total of true negative and false positive (TN+FP). TNR is given by Equation (6).

$$TNR = \frac{TN}{TN + FP} \tag{6}$$

3.7.7. True positive rate

TPR is the proportion of correctly classified predictions to the total of TP and FN. Also known as sensitivity. TPR is given by Equation (7).

$$TPR = \frac{TP}{TP + FN} \tag{7}$$

3.7.8. False negative rate

FNR is the proportion of misclassified predictions to the total of FN and TP (FN+TP). FNR is given by Equation (8).

$$FNR = \frac{FN}{FN + TP} \tag{8}$$

Various evaluation measures possess distinct advantages and disadvantages. Precision is readily comprehensible and interpretable and is well-suited for datasets that are evenly distributed, but it can be deceptive for datasets that are unevenly distributed as it disregards the distribution of classes and does not account for false positives and false negatives. Precision is primarily concerned with reducing the occurrence of FP, which is particularly important when FP have significant consequences. However, precision does not take into account TN, thus a high precision value may result in a poor recall rate. Recall is primarily concerned with reducing the number of FN and is crucial in situations when failing to identify positive cases is very consequential. However, a high recall rate can result in a lower level of precision.

The F1 score achieves a compromise between precision and recall, but it is susceptible to being influenced by class imbalance. The FPR quantifies the percentage of TN that are mistakenly identified as positives. It is particularly important in situations when false alarms have significant consequences. However, it does not take into account the TN; therefore, a high FPR can result in low specificity. The TNR quantifies the accuracy of correctly predicting genuine negatives. It is particularly valuable in situations where minimizing FP is essential. However, it is important to note that a high TNR may result in low sensitivity. The FNR quantifies the percentage of TP cases that are mistakenly classified as negative. It is particularly relevant in situations where it is crucial to identify all positive cases accurately. However, a high FNR can result in a low recall rate. Each of these measures holds significance and should be taken into account depending on the specific demands of the task at hand [52, 53].

Precision, the proportion of TP predictions to the overall number of positive predictions may not be the most suitable metric for imbalanced datasets because of its potential for misinterpretation. In an imbalanced dataset, a model has the potential to attain high accuracy by predicting the majority class for all instances. However, this approach entirely disregards the minority class, which is typically the class of primary interest. However, accuracy and recall are better suited for datasets that have an imbalance in the distribution of classes. Precision, defined as the quotient of true positives divided by the sum of TP and FP, serves to reduce the occurrence of FP and is particularly valuable in situations where the consequences of FP are significant [54].

Recall, defined as the quotient of true positives divided by the sum of TP and FN, holds significance in situations when the consequences of FN are costly. Both precision and recall provide a deeper understanding of a model's performance on the minority class, making them more appropriate for datasets with imbalanced distribution [55].

4. Discussion

The training and testing results on 10 algorithms using the 4 datasets are shown in Figure 4. The attained training and testing times and accuracy are shown.

Figure 4(a) shows the time taken for training and testing, as well as the accuracy of models, using different algorithms on the KDD Cup dataset.

Figure 4
Performance measure for chosen datasets and algorithms

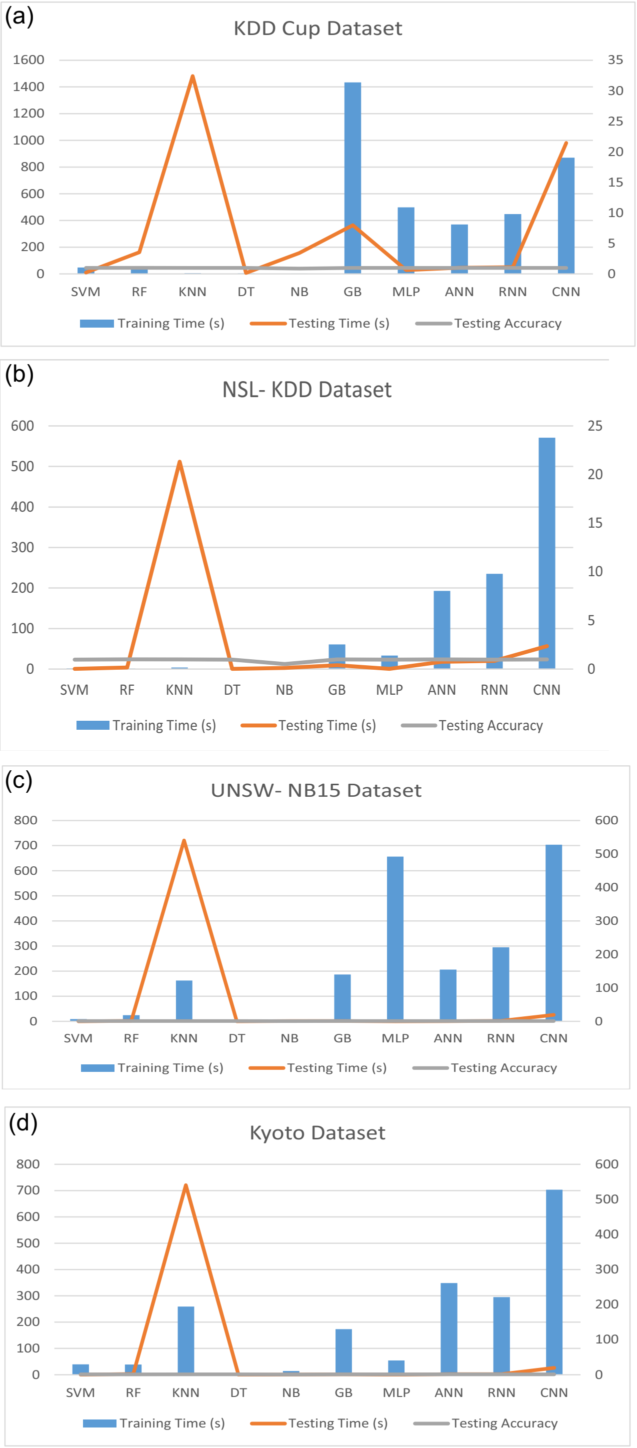


Figure 4(b) shows the time taken for training and testing, as well as the accuracy of models, using different algorithms on the NSL-KDD Cup dataset.

Figure 4(c) shows the time taken for training and testing, as well as the accuracy of models, using different algorithms on the UNSW- NB15 dataset.

Figure 4(d) shows the time taken for training and testing, as well as the accuracy of models, using different algorithms on the Kyoto dataset.

Table 2
Average training and testing time and accuracy for chosen algorithms

Algorithm	Avg. Train Time (s)	Avg. Test Time (s)	Avg. Accuracy (%)
SVM	24.65	0.104	96.37
RF	27.22	1.91	99.51
KNN	107.92	283.75	97.82
DT	2.8	0.08	97.46
NB	4.40	1.28	74.22
GB	463.52	2.62	99.22
MLP	310.58	0.25	96.16
ANN	279.76	0.82	97.76
RNN	318.29	1.07	97.89
CNN	712.23	15.72	97.61

CNN took the largest training time on NSL-KDD dataset, UNSW-NB15 dataset, and Kyoto dataset, while GB took the largest training time in KDD Cup dataset. KNN took the largest testing time on all four datasets.

Table 2 mentions the average training and testing times and accuracy for the chosen algorithms on all four datasets. CNN has the longest average training time followed by GB. DT has the lowest average training and testing time. NB has the minimum test accuracy of 74.22%. All nine algorithms (except NB) achieved average accuracy of more than 95% on the 4 datasets. RF is the ideal option considering the average training and testing times and the obtained accuracy (99.51%) on all four datasets [45].

4.1. Challenges

IDS encounters many challenges. Timely detection of attacks and updating of signatures are essential for real-time intrusion detection but can be challenging due to the continuous evolution of threats. Additionally, there is a significant prevalence of false alarms in which routine network traffic or a genuine attack may be erroneously identified by an IDS. Dealing with different types of network data is a difficult task because of the ability to handle large amounts of data and the global connectedness of network devices that generate diverse data. ML-based IDS face a substantial danger from intrusion evasion assaults, in which attackers alter the ML models to avoid being detected. The accessibility of information for different networks is a matter of concern. Although datasets are widely available, their extensive size, diversity, and quality might result in time-consuming research. Lastly, conducting a thorough assessment of potential dangers is essential for averting any attacks on the system and its resources. However, this task can be intricate and needs significant effort [56].

4.2. Optimization and fine-tuning challenges

Potential biases include dataset bias and label biases. Dataset bias refers to the unique properties of each dataset, which may not accurately represent all types of network traffic. For instance, the KDD Cup '99 dataset, although extensively utilized, has faced criticism due to its inclusion of a substantial number of redundant records, which may result in biased outcomes. The NSL-KDD dataset is an updated iteration of the KDD Cup '99 dataset that specifically tackles this concern. However, it still captures the network traffic patterns from the late 1990s, which might not accurately mirror the current patterns. In Label Bias, the categorizations in these datasets are established according to specific criteria, which may not be flawless. Errors in labeling can lead to biased models when training on these datasets [57].

Confounding variables include feature selection and preprocessing approaches. Feature selection plays a crucial role in determining the effectiveness of IDS models since it can have a substantial impact on their performance. Using different selections of features from these datasets in various research can introduce a confounding element that complicates the comparison of their results. Various preprocessing approaches, such as normalization or dimensionality reduction, can exert a substantial influence on the outcomes. In the absence of a uniform application of these methodologies across much research, there is a possibility that the results may be confounded.

4.3. Open area

Existing datasets, such as KDD Cup and NSL-KDD, have limited attack categories and redundant samples. The CIC-IDS-2017 dataset, for example, should be modified with new attack categories and enhanced input samples. One of the interesting avenues for future study in intrusion detection systems is to take concept drift into account when developing ML- and DL-based IDS, given the evolving nature of networking technology and attack execution concepts. IDS created with ML and DL may experience concept drift because of the static nature of the detection process and the dynamic characteristics of actual network traffic. Therefore, one interesting area is taking the concept drift into account while creating ML and DL-based IDS [1].

The datasets utilized in research studies may not precisely depict network traffic as it occurs in the real world. Real-world networks exhibit dynamism and continual evolution, frequently encountering novel forms of threats. If the IDS is exclusively trained on past data, it might not exhibit satisfactory performance when faced with novel attack patterns.

The features utilized in research studies may not be the most pertinent or all-encompassing for practical applications. The choice of features has a significant impact on the performance of ML and DL models. Real-world situations often involve dynamic changes in the significance of traits, a factor that may not be well accounted for in research investigations.

DL models, specifically, might possess considerable intricacy and necessitate substantial computational resources. This can provide a constraint in practical implementations, particularly in contexts with limited resources [58].

In reality, malicious individuals may employ advanced methods to avoid being detected, such as adversarial ML techniques. These can intentionally deceive ML and DL models and are usually not considered in research papers.

Privacy considerations can arise in real-world deployments of IDS due to the need for access to network traffic data. This is particularly accurate when employing ML/DL techniques, as they have the potential to acquire sensitive information from the data.

Evaluation criteria, including accuracy, precision, and recall, are commonly employed in research studies to assess the performance of IDS. Nevertheless, in practical implementations, additional considerations such as computing expense, response time, and false alarm rate may carry greater significance.

5. Conclusion

The development of IDS has advanced significantly in the realm of network security in recent years. These technologies are essential for preventing harmful activity and illegal access to computer networks. IDS can efficiently detect anomalies and unusual patterns in network activity, both known and unknown, by incorporating ML and DL techniques. The advantage consists of increased scalability, decreased FP, and higher detection accuracy. This paper investigated 10 classification methods to detect malwares within 4 distinct datasets. The DT classifier has the best average training time of 2.8s, the best average testing time of 0.08s, but achieved an average accuracy of 97.46% across all datasets. On the other hand, the NB classifier is easier to implement but took an average of 4.40s in training time, 1.28s in testing time, and has the least average accuracy of 74.22% across all datasets. The RF Classification method is the one that has shown the highest accuracy of 99.51% among all the algorithms.

Conflicts of Interest

The authors declare that they have no conflicts of interest to this work.

Data Availability Statement

Data sharing is not applicable to this article as no new data were created or analyzed in this study.

Author Contribution Statement

Sneha Leela Jacob: Methodology, Software, Validation, Formal analysis, Investigation, Writing – original draft, Writing – review & editing, Visualization. **Parveen Sultana Habibullah:** Conceptualization, Validation, Supervision, Project administration.

References

[1] Thakkar, A., & Lohiya, R. (2023). A review on challenges and future research directions for machine learning-based intrusion detection system. *Archives of Computational Methods in Engineering, 30*(7), 4245–4269. https://doi.org/10.1007/s11831-023-09943-8

[2] Kasongo, S. M. (2023). A deep learning technique for intrusion detection system using a Recurrent Neural Networks based framework. *Computer Communications, 199,* 113–125. https://doi.org/10.1016/j.comcom.2022.12.010

[3] Kok, S. H., Abdullah, A., Jhanjhi, N. Z., & Supramaniam, M. (2019). A review of intrusion detection system using machine learning approach. *International Journal of Engineering Research and Technology, 12*(1), 8–15.

[4] Surakhi, O., García, A., Jamoos, M., & Alkhanafseh, M. (2022). The intrusion detection system by deep learning methods: Issues and challenges. *The International Arab Journal of Information Technology, 19*(3A), 501–513. https://doi.org/10.34028/iajit/19/3A/10.

[5] Slimani, I., Slimani, N., Achchab, S., Saber, M., El Farissi, I., Sbiti, N., & Amghar, M. (2022). Automated machine learning: The new data science challenge. *International Journal of Electrical and Computer Engineering, 12*(4), 4243–4252. http://doi.org/10.11591/ijece.v12i4.pp4243-4252

[6] Khoei, T. T., Ould Slimane, H., & Kaabouch, N. (2023). Deep learning: Systematic review, models, challenges, and research directions. *Neural Computing and Applications, 35*(31), 23103–23124. https://doi.org/10.1007/s00521-023-08957-4

[7] Scheerder, A. J., van Deursen, A. J. A. M., & van Dijk, J. A. G. M. (2019). Negative outcomes of Internet use: A qualitative analysis in the homes of families with different educational backgrounds. *The Information Society, 35*(5), 286–298. https://doi.org/10.1080/01972243.2019.1649774

[8] Othman, S. M., Ba-Alwi, F. M., Alsohybe, N. T., & Al-Hashida, A. Y. (2018). Intrusion detection model using machine learning algorithm on Big Data environment. *Journal of Big Data, 5*(1), 34. https://doi.org/10.1186/s40537-018-0145-4

[9] Al-Bakaa, A., & Al-Musawi, B. (2022). A new intrusion detection system based on using non-linear statistical analysis and features selection techniques. *Computers & Security, 122,* 102906. https://doi.org/10.1016/j.cose.2022.102906

[10] Sarkar, A., Sharma, H. S., & Singh, M. M. (2023). A supervised machine learning-based solution for efficient network intrusion detection using ensemble learning based on hyperparameter optimization. *International Journal of Information Technology, 15*(1), 423–434. https://doi.org/10.1007/s41870-022-01115-4

[11] Debicha, I., Cochez, B., Kenaza, T., Debatty, T., Dricot, J. M., & Mees, W. (2023). Adv-Bot: Realistic adversarial botnet attacks against network intrusion detection systems. *Computers & Security, 129,* 103176. https://doi.org/10.1016/j.cose.2023.103176

[12] Jmila, H., & Khedher, M. I. (2022). Adversarial machine learning for network intrusion detection: A comparative study. *Computer Networks, 214,* 109073. https://doi.org/10.1016/j.comnet.2022.109073

[13] Chen, A., Fu, Y., Zheng, X., & Lu, G. (2022). An efficient network behavior anomaly detection using a hybrid DBN-LSTM network. *Computers & Security, 114,* 102600. https://doi.org/10.1016/j.cose.2021.102600

[14] Douiba, M., Benkirane, S., Guezzaz, A., & Azrour, M. (2023). An improved anomaly detection model for IoT security using decision tree and gradient boosting. *The Journal of Supercomputing, 79*(3), 3392–3411. https://doi.org/10.1007/s11227-022-04783-y

[15] Ali Khan, M., Iqbal, N., Imran, Jamil, H., & Kim, D. H. (2023). An optimized ensemble prediction model using AutoML based on soft voting classifier for network intrusion detection. *Journal of Network and Computer Applications, 212,* 103560. https://doi.org/10.1016/j.jnca.2022.103560

[16] Sadikin, F., van Deursen, T., & Kumar, S. (2020). A ZigBee intrusion detection system for IoT using secure and efficient data collection. *Internet of Things, 12,* 100306. https://doi.org/10.1016/j.iot.2020.100306

[17] Sarhan, M., Layeghy, S., Moustafa, N., Gallagher, M., & Portmann, M. (2024). Feature extraction for machine learning-based intrusion detection in IoT networks. *Digital Communications and Networks, 10*(1), 205–216. https://doi.org/10.1016/j.dcan.2022.08.012

[18] Colleoni Couto, J., Damasio, J., Bordini, R., & Ruiz, D. (2022). New trends in big data profiling. In *Intelligent Computing: Proceedings of the 2022 Computing Conference, 1,* 808–825. https://doi.org/10.1007/978-3-031-10461-9_55

[19] Hosseinzadeh, M., Azhir, E., Ahmed, O. H., Ghafour, M. Y., Ahmed, S. H., Rahmani, A. M., & Vo, B. (2023). Data cleansing mechanisms and approaches for big data analytics: A systematic study. *Journal of Ambient Intelligence and Humanized Computing, 14*(1), 99–111. https://doi.org/10.1007/s12652-021-03590-2

[20] Peng, M., Southern, D. A., Ocampo, W., Kaufman, J., Hogan, D. B., Conly, J., ..., & Ghali, W. A. (2023). Exploring data reduction strategies in the analysis of continuous pressure imaging technology. *BMC Medical Research Methodology, 23*(1), 56. https://doi.org/10.1186/s12874-023-01875-y

[21] Peterson, R. A. (2021). Finding optimal normalizing transformations via bestNormalize. *The R Journal, 13*(1), 310–329. https://doi.org/10.32614/rj-2021-041

[22] Zhou, Y., Ma, L., Ni, W., & Yu, C. (2023). Data enrichment as a method of data preprocessing to enhance short-term wind power forecasting. *Energies, 16*(5), 2094. https://doi.org/10.3390/en16052094

[23] Liu, R., Chen, Z., & Liu, J. (2023). A hybrid intrusion detection system based on feature selection and voting classifier. In *IEEE 47th Annual Computers, Software, and Applications Conference,* 203–212. https://doi.org/10.1109/COMPSAC57700.2023.00034

[24] Singh, V., Balyan, S., & Mathur, M. (2021). Detecting network anomalies using multilayer feature selection techniques and machine learning algorithms. In *2nd Global Conference for Advancement in Technology,* 1–6. https://doi.org/10.1109/GCAT52182.2021.9587542

[25] Pudjihartono, N., Fadason, T., Kempa-Liehr, A. W., & O'Sullivan, J. M. (2022). A review of feature selection methods for machine learning-based disease risk prediction. *Frontiers in Bioinformatics, 2,* 927312. https://doi.org/10.3389/fbinf.2022.927312

[26] Jain, R., & Xu, W. (2023). Artificial Intelligence based wrapper for high dimensional feature selection. *BMC Bioinformatics, 24*(1), 392. https://doi.org/10.1186/s12859-023-05502-x

[27] Mutlag, W. K., Ali, S. K., Aydam, Z. M., & Taher, B. H. (2020). Feature extraction methods: A review. *Journal of Physics: Conference Series, 1591*(1), 012028. https://doi.org/10.1088/1742-6596/1591/1/012028

[28] Bichri, H., Chergui, A., & Hain, M. (2024). Investigating the impact of train/test split ratio on the performance of pre-trained models with custom datasets. *International Journal of Advanced Computer Science and Applications, 15*(2), 331–339. https://doi.org/10.14569/ijacsa.2024.0150235

[29] Gholamy, A., Kreinovich, V., & Kosheleva, O. (2018). Why 70/30 or 80/20 relation between training and testing sets: A pedagogical explanation. *International Journal of Intelligent Technologies and Applied Statistics, 11*(2), 105–111. https://doi.org/10.6148/IJITAS.201806_11(2).0003

[30] Uçar, M. K., Nour, M., Sindi, H., & Polat, K. (2020). The effect of training and testing process on machine learning in biomedical datasets. *Mathematical Problems in Engineering, 2020*(1), 2836236. https://doi.org/10.1155/2020/2836236

[31] Su, L., Bai, W., Zhu, Z., & He, X. (2021). Research on application of support vector machine in intrusion detection. *Journal of Physics: Conference Series, 2037*(1), 012074. https://doi.org/10.1088/1742-6596/2037/1/012074

[32] Thakkar, A., & Lohiya, R. (2021). Attack classification using feature selection techniques: A comparative study. *Journal of Ambient Intelligence and Humanized Computing, 12*(1), 1249-1266. https://doi.org/10.1007/s12652-020-02167-9

[33] Gad, A. G. (2023). Particle Swarm Optimization algorithm and its applications: A systematic review. *Archives of Computational Methods in Engineering, 30*(5), 3471. https://doi.org/10.1007/s11831-021-09694-4

[34] Dey, P., & Bhakta, D. (2023). A new random forest and support vector machine-based intrusion detection model in networks. *National Academy Science Letters, 46*(5), 471–477. https://doi.org/10.1007/s40009-023-01223-0

[35] Mohy-Eddine, M., Guezzaz, A., Benkirane, S., & Azrour, M. (2023). An efficient network intrusion detection model for IoT security using K-NN classifier and feature selection. *Multimedia Tools and Applications, 82*(15), 23615–23633. https://doi.org/10.1007/s11042-023-14795-2

[36] Costa, V. G., & Pedreira, C. E. (2023). Recent advances in decision trees: An updated survey. *Artificial Intelligence Review, 56*(5), 4765–4800. https://doi.org/10.1007/s10462-022-10275-5

[37] Wickramasinghe, I., & Kalutarage, H. (2021). Naive Bayes: Applications, variations and vulnerabilities: A review of literature with code snippets for implementation. *Soft Computing, 25*(3), 2277–2293. https://doi.org/10.1007/s00500-020-05297-6

[38] Vaishali, R. (2023). A hybrid gradient boost model for intrusion detection. In *7th International Conference on Computing Methodologies and Communication,* 1106–1111. https://doi.org/10.1109/ICCMC56507.2023.10084018

[39] Reddy, G. V., Kadiyala, S., Potluri, C. S., Saravanan, P. S., Athisha, G., Mukunthan, M. A., & Sujaritha, M. (2022). An intrusion detection using machine learning algorithm Multi-Layer Perceptron (MlP): A classification enhancement in wireless sensor network (WSN). *International Journal on Recent and Innovation Trends in Computing and Communication, 10,* 139-145. https://doi.org/10.17762/ijritcc.v10i2s.5920

[40] Çolakoğlu, N., & Akkaya, B. (2019). Comparison of multi-class classification algorithms on early diagnosis of heart diseases. In *y-BIS 2019 Conference Book: Recent Advances in Data Science and Business Analytics,* 162–171.

[41] Kumar, K. S., & Nagalakshmi, T. J. (2022). Design of intrusion detection system for wireless ad hoc network in the detection of man in the middle attack using principal component analysis classifier method comparing with ANN classifier. In *14th International Conference on Mathematics, Actuarial Science, Computer Science and Statistics,* 1–6. https://doi.org/10.1109/MACS56771.2022.10022884

[42] Rezk, N. M., Purnaprajna, M., Nordström, T., & Ul-Abdin, Z. (2020). Recurrent Neural Networks: An embedded computing perspective. *IEEE Access, 8,* 57967–57996. https://doi.org/10.1109/ACCESS.2020.2982416

[43] Alzubaidi, L., Zhang, J., Humaidi, A. J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., ..., & Farhan, L. (2021). Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions. *Journal of Big Data, 8*(1), 53. https://doi.org/10.1186/s40537-021-00444-8

[44] Kumar, S., Gupta, S., & Arora, S. (2022). A comparative simulation of normalization methods for machine learning-based intrusion detection systems using KDD Cup'99 dataset. *Journal of Intelligent & Fuzzy Systems, 42*(3), 1749–1766. https://doi.org/10.3233/JIFS-211191

[45] Note, J., & Ali, M. (2022). Comparative analysis of intrusion detection system using machine learning and deep learning algorithms. *Annals of Emerging Technologies in Computing, 6*(3), 19–36. https://doi.org/10.33166/AETiC.2022.03.003

[46] Mohammed, B., & Gbashi, E. K. (2021). Intrusion detection system for NSL-KDD dataset based on deep learning and recursive feature elimination. *Engineering and Technology Journal, 39*(7), 1069–1079. https://doi.org/10.30684/etj.v39i7.1695

[47] Zoghi, Z., & Serpen, G. (2024). UNSW-NB15 computer security dataset: Analysis through visualization. *Security and Privacy, 7*(1), e331. https://doi.org/10.1002/spy2.331

[48] Protić, D. D. (2018). Review of KDD Cup '99, NSL-KDD and Kyoto 2006+ datasets. *Military Technical Courier, 66*(3), 580-596. https://doi.org/10.5937/vojtehg66-16670

[49] Fränti, P., & Mariescu-Istodor, R. (2023). Soft precision and recall. *Pattern Recognition Letters, 167,* 115–121. https://doi.org/10.1016/j.patrec.2023.02.005

[50] Vakili, M., Ghamsari, M., & Rezaei, M. (2020). Performance analysis and comparison of machine and deep learning algorithms for IoT data classification. *arXiv Preprint:2001.09636.* https://doi.org/10.48550/arXiv.2001.09636

[51] Rao, K. B. V. B., Prasad, G. N. R., Amudha, S., Kumar, A., Prasad, D. V. S. S. S. V., & Mohanavel, V. (2022). A significant feature selection to improve the accuracy of a classification algorithm for steel defect. In *4th International Conference on Inventive Research in Computing Applications,* 200–206. https://doi.org/10.1109/ICIRCA54612.2022.9985717

[52] 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

[53] Naidu, G., Zuva, T., & Sibanda, E. M. (2023). A review of evaluation metrics in machine learning algorithms. In *Artificial Intelligence Application in Networks and Systems: Proceedings of 12th Computer Science On-line Conference 2023, 3,* 15–25. https://doi.org/10.1007/978-3-031-35314-7_2

[54] Chen, W., Yang, K., Yu, Z., Shi, Y., & Chen, C. L. P. (2024). A survey on imbalanced learning: Latest research, applications and future directions. *Artificial Intelligence Review, 57*(6), 137. https://doi.org/10.1007/s10462-024-10759-6

[55] Kulkarni, A., Chong, D., & Batarseh, F. A. (2020). Foundations of data imbalance and solutions for a data democracy. In F. A. Batarseh, & R. Yang (Eds.), *Data democracy: At the nexus of artificial intelligence, software development, and knowledge engineering* (pp. 83–106). Academic Press. https://doi.org/10.1016/B978-0-12-818366-3.00005-8

[56] Khraisat, A., Gondal, I., Vamplew, P., & Kamruzzaman, J. (2019). Survey of intrusion detection systems: Techniques, datasets and challenges. *Cybersecurity, 2*(1), 20. https://doi.org/10.1186/s42400-019-0038-7

[57] Shah, M., & Sureja, N. (2025). A comprehensive review of bias in deep learning models: Methods, impacts, and future directions. *Archives of Computational Methods in Engineering, 32*(1), 255–267. https://doi.org/10.1007/s11831-024-10134-2

[58] Tufail, S., Riggs, H., Tariq, M., & Sarwat, A. I. (2023). Advancements and challenges in machine learning: A comprehensive review of models, libraries, applications, and algorithms. *Electronics, 12*(8), 1789. https://doi.org/10.3390/electronics12081789

How to Cite: Jacob, S. L., & Habibullah, P. S. (2025). A Systematic Analysis and Review on Intrusion Detection Systems Using Machine Learning and Deep Learning Algorithms. *Journal of Computational and Cognitive Engineering*, *4*(2), 108–120. https://doi.org/10.47852/bonviewJCCE42023249