Mesopotamian journal of Cybersecurity
Vol.4,No.3, **pp**. 216–232
DOI: https://doi.org/10.58496/MJCS/2024/024; ISSN: 2958-6542
https://mesopotamian.press/journals/index.php/cybersecurity

Research Article

A Novel Hybrid Fusion Model for Intrusion Detection Systems Using Benchmark Checklist Comparisons

Widad K. Mohammed [1,*], Mohammed A. Taha [2], Saleh M. Mohammed [3]

[1] *Ministry of Education, Baghdad, Iraq*

[2] *Ministry of Education, Babylon Education Directorates, Babylon, Iraq*

[3] *Department of Computer Technology Engineering, Technical College, Imam Ja'afar Al-Sadiq University, Baghdad, Iraq.*

ARTICLEINFO

Article history

Received 26 Sep 2024
Accepted 28 Nov 2024
Published 22 Dec 2024

Keywords

IDS

PCA

XGBoost

Random forest

Machine Learning

ABSTRACT

Due to the quick development of network technology, assaults have become more sophisticated and dangerous. Numerous strategies have been put out to target different types of attacks and conduct trials using various approaches. In order to maintain network integrity and ensure network security, intrusion detection systems, or IDSs, are necessary. In this work, we investigate the effects of several feature extraction methods on IDS performance. We analyze the performance of various feature extraction techniques on two well-known intrusion detection datasets, NSL-KDD and CICIDS2017. Two datasets are used to test these approaches. By lowering dimensionality, enhancing data quality, and enabling visualization, principal component analysis (PCA) is a useful preprocessing method. But it's crucial to take into account its drawbacks and use it in conjunction with other preprocessing methods as necessary. The results are classified using the Decision Tree (DT), Random Forest (RF), Extreme Gradient Boosting (XGBoost), and Naive Bayes algorithms. This study aims to compare the final intrusion detection accuracy of each model in order to assess the performance of these approaches and gain a better understanding of the robustness and generalizability of each strategy across different dataset features. The experimental findings showed that the RF method reached a maximum accuracy of 98.57% on the NSL-KDD dataset and 97.10% on the CICIDS2017 dataset when conventional preprocessing was applied. However, with an accuracy of 97.85%, the RF model proved to be the most dependable model when used on the NSL-KDD dataset with both standard and fusion preprocessing.With standard and fusion preprocessing, the RF model achieved the best accuracy of 98.56% in the instance of the CICIDS2017 dataset. The findings demonstrated that PCA-based fusion preprocessing is not always the best option.

1. INTRODUCTION

Intrusion detection systems (IDSs) have become increasingly prominent in the rapidly evolving field of modern technology. As digital infrastructures evolve and develop, so are the complex threats that exist there. Because of the ongoing competition between malevolent entities and safeguarding mechanisms, IDS is not only a defensive measure but also a vital bulwark in maintaining the integrity of the digital world. [1]. An IDS system that monitors network traffic for questionable activity and sends out notifications when it finds it is known as an. It is software that searches a system or network for malicious activities or policy violations. Typically, a security information and event management (SIEM) system is used to collect data centrally or to report any harmful activity or violation to an administrator. An SIEM system utilizes alarm filtering algorithms and aggregates outputs from several sources to differentiate between hostile activity and false warnings [2] [3]. IDSs are categorized based on their location within a system and the kind of activity they monitor. The first intrusion detection system (IDS) to be deployed on a particular endpoint and designed to protect it from both internal and external threats is called a host-based HIDS. In addition to monitoring network traffic to and from the device, such an IDS may be able to look at system logs and track running processes. Even though a HIDS can only see the host machine, it nonetheless has a great deal of internal visibility into the host computer [4] [5]. Network-based intrusion detection systems (NIDS), the second kind of IDS, are made to concentrate on a whole secured network. It can see every item of data traveling over the network and uses each packet's contents and metadata to determine what to do. Although these systems lack internal visibility of the endpoints they defend, their broader perspective provides additional context

*Corresponding author. Email: widadalsaedy@gmail.com

and the capacity to detect widespread threats [6]. An IDS employs one or both main techniques for detecting threats. The signature-based method identifies attacks on particular patterns found in network data, such as the quantity of bytes, 1 s, or 0 s. Additionally, it performs detection on the basis of the malware's known malicious instruction sequence. Signatures are the patterns that the IDS has identified. While fresh malware assaults are more difficult to detect because their pattern (signature) is unknown, signature-based intrusion detection systems can rapidly identify attacks whose pattern (signature) already exists in the system[7] [8]. The other technique is the Anomaly based Method, which was created to identify malware attacks that were unknown since new malware was created so quickly. A trustworthy activity model is created via machine learning in an anomaly based IDS. Any new information is compared to this model and deemed suspicious if it does not match the model. Compared with signature-based IDSs, machine learning-based techniques offer a larger feature set since their models may be tailored to specific hardware and application setups [9] [10]. Preprocessing is a crucial stage in the creation of Machine Learning (ML) models since the caliber of the method used has a big impact on the correctness of the results. The best preprocessing technique must therefore be used from the beginning of the machine learning (ML) and artificial intelligence (AI) development pipeline. In this context, two preprocessing techniques—fusion preprocessing and standard preprocessing—need to be evaluated. Standard preprocessing uses preprocessing techniques to each feature in the dataset separately, whereas fusion preprocessing applies preprocessing techniques to the integrated features after combining several features. Fusion preprocessing techniques are crucial for acquiring data that is prepared for analysis or modeling. By merging, cleaning, converting, and aligning data from numerous sources, these techniques ensure that the finished dataset is comprehensive, trustworthy, and of the highest quality. Consequently, this improves the performance, accuracy, and dependability of later studies and models. Training and testing of both methods should be conducted on intrusion detection datasets [11][42]. The selection of the best ML model depends on several evaluation metrics. These metrics include the F score, precision, recall, and accuracy on normal test samples. The main contributions of this research study are as follows:

1. Development of multi-ML models via feature fusion and standard preprocessing for standard attacks.
2. Two different datasets are used to learn more about each method's robustness and generalizability across various dataset properties.
3. Various algorithms (Naïve Bayes, DT, RF, and XGBoost) are utilized for both training and testing the data and data separation.
4. The results are analysed via different evaluation metrics, and the execution of the algorithms is evaluated on the basis of their ability to predict intrusion.
5. Evaluation of the developed model through explainability analysis.

The work is noteworthy because of its IDS. It includes preparation phases with common preprocessing procedures, such as standard preprocessing and fusion preprocessing, especially PCA. The study also includes the creation of numerous machine learning models via the four machine learning algorithms RF, DT, Naive Bayes, and XGBoost while taking into account typical test cases in learning scenarios related to intrusion detection.

This paper is organized as follows: Section 2 presents a literature review, Section 3 outlines the materials and methods, and Section 4 provides a discussion of the results. Finally, Section 5 concludes with a summary of our findings.

2. LITERATURE SURVEY

In this section, it is essential to explain why the study is done. To accomplish this, we discuss the potential responses to the following queries, which provide more context for the conditions of this field. ***What is the state of the art in terms of attack detection literature and gap analysis?*** Albara Awajan [12] proposed an exciting intrusion detection method for Internet of Things devices, which is based on Deep Learning (DL). To identify malicious traffic that might start an assault on linked Internet of Things devices, the intelligent system employed a four-layer deep fully linked network architecture. The system shows dependable performance during the experimental performance examination for both simulated and actual invasions. The average accuracy was 93.74%. Saeid Sheikhi and PanosKostakos [13] introduced a novel intrusion detection model that selects features via a genetic algorithm (GA) and gradient descent optimization techniques. Initially, a collection of strongly correlated characteristics from the dataset is chosen. Next, a hybrid combination of the particle swarm optimization and grey wolf optimization algorithms is used to train a back propagation neural network. Naveed Chouhan et al. [14] proposed a novel deep convolutional neural network architecture for network intrusion detection that leverages channel boosting and residual learning. This method, which uses a single-class classification algorithm to identify network intrusions, is based on anomaly detection concepts. With accuracy, AU-ROC, and AU-PR scores of 89.41%, 94.73%, and 94.43%, respectively, the model produced results that showed significant enhancements over the prior approach. Huafeng Zhang et al. [15] proposed a method for detecting SQL injection attacks via DL. This approach uses DL techniques to identify SQL injection

attempts within network traffic. The method selects target features on the basis of the specific characteristics of SQL injection attacks and employs requests from URLs or POST packets as training data. These selected features, along with sample data, are then trained via a deep belief network model, resulting in an effective and identifiable model for detecting SQL injection attacks. Sharuka Promodya Thirimanne et al. [16] developed a real-time IDS that can analyse incoming and outgoing network data in real time to detect intrusions. A Deep Neural Network (DNN) trained on 28 characteristics from the dataset makes up the suggested system. The DNN has demonstrated testing performance, with corresponding f1-scores, accuracies, precisions, and recall scores of 81%, 96%, 70%, and 81%, respectively.

The aforementioned research experiments were conducted via constrained ML models, which limited the range of approaches that could be compared and assessed. Finding the most attack detection would be made easier by investigating a wider variety of models. Although these studies have offered insightful information about the creation of ML models, there is always a need for more investigation and growth, especially in the areas of data variety, model selection, and benchmarking against accepted norms. To close these gaps, a new framework that provides transparency and dependability when choosing the best model to identify threats is presented.

A possible study topic could be developed in light of the gap discussion: How might ML improve system security against intrusions while preserving system dependability and effectiveness? In regard to assaults on ML models, it is imperative to consider the optimal model. However, the ML model's construction and selection procedure can be difficult. The first problem involves applying various preprocessing methods to the datasets. The quality of the preprocessing approach utilized has a significant effect on the accuracy of the findings, making it an essential step in the development of ML models. Consequently, it is critical to use the optimum preprocessing method from the outset of the ML and AI development pipeline. Two preprocessing methods, standard preprocessing and fusion preprocessing, need to be assessed in this regard. While fusion preprocessing combines numerous features and applies preprocessing techniques to the integrated features, standard preprocessing applies preprocessing techniques to each feature in the dataset individually. It is necessary to train and test both techniques on the selected dataset k[43][17].

The second problem is choosing the optimal ML model, which is dependent on multiple assessment criteria. The F1 score, precision, recall, and accuracy are the metrics that are measured in the test cases. However, finding the best ML model can include more than just depending on these metrics. Depending on the particular use case, each statistic may be more or less important; thus, a more thorough analysis of the other metrics is needed[44][18].

The third question is as follows: What are the strategies and instruments for achieving the study's goal? It is crucial to establish an assessment and comparison structure to choose the optimal ML model for learning test cases, as explained below:

1. Including the potential for targeted attacks on the model is a necessary step in creating ML models with attack instances [19]. This is accomplished by adding attack example inputs that are purposefully made to trick the model during training. By subjecting the model to these instances, it becomes more resilient to attacks and develops improved attack detection and defense capabilities.
2. It is essential to consider the models' performance on various preprocessing techniques when choosing the optimal ML model to handle attacks [20]. As a result, assessing a model's performance in relation to an attack scenario under normal and fusion preprocessing techniques should be taken into consideration as a useful way to gauge its resilience.

3. METHODOLOGY

This section is divided into three parts. System architecture and formulation in the first section. The datasets and their importance are discussed in the second section, preprocessing techniques can be used to obtain the intrusion detection system discussed in the third section, and the techniques utilized to categorize this dataset are briefly discussed in the fourth section. Fig. 1 describes the proposed system structure in this section.

3.1 System Architecture

Using massive datasets to develop novel ways to enhance IDSs is one of the problems, which is why data mining is used. One method of preprocessing before employing ML models is data mining. Before ML methods are used, they are utilized to explore, extract, and decrease the dimensionality of the data. The approach that is recommended is thoroughly clarified in this paper. The primary goal of this approach is to enhance intrusion detection and identify all forms of attacks through the use of various ML techniques. Fig. 1 depicts the flowchart of the suggested IDS. To increase accuracy and lower the false positive rate, this work builds a robust IDS by utilizing several feature extraction and dimensionality reduction techniques each time.

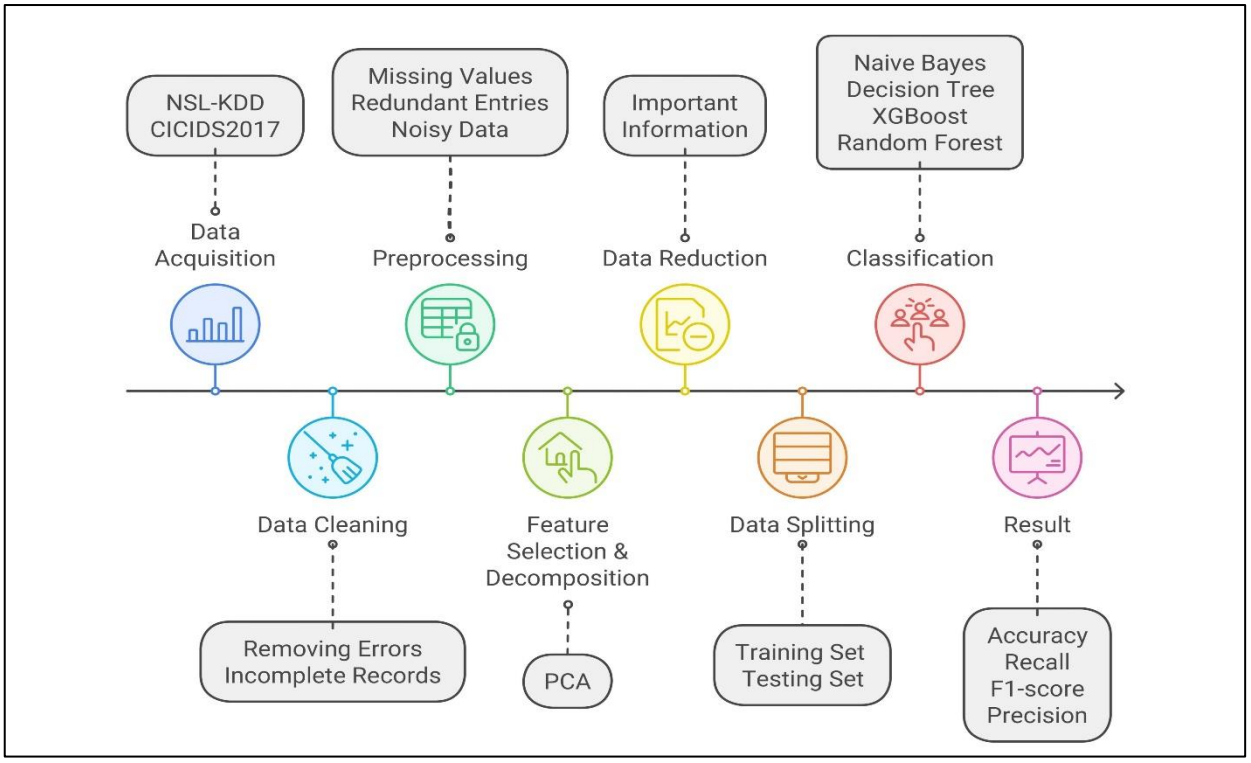

Fig. 1. IDS Structure

The system structure shown in Fig. 1 describes in depth the procedures involved in creating an IDS. First, a dataset containing data on network traffic is acquired. An intrusion detection dataset called NSL-KDD, CICIDS-2017, is utilized. Handling missing values comes in second. Look for any null or missing values or redundant values in the dataset and then select the appropriate treatment technique (imputation or removal). There is encryption on the category variables. The categorical variables in the dataset are converted into numerical values via techniques such as label encoding and one-hot encoding. Numerical features are scaled to a similar range via feature scaling, which keeps any one element from outweighing the others. Two common scaling strategies are minimax scaling and standardization. The relevant features that are most useful in identifying intrusions are taken into consideration when selecting features. To do this, techniques such as correlation analysis and feature importance ranking can be applied.

Using PCA as a fusion preprocessing technique, the following step involves feature selection and decomposition to exclude the covariates that do not correlate with the outcome. After that, the dataset is separated into training and testing sets, one for training and one for evaluating the model's performance. The data is frequently divided into two sets: a training set, which comprises about 80% of the data, and a testing set, which comprises around 20% of the data. This guarantees that the model has been trained on a substantial amount of data while also having enough unknown data to evaluate its generalization ability. The following phases in choosing the best machine learning models for intrusion detection are model selection and training. Commonly used algorithms include the DT, RF, XGBoost, and naïve Bayes approaches. To modify the hyperparameters and prevent overfitting, several models are trained on the training set using appropriate techniques like cross-validation. Measures including F1 score, recall, accuracy, and precision are employed. Selecting the model that performs the best on the testing set based on the assessment parameters chosen is the final step following successful evaluation results. The specifics of the intrusion detection application are typically used to determine the model with the highest accuracy, or F1 score. The suggested system is compared to current models in the comparison study using important assessment criteria like precision, accuracy, recall, and F1 score. In order to evaluate overall system performance and intrusion detection applicability, comparisons also incorporate feature fusion approaches, model selection strategies, and the explainability of model conclusions.

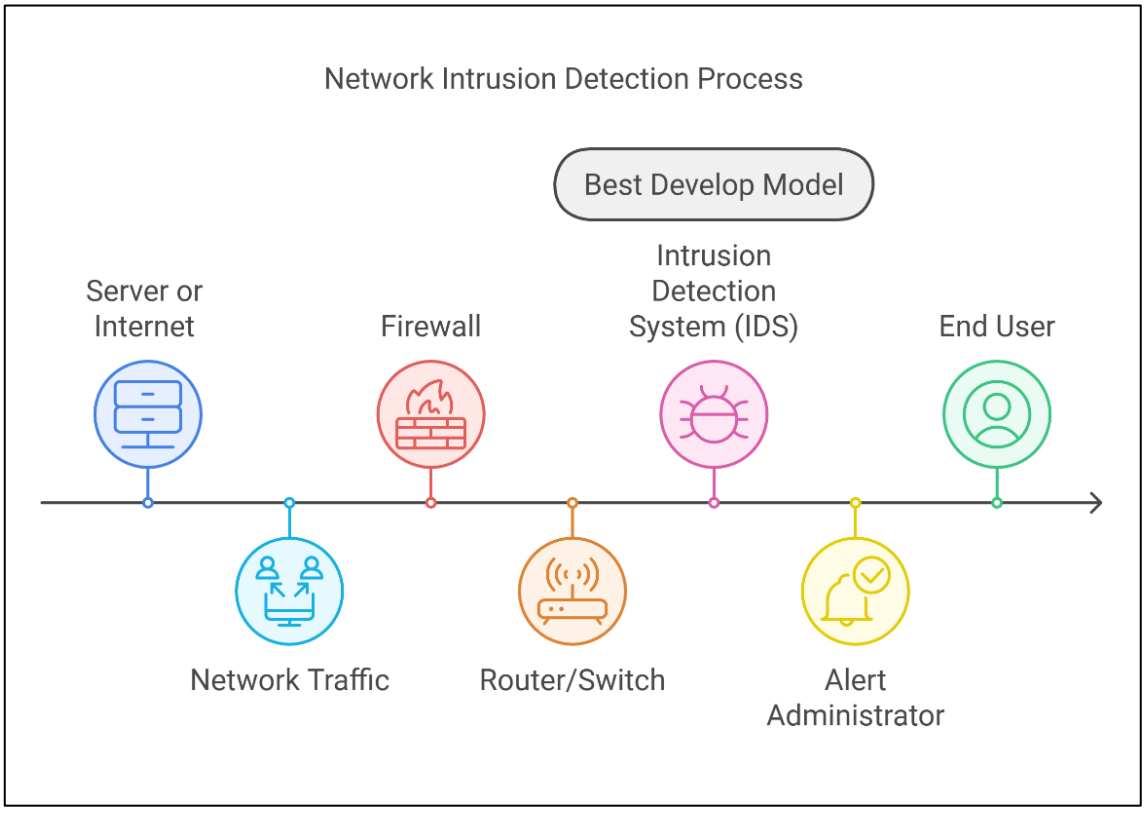

Fig. 2. Network-based IDS

In Fig. 2, By methodically monitoring, identifying, and reacting to possible attacks, the process illustrated provides a strong network intrusion detection system that benefits organizations in real time. This methodical technique guarantees thorough network coverage, facilitating real-time intrusion detection and reaction, which makes it extremely helpful for protecting sensitive institutional data and systems.

3.2 Dataset Description

The NSL-KDD and CICIDS2017 datasets are two distinct and well-known datasets to which the study will be applied. These datasets have been extensively utilized by numerous researchers worldwide in a variety of ways to assess IDS and the efficacy of these approaches in the cybersecurity field.

- • The "NSL-KDD Data Set for Network Intrusion Detection," or NSL-KDD dataset, is frequently used to evaluate how well IDS and ML models detect network intrusions. It includes both regular and labeled assault traffic, where the nature of attack is indicated by each record. This dataset, which is based on a U.S. Air Force LAN, replicates a real military network environment under several types of intrusions. TCP packets labeled as either normal or a particular attack type are transmitted between source and destination IPs in every connection. It includes 41 features (3 qualitative, 38 quantitative) for each connection, categorized as either normal or anomalous.

- The CICIDS-2017 dataset replicates real-world network traffic, including safe data and common attacks, captured as PCAPs. It provides labelled flows with timestamps, IP addresses, ports, protocols, and attack vectors via CICFlowMeter. The data were collected over five days (July 3–7, 2017) and included typical light traffic on Monday and various attacks, such as Brute Force FTP/SSH, DoS, Heartbleed, Web Attack, Infiltration, Botnet, and DDoS, which were conducted on subsequent days in both morning and afternoon sessions [21] [22].

TABLE I. DESCRIPTION OF FILES CONTAINING THE CICIDS2017 DATASET [23].

Name of Files	Day Activity	Attacks Found
Monday-WorkingHours.pcap_ISCX.csv	Monday	Benign (Normal human activities)
Tuesday-WorkingHours.pcap_ISCX.csv	Tuesday	Benign, FTP-Patator, SSH-Patator
Wednesday-workingHours.pcap_ISCX.csv	Wednesday	Benign, DoS GoldenEye, DoS Hulk, DoS slowloris, DoS Slowhttptest, Heartbleed
Thursday-WorkingHours-Morning-WebAttacks.pcap_ISCX.csv	Thursday	Benign, Web Attack - Brute Force, Web Attack - Sql Injection, Web Attack - XSS
Thursday-WorkingHours-Afternoon-Infiltration.pcap_ISCX.csv	Thursday	Benign, Infiltration
Friday-WorkingHours.pcap_ISCX.csv	Friday	Benign, Bot
Friday-WorkingHours-Afternoon-PortScan.pcap_ISCX.csv	Friday	Benign, PortScan
Friday-WorkingHours-Afternoon-DDos.pcap_ISCX.csv	Friday	Benign, DDoS

3.3 Dataset Preprocessing

The dataset preprocessing step is essential for creating successful multi-ML models for possible intrusion detection. The models' capacity to learn and function effectively despite varying assault scenarios is heavily influenced by the evaluation measures for accuracy and relevance.

3.3.1 Standard Preprocessing

Several crucial actions are involved in this stage, including outlining the features of the two datasets (NSL-KDD, CICIDS2017)

- Warnings are ignored to avoid cluttering the output with noncritical messages, and then options are set for pandas to ensure that all columns are displayed when printing data frames. After this, the training and testing data from the CSV files are loaded into Pandas Data Frames [24].
- Data cleaning includes removing irrelevant columns since they have only one value and offer no meaningful information, such as 'num_outbound_cmds'. The NSL-KDD dataset contains several columns from both the training and testing datasets.
- Scaling Numerical Attributes uses StandardScaler from Scikit-Learn to scale the numerical attributes to have a zero mean and unit variance. In CICIDS2017, a couple of rows with information values are removed. This step ensures that all the features contribute equally to the model.
- Assess how well ML models perform using a dataset that has undergone common preprocessing procedures.
- The feature fusion technique combines several features or variables into a single set of features while keeping crucial details from the original features. PCA is a feature fusion method used to reduce the dimensions of datasets.
  The most important patterns or variations in the data are captured by transforming the original features into a new collection of principle components and a linearly uncorrelated set of variables. These two tests are used to build and assess machine learning models' performance in light of the usefulness of the feature fusion method. This phase's output is utilized to assess how well multi-ML models learn from typical test cases.
- Data partitioning: Using train_test_split from scikit-learn, the training dataset was divided into training and validation sets. This stage aids in assessing how well the model performs with unknown data.

These pretreatment steps ensure that the data are clean, correctly formatted, and ready to train machine learning models. Each step enhances the data's quality and, in the end, enhances the models' output.

3.3.2 Fusion Preprocessing using PCA

The feature fusion of the raw data via PCA substantially enhanced the quality of the dataset. PCA, a technique known as feature fusion, combines several features or variables into a single set of features while keeping crucial details from the original features. PCA is a popular method for lowering a dataset's dimensionality, especially in high-dimensional spaces such as those seen in datasets for IDSs that contain network traffic. There are two main advantages of dimensionality

reduction with PCA. The training time of the algorithms decreases significantly with a smaller number of features. It is not always possible to analyse data in high dimensions Although PCA works well for data compression, using it comes with several trade-offs that affect how well IDS performs [25].

1. **Dimensionality** reduction and information loss: IDSs often rely on subtle anomalies to identify advanced attacks. While dimensionality reduction, such as PCA, improves computational efficiency, excessive reduction may lead to false negatives by ignoring critical but subtle indicators of intrusion [26].
2. **Feature interpretability:** PCA transforms original variables into principal components, which reduces redundancy but obscures the physical meaning of the features. Understanding which network attributes indicate an attack is crucial for diagnosing vulnerabilities in IDSs [27].
3. **Capturing Nonlinear Patterns:** As a linear technique, PCA captures only linear relationships. However, many network traffic anomalies detected by IDSs are nonlinear [28].
4. **Selecting the number of components:** Choosing the optimal number of components is critical; too many components can add unnecessary complexity, while too few components risk omitting important information.
5. **Noise sensitivity:** PCA is sensitive to noise, which may result in noise being captured in the leading components, obscuring relevant attack patterns in IDS data [29].
6. **Computational complexity**: Despite reducing the feature count, computing PCA components can be resource intensive, especially with large network traffic datasets. Balancing dimensionality reduction with computational demands is key for real-time IDSs.

Despite the trade-offs of PCA, several strategies can be employed to mitigate the trade-offs associated with PCA in the context of IDS:

1. **Hybrid techniques:** Combining PCA with methods such as autoencoder or kernel PCA helps retain critical information and capture nonlinear relationships [30].
2. **Incremental PCA**: Incremental PCA processes data in smaller batches, reducing the computational load in large-scale systems.
3. **Feature Selection & Cross-Validation:** Applying cross-validation and domain expertise before PCA minimizes the risk of information loss.
4. **Post-PCA Interpretation:** Interpretable models on PCA-transformed data aid analysts in connecting transformed features back to their original meanings.

3.4 Model development

Machine learning, a key component of AI, enables systems to improve user experiences without explicit programming. It employs algorithms and statistical tools for tasks such as diagnosis, prediction, and pattern analysis in datasets of any size. These algorithms support professionals in making informed decisions and achieving positive outcomes. The following are the IDS methods utilized in this work:

3.4.1 Naive Bayes

The Naive Bayes classifier, a superficial learning technique, offers high classification accuracy with efficient computation due to its strong independence assumption. It is perfect for a variety of jobs since it uses probability models and assumes that attributes are conditionally independent within classes. Naive Bayes in IDS determines the probability that each occurrence belongs to a particular class in order to differentiate malicious traffic from legitimate network traffic. It allows for accurate and efficient categorization by allocating the instance to the class with the highest probability [31] [32]. However, this study has two limitations:

1. the assumption of feature independence, which may not hold for network attributes such as packet size and protocol type.
2. difficulty in capturing nonlinear correlations, limiting its effectiveness for sophisticated attacks.

3.4.2 Decision Tree

For dataset analysis and decision-making, the Decision Tree method is frequently utilized in regression and classification. It has a form similar to a flowchart, with internal nodes standing in for features, branching for decisions, and leaf nodes for class labels. The DT creates a predictive model by iteratively selecting the best feature to divide data into homogenous subsets, supporting both continuous and categorical input-output variables. [33].

In IDSs, DTs analyse network traffic anomalies via a tree-based representation of decisions and consequences. When trained with normal traffic patterns, they compare new samples to detect deviations and maintain speed and accuracy in identifying intrusions. When paired with techniques such as autoencoders, DTs can outperform other algorithms in terms of accuracy and precision [34] [35]. However, they may overfit to irregular traffic patterns, leading to false positives, and increased feature complexity (e.g., after PCA) can make trees computationally intensive and less interpretable. Although capable of handling some nonlinear relationships, DTs may struggle with complex, high-dimensional traffic correlations.

3.4.3 XGBoost

XGBoost is an ML technique known for generating accurate predictions using DT. Used in fields such as optical networks, materials research, and human activity recognition, XGBoost handles diverse input dimensions with a specialized decision tree-based method, balancing algorithm complexity and accuracy [36] [37].
In handling noisy network traffic data, XGBoost requires careful hyperparameter tuning to avoid overfitting or poor generalization, with key parameters including the learning rate, estimator count, and tree depth. Although it is faster than traditional gradient-boosting models are, XGBoost remains computationally intensive for large datasets. It captures nonlinear patterns better than PCA alone does, but its performance is heavily reliant on high-quality preprocessing. However, XGBoost's complexity limits interpretability, complicating network incident investigations.

3.4.4 Random Forest

The RF algorithm is an ML method that uses an ensemble of decision trees to classify and evaluate data. Known for its flexibility, it can handle diverse datasets, including those with sparse or ambiguous data. By training multiple decision trees on different data subsets and combining their predictions, the RF increases classification resilience and accuracy.
In an IDS context, RF reduces overfitting and enhances generalization by training each tree on a unique subset of features and data, using a majority vote or average for final predictions. While highly effective in various sectors [38] [39], RFs can be computationally demanding for high-dimensional network data, especially in real-time applications. Training it on PCA-transformed data further increases computational costs, and like PCA, RF can become a "black box" model, where extracting specific feature contributions is challenging. Although RF captures nonlinear patterns, its complexity may overlook subtle high-dimensional patterns not strongly represented in the training set.

3.5 Evaluation Metrics

To identify the most effective classification algorithm, four algorithms were evaluated via performance metrics, including accuracy, recall, F1 score, and precision. This comprehensive evaluation allowed for a robust comparison of each algorithm's effectiveness and inaccurate data classification.

$$Accuracy = \frac{TP+TN}{TP+FP+TN+FN} \quad (1)$$

$$Precision = \frac{TP}{TP+FP} \quad (2)$$

$$Recall = \frac{TP}{TP+FN} \quad (3)$$

$$\text{F1} - \text{score} = 2.\frac{precision.recall}{precision+recall} \quad (4)$$

True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN). Accuracy is the percentage of instances correctly classified by the classifier, and it represents the proximity to the target on average. Different statistical measures, such as precision, recall, and the F-measure, were employed to compare the performance of different algorithms. Precision measures the percentage of expected positives that are actually positive, recall represents the proportion of positives that are correctly classified, and the F-measure balances precision and recall for a classifier [40]

4. RESULTS AND DISCUSSION

This section aims to carry out a broad analysis and discussion of the experimental results of standard and fusion preprocessing and the issues faced by multi-ML algorithms and compare the results obtained with those of previous studies.

4.1 Standard Preprocessing Results of the ML Models

This section discusses the algorithms' performance and classification capabilities for the two datasets. First, four popular classifiers (DT, Naive Bayes, RF, and XGBoost) were applied to the NSL-KDD dataset of intrusion events without fusion preprocessing, as shown in Table II. The CICIDS2017 dataset was subsequently used with the same classifiers to compare the performance with other ML methods by using standard preprocessing without Fusion preprocessing. The performance measurements for the ML algorithms with the CICIDS2017 dataset are shown in Table III.

TABLE II. PERFORMANCE MEASUREMENTS FOR ML MODELS WITH STANDARD PREPROCESSING (NSL-KDD)

CLASSIFIERS	ACCURACY	PRECISION	RECALL	F1-SCORE
Naive Bayes	93.64	83.15	83.20	88.10
Decision Tree	98.26	85.76	85.80	91.60
XGBoost	95.87	96.48	94.70	95.60
Random Forest	98.57	97.91	99.10	98.50

In Table II, among the algorithms, the RF algorithm achieved the highest accuracy at 98.57%, followed closely by the DT algorithm at 98.26%. The RF algorithm also achieves balanced and high values across precision, recall, and the F1 score, indicating strong consistency in correctly identifying both attack instances and normal instances. XGBoost also performed well, reaching an accuracy of 95.87% and a similarly high recall, demonstrating its ability to detect a wide range of intrusion patterns. However, the XGBoost algorithm may still suffer from overfitting, as it tends to create more complex decision boundaries, which can be sensitive to noise. On the other hand, Naive Bayes achieves the lowest accuracy at 93.64% and slightly lower recall and F1-scores than the other classifiers do, suggesting limitations in capturing the complexity of the NSL-KDD dataset.

The relatively lower performance of Naive Bayes likely stems from its assumption of feature independence, which may not hold for network traffic, where features can often be correlated. Overall, the results from NSL-KDD indicate that ensemble methods such as RF and DT provide better generalizability and reliability in detecting intrusions without requiring extensive preprocessing, while Naive Bayes may be more suitable for simpler datasets or as a baseline model. See Fig. 3.

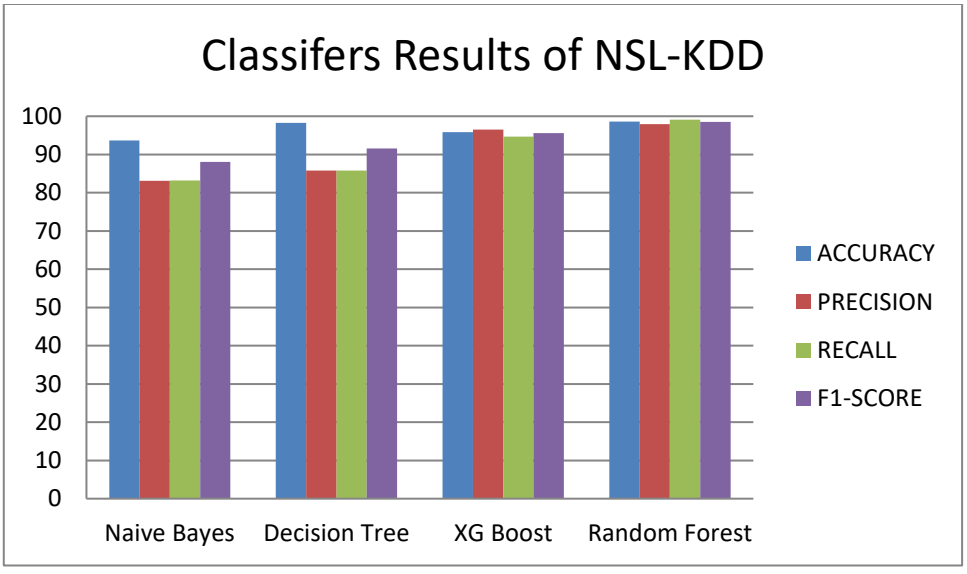

Fig. 3. Results of classifiers with standard preprocessing (NSL-KDD)

Table III presents the performance metrics for the same four classifiers applied to the CICIDS2017 dataset, also without PCA.

TABLE III. PERFORMANCE MEASUREMENTS FOR ML MODELS WITH STANDARD PREPROCESSING (CICIDS2017)

CLASSIFIERS	ACCURACY	PRECISION	RECALL	F1-SCORE
Naive Bayes	94.30	83.10	82.00	81.00
Decision Tree	92.60	86.24	88.00	96.00
XGBoost	97.10	98.06	97.00	96.00
Random Forest	97.10	98.40	99.80	98.80

RF and XGBoost outperformed the other algorithms, achieving the highest accuracy at 97.10%. RF outperforms XGBoost with high precision, recall, and F1-scores. This result suggests that the RF effectively handles the diverse and complex patterns in CICIDS2017, likely due to its ensemble approach, which reduces overfitting while capturing nonlinear relationships in network traffic. DT, while slightly lower in accuracy at 92.60%, excelled in terms of recall and F1-scores, particularly with its F1-score value reaching 96.00%. This high F1 score indicates DT's effectiveness in minimizing false positives and false negatives, making it highly suitable for scenarios where accurate classification of normal traffic is critical. Naive Bayes also achieved high accuracy at 94.30%, with the lowest recall and F1-scores, with a recall of 82.00% and F1-scores of 81.00%, indicating challenges in detecting all intrusion patterns within CICIDS2017. However, similar to NSL-KDD, this outcome reinforces that Naive Bayes may be less effective for datasets with complex, correlated features. The results for CICIDS2017 emphasize the advantages of ensemble models such as RF and XGBoost, which offer higher accuracy and stability across various metrics, particularly when dealing with high-dimensional, complex data such as network traffic. See Fig. 4.

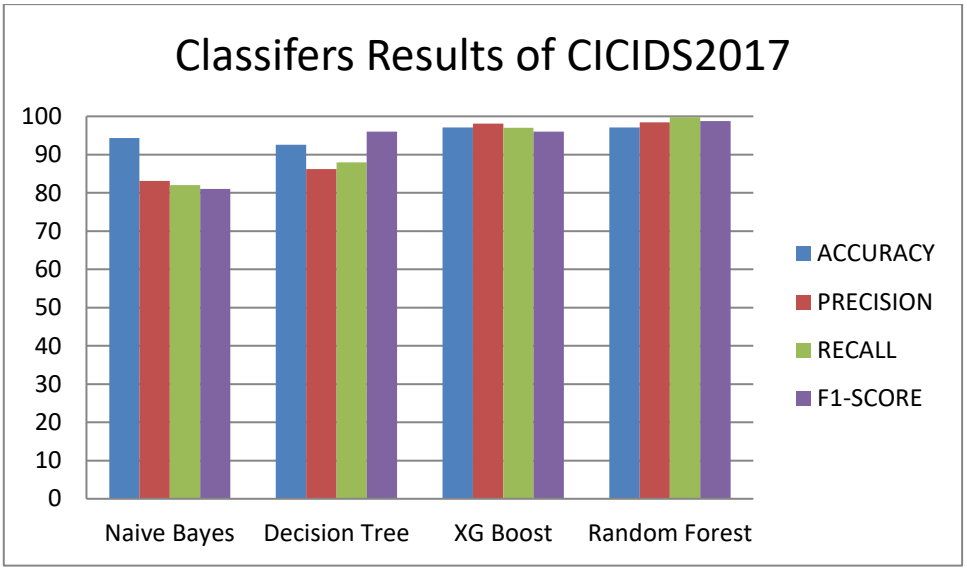

Fig. 4. Results of classifiers with standard preprocessing (CICIDS2017)

4.2 Fusion preprocessing results of the ML models

Fusion preprocessing is the collection of methods and processes used on data before fusion. We are combining various data sources to provide more consistent, accurate, and practical information than what each data source could supply alone. PCA is an important type of fusion preprocessing and is a statistical method for reducing dimensionality in the data while maintaining the greatest amount of data variability. PCA preprocessing was performed on the two datasets of intrusion events, and the same four algorithms were used in this work. In the NSL-KDD dataset, the RF classifier had the highest accuracy, followed by XGBoost, as shown in Table IV.

TABLE IV. PERFORMANCE MEASUREMENTS FOR ML MODELS WITH FUSION PREPROCESSING (NSL-KDD)

CLASSIFIERS	ACCURACY	PRECISION	RECALL	F1-SCORE
Naive Bayes	84.71	85.57	85.51	85.54
Decision Tree	92.53	92.43	93.54	92.98
XGBoost	97.22	96.19	98.64	97.40
Random Forest	97.85	97.82	98.12	97.97

The findings demonstrate that the RF classifier classified intrusion events with a high accuracy of 97.85%. To fill in the gaps and remove extraneous characteristics, the dataset is preprocessed. The traits that would be most useful for the classification task were then found using a feature selection technique. We chose a set of traits based on this analysis because we thought they would yield the best classification outcomes. The RF classifier's 97.85% accuracy rate shows that it can accurately detect intrusion events using the chosen set of attributes. These findings imply that the RF classifier could be a helpful instrument for real-world intrusion detection.

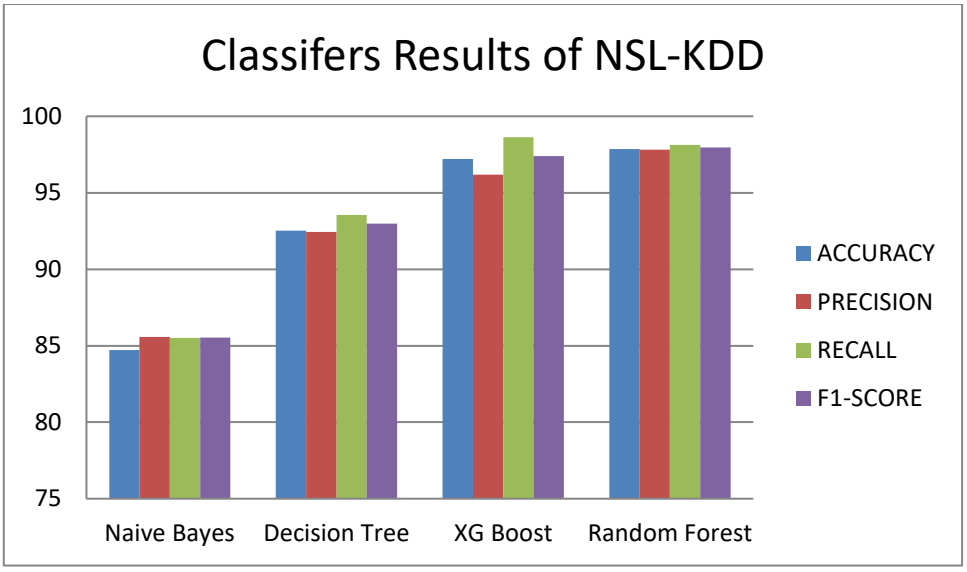

Fig. 5. Results of classifiers with fusion preprocessing (NSL-KDD)

The evaluation results reveal each classifier's performance in terms of accuracy, precision, recall, and F1 score. Naive Bayes achieved 84.71% accuracy but showed slightly lower precision and recall, suggesting challenges in accurately identifying positive cases. XGBoost, with an accuracy of 97.22% and an F1 score of 97.40%, effectively balances recall and precision, minimizing misclassifications. DT performed well, with precision and recall above 92%, making it ideal for applications in which both metrics are balanced. RF was the top performer, achieving 97.85% accuracy, 98.12% recall, and 97.82% precision, demonstrating robust classification and generalization capabilities.

When the results obtained before using the PCA and after using them in the RF, the accuracy decreased because the dimensions of the PCA were reduced. In contrast, for the XGBoost algorithm, the accuracy was higher, where the accuracy reached 97.22%. However, in both cases, RF achieves the highest accuracy before and after PCA is performed. Fig. 5 shows the results of the classifiers with PCA (NSL-KDD).

Table V presents the performance metrics for the same four classifiers applied to the CICIDS2017 dataset, also with PCA.

TABLE V. PERFORMANCE MEASUREMENTS FOR ML MODELS WITH FUSION PREPROCESSING (CICIDS2017)

CLASSIFIERS	ACCURACY	PRECISION	RECALL	F1-SCORE
Naive Bayes	86.00	85.40	81.00	84.20
Decision Tree	93.60	93.30	94.30	93.50
XGBoost	92.10	99.30	98.80	97.50
Random Forest	98.56	98.29	98.40	97.60

Table V shows the performance measurements for the same four algorithms when training on the CICIDS2017 dataset via PCA during preprocessing to reduce dimensionality. The evaluation results of the classifiers provide useful details on how effective the function is in terms of the recall, accuracy, precision, and F1 score. The 98.56% accuracy rate of the RF classifier indicates its ability to accurately detect intrusion events via the chosen set of features. It is followed by the XGBoost classifier; although it achieved a somewhat low accuracy of 92.10%, it achieved high recall, precision, and F1-scores. The DT classifier achieves a satisfactory accuracy of 93.60%, although it falls considerably behind the other classifiers in terms of precision, recall, and F1-score parameters. Naive Bayes obtained the lowest results among the four models, with an accuracy of 86.00%.

Analysing the results before and after applying PCA reveals noticeable variations in the performance of the classification algorithms. While some algorithms experienced improvements in their performance metrics following PCA application, others demonstrated a decline. Despite these fluctuations, the RF method consistently achieves the highest accuracy both with and without PCA preprocessing. Fig. 6 illustrates the performance of the classifiers on the CICIDS2017 dataset after PCA preprocessing.

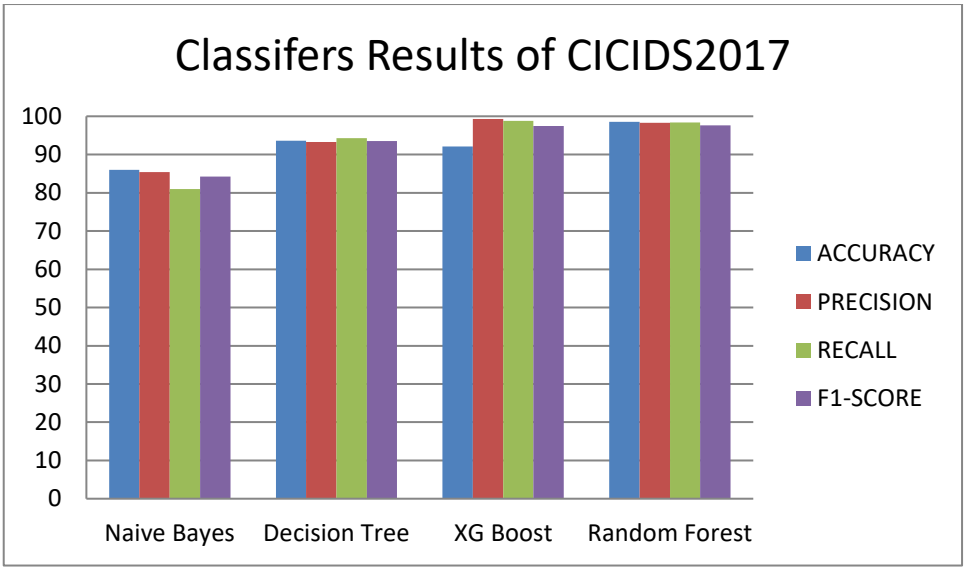

Fig. 6. Results of classifiers with fusion preprocessing (CICIDS2017)

PCA effectively mitigates overfitting and computational constraints by projecting data onto a lower-dimensional subspace defined by principal components, reducing noise and aiding feature extraction for more robust, interpretable models. Among the classifiers, RF achieves the highest accuracy, F1 score, recall, and precision on both the CICIDS2017 and NSL-KDD datasets, with or without PCA, followed by DT, XGBoost, and Naive Bayes. The choice of classifier depends on task requirements and balances precision, recall, and processing efficiency.

This study builds on past research by addressing previous limitations and enhancing ML algorithm performance, underscoring the evolving need for accuracy and adaptability in complex applications.

4.3 Exploitability Analysis

An technique called LIME (Local Interpretable Model-agnostic Explanations) is used to provide an interpretable explanation for the predictions made by intricate machine learning models. In order to approximate the behavior of the original model around a certain prediction, it generates a basic local model, also known as a surrogate model. To find the most important features, this surrogate is trained using significantly altered versions of the input data. LIME supports in the comprehension of black-box models like random forests and neural networks by offering an explanatory list that highlights the significance of features in forecasts. LIME increases the transparency and reliability of machine learning models by providing concise justifications. [41].

To explain its predictions, the study used the LIME method and the best-performing machine learning model. Both the enhanced fusion feature preprocessing model and the conventional preprocessed dataset yielded the best results, according to the evaluation. The results demonstrated the usefulness of LIME in enhancing interpretability and trust in complex models by providing clear explanations for the ML model's predictions.

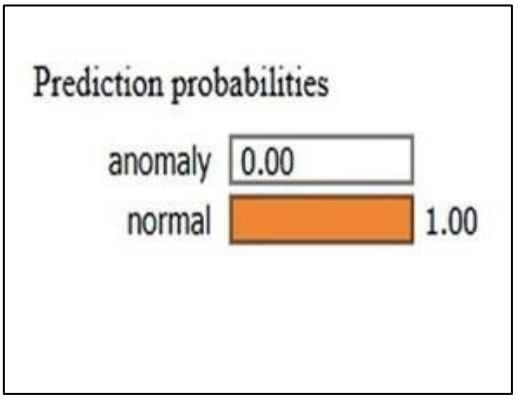

Fig. 7. Results of the prediction probabilities

Fig. 7 shows the plot of the prediction probabilities for the two classes. The probability for the "normal" class is 1.00, which means that the model is completely confident that the sample is classified as "normal". The probability for "anomaly" has a value of 0.00, which means that the model eliminates the possibility that the sample is "anomaly".

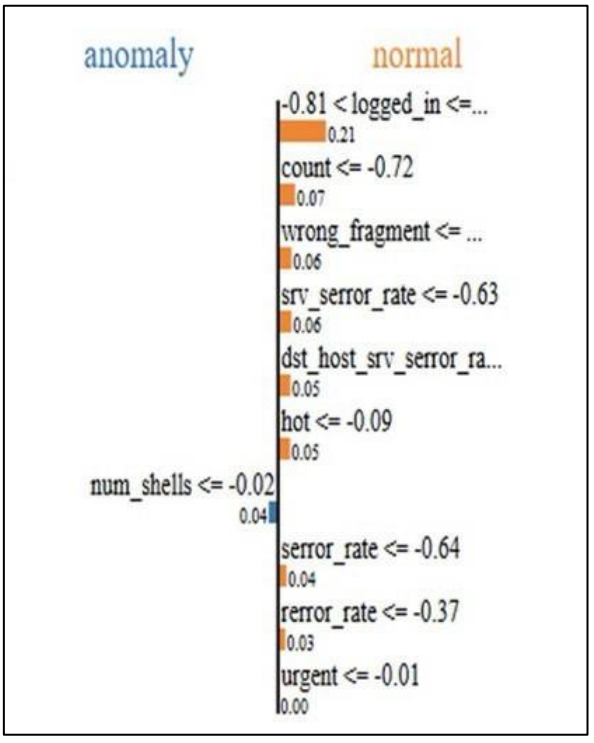

Fig. 8. Illustration of feature contributions in the RF model

Fig. 8 displays the feature contribution interpretation. The graph shows a bar for each feature, indicating its impact on the final result. The features are divided into:

1. Features supporting "normal" classification (orange) strengthened the model's decision that the sample belongs to the "normal" class.
2. Features opposing (blue) these are the features that reduce the probability of classifying the sample as "normal".

The most important feature is **logged_in.** This feature had the most positive impact on strengthening the classification of the sample as "normal". The feature **count** had a relatively weak negative impact but still supported "normal". The other features, **wrong_fragment, srv_serror_rate, and dst_host_srv_serror_rate**, play supporting roles in "normal" classification, but they are less influential than the first feature.

Table VI shows that the feature values with **logged_in = 1.24** contributed significantly to the final score. On the other hand, **num_shells = -0.02** had no significant effect on the final decision.

TABLE VI. FEATURE VALUES FOR THE RF MODEL IN THE LIEM

Feature	Value
logged_in	1.24
Count	-0.73
wrong_fragment	-0.09
srv_serror_rate	-0.63
dst_host_srv_serror_rate	-0.63
hot	-0.09
num_shells	-0.02
serror_rate	-0.64
rerror_rate	-0.37
urgent	-0.01

4.4 Comparative study

A benchmarking checklist was established to compare the results of this study with those of prior studies. presented in Table VI, the checklist includes key works related to IDS models, providing a basis for assessing and comparing the developed framework. This checklist incorporates several critical aspects that are central to the direction of this research, with a particular focus on four main evaluation criteria for IDS models. This study addresses the limitations present in current IDS models and suggests potential areas for future research.
The following factors were used in the comparative analysis:

1. **Feature Fusion**: Multiple features or variables are integrated into a unified feature set while preserving essential information from the original data, substantially improving dataset quality.
2. **Model Selection Process:** This process highlights the benchmarking and selection of development models to identify the most suitable model for IDS applications.
3. **Model** evaluation: This method emphasizes the importance of metric analysis or other evaluation methods to assess the effectiveness of the proposed framework in ranking hybrid diagnostic models.
4. **Model Explainability:** This method focuses on clarifying the IDS method to facilitate an accurate and comprehensive ranking model.

TABLE VII. COMPARISON POINTS IN THE BENCHMARKS AND THE PROPOSED SYSTEM

Comparison points/ benchmarks		**Benchmark#1** [12]	**Benchmark#2** [13]	**Benchmark#3** [14]	**Benchmark#4** [15]	**Benchmark#5** [16]	**Proposed**
Evaluation Metrics	Accuracy	✓	✓	✓	✓	✓	✓
	Recall	✓	✓	✓	X	✓	✓
	F1-score	✓	✓	X	X	✓	✓
	Precision	✓	✓	X	X	✓	✓
	Feature Fusion	X	X	X	X	X	✓
	Model Selection	✓	✓	✓	X	X	✓
	Model Evaluation	X	✓	X	X	X	✓
	Model Explainability	X	X	X	X	X	✓
Total score		62.5%	75%	37.5%	12.5%	50%	100%
Difference		37.5%	25%	62.5%	87.5%	50%	

Table VII highlights the proposed system's clear advantage over prior IDS models across key evaluation criteria. The system achieves high detection accuracy through standard metrics such as accuracy, recall, F1 score, and precision, along with feature fusion for improved data quality, aiding in complex threat detection. Unique to this model is an advanced selection process, allowing for optimal model choice on the basis of data traits and enhancing flexibility and effectiveness. The system also emphasizes evaluation and explainability, increasing transparency and understanding in decision-making. With a total score of 100%, the proposed system represents an efficient solution for network security and intrusion detection.

5. CONCLUSIONS

Machine learning-based network intrusion detection has been created and put into use to improve computer network security. Several intrusion detection techniques are being developed in this work to outperform earlier strategies by offering greater accuracy and superior outcomes. In this study, PCA for network intrusion detection is used to examine the CICIDS2017 dataset and NSL-KDD dataset to assess the efficacy of multiple PCA-based models. The purpose of this study is to evaluate their broad ability to detect assault. Accurate classification and identification of network breaches have been achieved through the use of algorithms such as XGBoost, DT, RF, and naive Bayes techniques. In terms of precision, recall, accuracy, and the F1 score. The experimental results indicate that fusion preprocessing often improves ML model

performance by integrating diverse data perspectives and reducing noise. However, it is not always optimal, as some algorithms have shown decreased accuracy, highlighting its variable effectiveness on the basis of algorithm and dataset characteristics. The findings show that the RF ensemble method continuously produced better results in the NSL-KDD dataset, with an accuracy of 98.57% without PCA and an accuracy of 97.85% with PCA. For the CICIDS2017 dataset, the RF ensemble method produced better results, with an accuracy of 98.56% with PCA, whereas RF and XGBoost had better accuracies of 98.56% without PCA. The framework uses machine learning (ML) to detect potential attacks and encourage prompt actions, strengthening network security and reducing the chance of data breaches. This framework's strong intrusion detection capabilities can significantly enhance computer networks' overall security posture. Future studies and developments in this field may concentrate on how machine learning algorithms evolve over time, incorporating cutting-edge methods like deep learning and anomaly detection while experimenting with feature reduction methods other than PCA. Improving IDSs using state-of-the-art techniques will be crucial for protecting vital systems and data from new threats as networks get more linked.

Conflicts of interest

The author has no conflicts of interest relevant to this article.

Funding

The absence of funding details in the author's paper suggests that the research was entirely self-funded.

Acknowledgement

The author would like to thank the institution for their institutional support, which played a vital role in the implementation of this study.

References

[1] J. K. Jain and A. A. Waoo, “An Artificial Neural Network Technique for Prediction of Cyber-Attack using Intrusion Detection System,” no. 02, pp. 33–42, 2023.

[2] M. Markevych and M. Dawson, “A REVIEW OF ENHANCING INTRUSION DETECTION SYSTEMS FOR CYBERSECURITY USING ARTIFICIAL INTELLIGENCE (AI),” vol. XXIX, no. 3, pp. 30–37, 2023, doi: 10.2478/kbo-2023-0072.

[3] M. Subhi, O. F. Rashid, S. A. Abdulsahib, M. K. Hussein, and S. M. Mohammed, “Anomaly Intrusion Detection Method based on RNA Encoding and ResNet50 Model,” *Mesopotamian J. CyberSecurity*, vol. 4, no. 2, pp. 120–128, 2024.

[4] J. Byrnes, T. Hoang, N. N. Mehta, and Y. Cheng, “A Modern Implementation of System Call Sequence Based Host-based Intrusion Detection Systems,” pp. 218–225, 2020, doi: 10.1109/TPS-ISA50397.2020.00037.

[5] A. Efe, “Comparison of the Host-Based Intrusion Detection Systems and Network-Based Intrusion Detection Systems,” vol. 18, no. 1, pp. 23–32, 2022, doi: 10.18466/cbayarfbe.832533.

[6] S. Kumar, S. Gupta, and S. Arora, “Research Trends in Network-Based Intrusion Detection Systems : A Review,” *IEEE Access*, vol. 9, pp. 157761–157779, 2021, doi: 10.1109/ACCESS.2021.3129775.

[7] H. Kwon, T. Kim, and M. Lee, “Advanced Intrusion Detection Combining Signature-Based and Behavior-Based Detection Methods †,” pp. 1–19, 2022.

[8] P. P. Ioulianou, V. G. Vassilakis, I. D. Moscholios, and M. D. Logothetis, “A Signature-based Intrusion Detection System for the Internet of Things,” no. July, 2018.

[9] S. Einy, C. Oz, and Y. D. Navaei, “The Anomaly- and Signature-Based IDS for Network Security Using Hybrid Inference Systems,” vol. 2021, 2021, doi: 10.1155/2021/6639714.

[10] M. Janati, H. Alami, A. El, and A. El, “Fed-ANIDS : Federated learning for anomaly-based network intrusion detection systems,” *Expert Syst. Appl.*, vol. 234, no. July, p. 121000, 2023, doi: 10.1016/j.eswa.2023.121000.

[11] M. A. Taha, “A Review of Classifications Techniques and computer aided used for Breast Cancer Detection Medical image Features Extraction Classification,” no. 1, pp. 260–271.

[12] I. Networks and A. Awajan, “A Novel Deep Learning-Based Intrusion Detection System for,” 2023.

[13] M. Alazab, M. Alazab, A. Shalaginov, and A. Mesleh, “Intelligent mobile malware detection using permission requests and API calls,” *Futur. Gener. Comput. Syst.*, vol. 107, pp. 509–521, 2020, doi: 10.1016/j.future.2020.02.002.

[14] N. Chouhan, A. Khan, and H. Khan, “Network anomaly detection using channel boosted and residual learning based deep convolutional neural network,” *Appl. Soft Comput. J.*, vol. 83, p. 105612, 2019, doi:

10.1016/j.asoc.2019.105612.
[15] H. Zhang and B. Zhao, “SQL Injection Detection Based on Deep Belief Network,” 2019.
[16] S. Promodya, T. Lasitha, J. Lasith, and Y. Pushpika, “Deep Neural Network Based Real - Time Intrusion Detection System,” 2022.
[17] C. Oumaima, C. Mouad, C. Khalid, and A. Ilyas, “Exploring the Impact of PCA Variants on Intrusion Detection System Performance,” vol. 15, no. 5, pp. 392–400, 2024.
[18] S. Abadi, O. Avram, S. Rosset, T. Pupko, and I. Mayrose, “ModelTeller : Model Selection for Optimal Phylogenetic Reconstruction Using Machine Learning,” vol. 37, no. 11, pp. 3338–3352, 2020, doi: 10.1093/molbev/msaa154.
[19] R. Arboretti, R. Ceccato, L. Pegoraro, and L. Salmaso, “Design choice and machine learning model performances,” no. April, pp. 3357–3378, 2022, doi: 10.1002/qre.3123.
[20] I. Firat, F. Ertam, and A. Sengur, “Machine learning methods for cyber security intrusion detection : Datasets and comparative study,” *Comput. Networks*, vol. 188, no. December 2020, p. 107840, 2021, doi: 10.1016/j.comnet.2021.107840.
[21] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, “Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization,” no. Cic, pp. 108–116, 2018, doi: 10.5220/0006639801080116.
[22] G. Engelen, V. Rimmer, W. Joosen, and K. U. Leuven, “Troubleshooting an Intrusion Detection Dataset : the CICIDS2017 Case Study”.
[23] R. Panigrahi and S. Borah, “A detailed analysis of CICIDS2017 dataset for designing Intrusion Detection Systems,” no. January, 2018.
[24] M. E. Cimen, O. F. Boyraz, M. Z. Yildiz, and A. F. Boz, “A New Dorsal Hand Vein Authentication System Based on Fractal Dimension Box Counting Method,” *Opt. - Int. J. Light Electron Opt.*, p. 165438, 2020, doi: 10.1016/j.ijleo.2020.165438.
[25] M. M. Time-dependent *et al.*, “Overview of PCA-Based Statistical Process- Dimensional Data Overview of PCA-Based Statistical Process-Monitoring Methods for Time-Dependent , High-Dimensional Data,” vol. 4065, no. January, 2018, doi: 10.1080/00224065.2015.11918137.
[26] T. A. H, “Dimensionality Reduction and Classification through PCA and LDA,” vol. 122, no. 17, pp. 4–8, 2015.
[27] M. Briscik, “I MPROVEMENT OF VARIABLES INTERPRETABILITY IN KERNEL,” 2023.
[28] X. Deng, X. Tian, S. Chen, and C. J. Harris, “Nonlinear Process Fault Diagnosis Based on Serial Principal Component Analysis,” pp. 1–13, 2016.
[29] J. R. Beattie and F. W. L. Esmonde-white, “Exploration of Principal Component Analysis : Deriving Principal Component Analysis Visually Using Spectra,” vol. 75, no. 4, pp. 361–375, 2021, doi: 10.1177/0003702820987847.
[30] D. Jain and V. Singh, “Efficient Hybrid Feature Selection model for Dimensionality on Feature Efficient Hybrid Selection model for Dimensionality An Efficient Hybrid Selection model for Dimensionality Reduction on Feature Efficient Hybrid Feature Selection model for Dimensional,” *Procedia Comput. Sci.*, vol. 132, no. Iccids, pp. 333–341, 2018, doi: 10.1016/j.procs.2018.05.188.
[31] H. Listiyono, Z. Budiarso, S. Susilowati, and A. P. Windarto, “Comprehensive Sentiment Analysis of Religious Content Naive Bayes Algorithm Model,” vol. 8, pp. 602–611, 2024, doi: 10.30865/mib.v8i1.7062.
[32] D. Jeevaraj, T. Vijayan, B. Karthik, and M. Sriram, “Feature Selection Model using Naive Bayes ML Algorithm for WSN Intrusion Detection System,” pp. 179–185.
[33] Y. Zhuang and C. Singh, “Learning a Decision Tree Algorithm with Transformers,” pp. 1–24, 2024.
[34] D. Chen, Q. Song, Y. Zhang, L. Li, and Z. Yang, “Identification of Network Traffic Intrusion Using Decision Tree,” vol. 2023, 2023, doi: 10.1155/2023/5997304.
[35] S. Atir, S. Shamim, A. Hannan, and A. Anwar, “Intrusion detection using decision tree classifier with feature reduction technique,” vol. 42, no. 2, pp. 30–37, 2023.
[36] S. Ben, J. Salma, M. Wali, and J. Laurent, “Forecasting gold price with the XGBoost algorithm and SHAP interaction values,” *Ann. Oper. Res.*, no. 0123456789, 2021, doi: 10.1007/s10479-021-04187-w.
[37] A. Prakash, J. Thangaraj, S. Roy, S. Srivastav, and J. K. Mishra, “Model-Aware XGBoost Method Towards Optimum Performance of Flexible Distributed Raman Amplifier,” vol. 15, no. 4, 2023.
[38] Y. Villuendas-rey, “Random forest Algorithm for the Classification of Spectral Data of Astronomical Objects,” 2023.
[39] N. Potyka and F. Toni, “Explaining Random Forests using Bipolar Argumentation and Markov Networks (Technical Report) arXiv : 2211 . 11699v1 [cs . AI] 21 Nov 2022,” pp. 1–23.
[40] S. M. Muhammed, G. Abdul-Majeed, and M. S. Mahmoud, “Prediction of heart diseases by using supervised machine learning algorithms,” *Wasit J. Pure Sci.*, vol. 2, no. 1, pp. 231–243, 2023.
[41] K. C. Pai, S. A. Su, M. C. Chan, C. L. Wu, and W. C. Chao, “Explainable machine learning approach to predict extubation in critically ill ventilated patients: a retrospective study in central Taiwan,” *BMC Anesthesiol.*, vol. 22, no. 1, pp. 1–11, 2022, doi: 10.1186/s12871-022-01888-y.
[42] R. H. K. Al-Rubaye and A. K. TÜRKBEN, “Using Artificial Intelligence to Evaluating Detection of Cybersecurity Threats in Ad Hoc Networks”, BJN, vol. 2024, pp. 45–56, Apr. 2024.

[43] Mohammad Aljanabi, "Safeguarding Connected Health: Leveraging Trustworthy AI Techniques to Harden Intrusion Detection Systems Against Data Poisoning Threats in IoMT Environments", BJIoT, vol. 2023, pp. 31–37, May 2023.

[44] A. S. . Bin Shibghatullah, "Mitigating Developed Persistent Threats (APTs) through Machine Learning-Based Intrusion Detection Systems: A Comprehensive Analysis", SHIFRA, vol. 2023, pp. 17–25, Mar. 2023, doi: 10.70470/SHIFRA/2023/003.