Ensemble-Based Machine Learning Algorithm for Intelligent Network Security Threat Detection

Ye Chunsheng[1*], Miaomiao Fan[2]
[1]School of Water Resources and Transportation,Zhengzhou University; ZhengZhou , 450000, China
[2]Zibo Vocational Institute, Zibo, Shandong, 255300, China
E-mail: yecs_2025@163.com
[*]Corresponding author

Keywords: classification, feature selection, normalization, network security threats, threat detector

Received: July 11, 2024

The rapid development of cyber threats in the cybersecurity field necessitates advanced strategies for prompt identification and reduction. Conventional approaches frequently struggle to adapt to the complexity of contemporary attacks, emphasizing the requirement for creative approaches utilizing machine learning. This paper creates and assesses the "IntelliGuard Threat Detector" algorithm, developed to independently identify and classify a variety of network security risks employing the CICIDS 2017 dataset. By utilizing advanced machine learning methods, the algorithm aims to enhance accuracy and effectiveness in locating abnormal behaviors suggestive of possible security violations. Present methods for network security usually depend on personal intervention and pre-established guidelines, which may not sufficiently handle the ever-changing nature of cyber threats. The "IntelliGuard Threat Detector" algorithm incorporates robust scaler normalization, Composite Rank Ensemble (CORE) feature selection, and a TrioBoost classifier model to boost predictive accuracy and robustness. The proposed IntelliGuard Threat Detector algorithm attains 94% accuracy, 92% precision, 95% recall, 94% F1-score, and 93% geometric mean, surpassing conventional techniques by up to 6% in accuracy, 8% in precision, 5% in recall, 7% in F1-score, and 7% in geometric mean, respectively. This algorithm provides a proactive and scalable approach for network security threat discovery, signifying a noteworthy development in the area of cybersecurity.

Povzetek: Algoritem IntelliGuard Threat Detector uporablja normalizacijo Robust Scaler, izbor značilk Composite Rank Ensemble (CORE) in ansambelski klasifikator TrioBoost (Decision Stump + Logistic Regression + SVM) za samodejno zaznavanje omrežnih groženj; na podatkovni zbirki CICIDS 2017 doseže dobre rezultate.

1 Introduction

The rapid evolution of cyber threats has drastically changed the cybersecurity landscape over the last few years [1]. These threats comprise a wide range of malevolent actions, from comparatively easy malware attacks that can affect individual systems and steal personal data, to modern advanced persistent threats (APTs) coordinated by incredibly talented and well-funded opponents [2]. APTs frequently entail protracted and focused campaigns intended to penetrate the defenses of particular nations or groups to pilfer confidential information, disrupt processes, or obtain illegal entry to vital infrastructure. These dangers are becoming increasingly complex due to their ability to circumvent conventional safety procedures, exploit zero-day vulnerabilities, and employ metamorphic and polymorphic approaches to alter their routines and signatures.

In addition, the number of cyberattacks has increased, with daily updates on ransomware events, phishing tactics, and data breaches impacting the public and private sectors globally [3]. The interconnectedness of contemporary digital ecosystems increases the potential harm of these threats, as a violation in one system can quickly spread to others, resulting in considerable disruption and monetary loss. This growing threat environment highlights the shortcomings of traditional cybersecurity approaches that depend mostly on personal supervision and static protections.

Thus, the critical requirement for more advanced and flexible cybersecurity measures has become evident.

These measures must be able to foresee, identify, and mitigate hazards instantly, adjusting to novel types of attack, and offering complete defense across diverse networked systems. The creation of such advanced defenses is crucial to the protection of not only personal and corporate resources but also national security concerns in a digital society.

Conventional cybersecurity methods frequently depend on signature-based discovery, manual interventions, and rule-based systems. While these techniques have been efficient to some extent, they have numerous drawbacks. Signature-based discovery fails to detect new attacks since it could solely discover recognized attack patterns [4]. Manual interventions, though occasionally essential, are prone to human error and take more time, restricting the receptiveness and scalability of attack discovery. Rule-based systems need constant maintenance and updating, making them incompetent against quickly altering threat settings [5]. To tackle these problems, this paper proposes the "IntelliGuard Threat Detector" algorithm. This new technique uses advanced machine learning methods to independently discover and classify network security attacks. The algorithm uses the CICIDS 2017 dataset, a benchmark for assessing intrusion detection systems, and incorporates numerous important mechanisms to improve its effectiveness. These mechanisms comprise robust scaler normalization to standardize numerical features, Composite Rank Ensemble (CORE) feature selection to prioritize the most indicative features of security attacks, and a TrioBoost classifier model to enhance predictive accuracy and sturdiness.

The main contributions of this paper are as follows:

1. Implementation of the "IntelliGuard Threat Detector" algorithm for independent threat identification and classification.

2. Assessment of the algorithm's effectiveness utilizing the CICIDS 2017 dataset, shows its high accuracy, precision, recall, F1-score, and geometric mean in detecting network security hazards.

This paper aims to increase the effectiveness of threat detection systems by offering a strong machine learning-based remedy capable of adjusting to changing cyber threats. The "IntelliGuard Threat Detector" algorithm is implemented for application across a range of fields, including national defense systems, vital infrastructure security, and enterprise network security.

The rest of this paper is structured as follows: Section 2 provides an overview of previous research conducted in the areas of network security and threat identification using machine learning techniques. Section 3 details the methodology of the "IntelliGuard Threat Detector" algorithm. Section 4 outlines the experimental setup and evaluation metrics used in this paper, as well as the algorithm's outcomes and performance examination. Finally, Section 5 concludes the paper with recommendations for future research directions.

2 Related works

In the swiftly expanding domain of cybersecurity, various research has examined different methodologies for threat identification and reduction. This section reviews notable contributions from recent literature, emphasizing present methodologies and their restrictions. By analyzing these works, existing gaps in research can be recognized, which the "IntelliGuard Threat Detector" algorithm aims to tackle.

Bouchama et al. [6] suggested improving cyber threat discovery by utilizing behavioral modeling of network traffic patterns using machine learning. The authors explored fundamental methodologies such as neural networks, support vector machines, and random forests, emphasizing their proficiency in modeling multifaceted traffic patterns. Their methodology assesses discovery rate, false positives, accuracy, precision, recall, and F1-score, demonstrating substantial advancements compared to traditional approaches.

Sarker et al. [7] suggested "IntrudTree," an intrusion detection approach that utilizes machine learning techniques intended to improve cyber security by ranking and choosing the most crucial security features. This model builds an intrusion detection system that uses a tree-based approach utilizing the chosen features, striving to enhance prediction accuracy and decrease computational difficulty. The efficiency of the IntrudTree model was validated through research on cybersecurity datasets, with performance metrics like precision, recall, F1-score, accuracy, and ROC values, and compared to conventional machine learning approaches such as Naive Bayes, logistic regression, support vector machines, and k-nearest neighbor.

Ferrag et al. [8] provide a comprehensive assessment of intrusion detection systems specifically designed for Agriculture 4.0. They analyze cybersecurity risks and the metrics employed to assess the effectiveness of these systems. The authors categorize intrusion detection systems according to new technologies, including cloud computing, fog/edge computing, network virtualization, autonomous tractors, drones, IoT, the agricultural industry, and smart grids. The authors also examined public datasets and execution frameworks for assessing the performance of these systems. They concluded by highlighting the problems

and potential research areas in cyber security for Agriculture 4.0.

Bertoli et al. [9] suggested the AB-TRAP framework for a network intrusion detection system that utilizes machine learning algorithms that tackle the out-of-date nature of existing datasets and practical considerations for implementation in real-world scenarios. The five-step AB-TRAP process consists of creating attack and legitimate datasets, training machine learning techniques, putting these models into practice, and assessing the models' performance after deployment. The framework was evaluated in both local and global environments to identify TCP port scanning attacks. The results showed that it achieved high accuracy and used low resources, making it a reliable and efficient solution for addressing contemporary network security concerns.

Saheed et al. [10] introduced a machine learning-based intrusion detection system (IDS) specifically designed for IoT contexts. The authors emphasize the increasing security hazards associated with the rapid expansion of IoT devices, as well as the urgent need for strong intrusion detection techniques. The research aims to utilize supervised machine learning techniques to identify different forms of assault in IoT networks. The author's methodology involves performing preprocessing stages on the UNSW-NB15 dataset, which includes feature scaling using min-max normalization. This is followed by dimensionality reduction utilizing principal component analysis (PCA). The authors assessed the efficacy of six machine learning techniques for identifying attacks, achieving comparable effectiveness in terms of accuracy and other evaluation metrics when compared to current methods. This highlights the importance of the author's methodology in tackling crucial security issues in IoT ecosystems.

Sarhan et al. [11] highlight IoT Network Intrusion Detection Systems (NIDS) flaws, discussing regular security breaches and data losses. Their study uses unique feature reduction (FR) and machine learning (ML) methods to improve NIDS technologies for broad applicability across heterogeneous datasets with different features and attack kinds. They evaluate six ML models (Deep Feed Forward, CNN, RNN, DT, LR, and NB) and three feature extraction methods (PCA, auto-encoder, and LDA) on benchmark datasets (UNSW-NB15, ToN-IoT, and CSE-CIC-IDS2018) and discover no single technique is more effective, emphasizing the importance of dataset selection. Standardized benchmark feature sets are recommended for future research on this important topic.

IoT devices come in many different types, which can be dangerous for security. Islam et al. [12] talk about these problems and suggest using machine learning-based intrusion detection systems (IDS) instead of the current methods. Strecker et al. [13] examine the effectiveness of three machine learning techniques—random forest (RF), support vector machine (SVM), and K-nearest neighbor (KNN)—in detecting malware and intrusions in IoT environments. For their investigation, they used the Aposemat IoT-23 dataset. They conclude that all three algorithms show promise for solving current IoT cybersecurity issues.

Lin et al. [14] presented an ensemble learning technique to threat classification in network intrusion detection that was particularly designed for a security monitoring system in renewable energy infrastructure. The technique employs numerous classifiers to enhance the accuracy and dependability of discovering different network attacks, tackling the distinct security difficulties presented by renewable energy systems.

Atul et al. [15] suggest an energy-aware smart home (EASH) framework that utilizes machine learning (ML) to increase the security of cyber-physical systems (CPS). The architecture specifically aims to improve intrusion detection accuracy by detecting anomalies and ensuring reliable communication.

These existing efforts in the field of cybersecurity have made substantial progress in tackling diverse risks, although there are still some areas that need to be addressed. Although the current detection systems are somewhat effective, they have challenges in identifying emerging and changing cybersecurity threats, keeping defenses up to speed, and reducing false alarms. These constraints impede their ability to quickly adjust to evolving threat environments.

The suggested method, called “IntelliGuard Threat Detector,” aims to address these shortcomings by utilizing advanced machine learning techniques. The algorithm utilizes advanced techniques such as data normalization, feature selection, and ensemble learning to improve accuracy, precision, recall, F1-score, and geometric mean in detecting abnormal behaviors that may indicate security breaches. This method allows for proactive and scalable identification of potential dangers, with the ability to adjust to new threats and reduce the occurrence of incorrect alerts. As a result, it provides a more efficient solution for addressing contemporary cybersecurity issues.

Table 1 contains a summary table outlining the findings of the reviewed works, comprising important metrics like accuracy, dataset utilized, and techniques utilized. This table helps readers rapidly contrast these techniques.

Table 1: Summary table

Study	Key Techniques	Dataset Used	Accuracy	Other Key Metrics	Identified Gaps
Bouchama et al. [6]	Neural Networks, SVM, Random Forest	Custom network traffic	High	Precision, Recall, F1-score	Constrained concentration on new attacks, moderate flexibility
Sarker et al. [7]	Tree-based methodology, Feature Selection	Cybersecurity datasets	High	Precision, Recall, F1-score, ROC	Computational intricacy, the requirement for decreased false positives
Ferrag et al. [8]	IDS for Agriculture 4.0, Cloud, IoT, Smart Grids	Public datasets	High	Evaluation of intrusion detection in Agriculture 4.0	Insufficient concentration on general cybersecurity, difficulties with outdated datasets
Bertoli et al. [9]	AB-TRAP framework, Machine Learning	TCP port scanning data	High	Resource efficiency, Practical challenges	Dataset aging and practical execution difficulties
Saheed et al. [10]	Supervised ML, PCA, Feature Scaling	UNSW-NB15	Similar to current methods	Accuracy, Precision, Recall, F1-score	Concentrate on IoT, the requirement to effectively handle new threats, minimal enhancement over previous techniques
Sarhan et al. [11]	Feature Reduction, ML (Deep Feed Forward, CNN, RNN)	UNSW-NB15, ToN-IoT, CSE-CIC-IDS2018	Varies	Feature selection impact, Dataset adaptability	There is no single better technique; it is important to choose suitable datasets
Islam et al. [12]	ML-based IDS (RF, SVM, KNN)	Aposemat IoT-23	High	Malware detection, Intrusion detection	Dataset-specific outcomes, difficulties in generalizing across IoT settings
Strecker et al. [13]	Random Forest, SVM, KNN	Aposemat IoT-23	High	Intrusion Detection, Malware Detection	Constrained generalization to larger IoT cybersecurity problems
Lin et al. [14]	Ensemble learning merging numerous classifiers	Custom dataset from renewable energy security monitoring system	High	Precision, Recall, F1-score	Restricted applicability to non-renewable energy settings; possible computational overhead in real-time applications
Atul et al. [15]	EASH framework, ML for CPS	Smart home data	High	Anomaly detection, Reliable communication	Concentrate on energy savings and generalization to other CPS settings

The reviewed techniques show substantial improvements in several facets of cybersecurity, including intrusion detection, feature selection, and resource effectiveness. However, previous techniques have significant shortcomings:

- **False positives:** Numerous techniques fail to balance detection rates with false positives, resulting in unneeded alerts that burden security teams.
- **Adaptability:** Many models are restricted by their dataset-specific nature, decreasing their efficacy in dynamic and changing threat settings.

- **Computational overhead:** A few techniques, while precise, impose significant computational requirements, rendering them unsuitable for real-time or resource-constrained settings.

The suggested "IntelliGuard Threat Detector" algorithm seeks to fill these gaps by:

- **Decreasing false positives:** Using advanced feature selection and ensemble learning methods to improve precision and recall, thereby reducing false alarms.

- **Enhancing adaptability:** Using a mixture of machine learning methods that are more responsive to new attacks and adaptable to different datasets.

- **Minimizing computational overhead:** Including effective data normalization and feature selection procedures to keep the algorithm scalable and appropriate for real-time applications.

This technique provides a more efficient and proactive solution for contemporary cybersecurity difficulties, establishing IntelliGuard as an important advancement over present SOTA techniques.

3 Methodology

"IntelliGuard Threat Detector" is a new algorithm that uses advanced machine learning techniques to improve the identification and categorization of network security threats. Given the ever-changing nature of cybersecurity threats, conventional approaches frequently prove inadequate for accurately detecting and addressing intricate and constantly changing vulnerabilities. The "IntelliGuard Threat Detector" algorithm tackles these problems by utilizing advanced algorithms and systematic techniques specifically designed for autonomous threat detection. Algorithm 1 presents the "IntelliGuard Threat Detector" algorithm.

Algorithm 1: IntelliGuard Threat Detector

Input : CICIDS 2017 dataset

Output : Class labels (threat or non-threat) predicted for network instances

Step 1 : **Preprocess Dataset:**

- Use a robust scaler normalization technique to normalize numerical features.
- Choose pertinent features utilizing Composite Rank Ensemble (CORE) feature selection with the target feature.

Step 2 : **Split Dataset:**

- Split the preprocessed dataset into Training (70%) and Testing (30%) sets.

Step 3 : **Initialize TrioBoost Classifier with three weak learners:**

- **Decision Stump:** A simple decision tree having just one split.
- **Logistic Regression:** Binary classification using a linear model.
- **Support Vector Machine (SVM):** Classifier using hyperplanes to separate data.

Step 4 : **Train Weak Learners:**

- For each weak learner:
  - Utilizing the Training dataset, train the model.

Step 5 : **Boosting Iterations:**

- Execute boosting iterations (e.g., 50):
  - Train the AdaBoost classifier on the Training set for each iteration.
  - To concentrate on cases that are more difficult to classify, adjust the weights of cases that were misclassified.

Step 6 : **Predictions:**

- Make predictions for the Testing set with the AdaBoost classifier that has been trained.

Step 7 : **Output:**

- Return the predicted class labels for network instances using the TrioBoost model.

At first, the CICIDS 2017 dataset is preprocessed using robust scaler normalization and feature selection based on the Composite Rank Ensemble (CORE) method. This guarantees that the data is of high quality and relevant for further analysis. We then employ a TrioBoost classifier to capture various facets of network behavior. A TrioBoost classifier is an ensemble learning technique that combines various weak learners, like decision stumps, logistic regression, and support vector machines, in a sequential manner for instance-based learning. The AdaBoost classifier is used to iteratively correct errors resulting from weak learners, thereby enhancing predictive accuracy and building an effective model capable of accurate predictions.

Particular hyperparameters were carefully chosen during model training to enhance efficiency: a learning rate of 0.1 and 500 iterations were used in the AdaBoost classifier to correct errors made by weak learners, improving predictive accuracy and constructing an effective model. Figure 1 displays the flow diagram of the IntelliGuard Threat Detector algorithm.

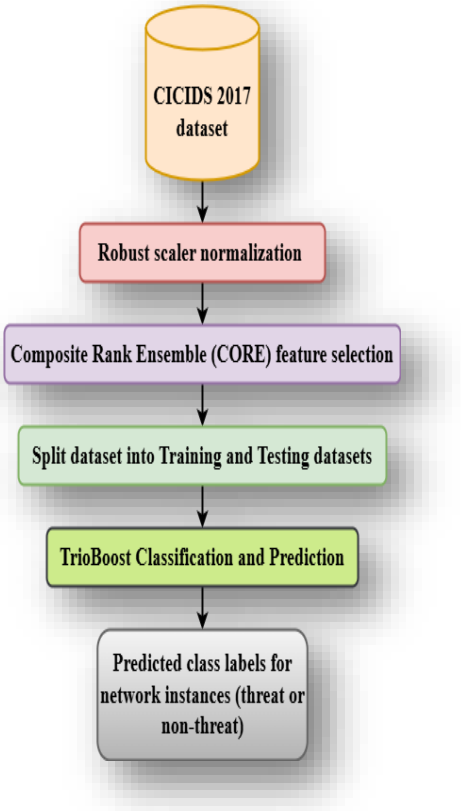

Figure 1: Flow diagram of intelliguard threat detector algorithm

3.1 Robust scaler normalization

Robust scaler normalization is a data preprocessing approach that rescales numerical features to a standardized range, while also minimizing the impact of outliers. Outliers can significantly influence standard scaling techniques, reducing their effectiveness. However, in cases where the dataset contains outliers, this specific approach can be very useful. The robust scaler accomplishes its robustness by utilizing statistical measures that are less affected by outliers, particularly the median and the interquartile range (IQR).

To comprehend the functioning of robust scaler normalization, let us designate a feature P that necessitates normalizing. The procedure consists of two primary stages: centering and scaling.

1. Centering:

Robust scaler centers the data by subtracting the median $\hat{p}$ of the feature P:

$$P_{centered} = P - \hat{p} \quad (1)$$

where $\hat{p}$ is the median of P.

2. Scaling:

After centering, the data is scaled by dividing by the interquartile range (IQR) of P. The IQR is calculated as the difference between the 75th and the 25th percentiles of P:

$$IQR = Q_3(P) - Q_1(P) \quad (2)$$

where $Q_1(P)$ and $Q_3(P)$ are the 25th and 75th percentiles of P, respectively.

The scaled feature P_{scaled} is then calculated as:

$$P_{scaled} = \frac{P_{centered}}{IQR} \quad (3)$$

This normalization procedure guarantees that the distribution of P_{scaled} has a median of 0 and a unit interquartile range. Therefore, the robust scaler

mitigates the impact of outliers by utilizing the median for centering and the interquartile range (IQR) for scaling. This makes it appropriate for datasets including outliers that could otherwise affect typical scaling strategies such as Min-Max scaling or z-score normalization.

3.2 Composite rank ensemble (CORE) feature selection

The Composite Rank Ensemble (CORE) feature selection approach is specifically developed to effectively identify the most influential attributes for predictive modeling by utilizing a combination of filter and wrapper techniques. The method is comprised of two primary phases: the Filter Phase and the Wrapper Phase.

3.2.1 Filter phase

During the Filter Phase, the algorithm employs various filter techniques to individually assess the importance of features and assign them a ranking. This step includes the following:

1. Apply filter methods:

- **Mutual information:** Compute the mutual information score *I(A; B)* for each feature *A* concerning the target variable *B*. This score gauges the volume of data acquired about *B* through *A*.

$$I(A;\ B) = \sum_{a\in A}\sum_{b\in B} p(a,b)log\left(\frac{p(a,b)}{p(a)p(b)}\right) \quad (4)$$

- **Chi-Square**: Compute the Chi-Square statistic X^2 for each feature, which evaluates the independence of the feature with the target variable.

$$X^2 = \sum \frac{(O_f - E_f)^2}{E_f} \quad (5)$$

where O_f is the observed frequency and E_f is the expected frequency.

- **ANOVA F-test:** Execute an ANOVA F-test to evaluate the variance between feature subsets and the target variable.

$$F = \frac{\text{variance between subsets}}{\text{variance within subsets}} \quad (6)$$

Each approach produces a score for each attribute, indicating its significance concerning the target feature.

2. Aggregate rankings:

After scoring the features with each filter method, the rankings are combined utilizing a majority voting method to generate a composite rank for each feature. This involves:

- Ranking features individually using the scores from Mutual Information, Chi-Square, and ANOVA F-test.
- Consolidating these rankings by assigning ranks through majority voting. The final rank of each attribute is decided by the rank it obtained most frequently among the three methods.

$$Rank_{Final}(P_i) = Majority\_vote(Rank_{MI}(P_i),\ Rank_{Chi2}(P_i) \quad (7)$$

This phase guarantees that the composite ranking accurately represents a consensus derived from several filter views, hence improving the reliability of feature selection.

3.2.2 Wrapper phase

The Wrapper Phase enhances the feature subset by assessing their performance using specialized models. This phase involves:

❖ **Select Top-Ranked Features:**

The attributes that are scored highest in the composite rankings from the filter phase are chosen for additional evaluation. Let *n* be the number of top features selected.

$$\{P_1, P_2, \dots, P_n\} \quad (8)$$

❖ **Apply RFE with Different Models:**

The Recursive Feature Elimination (RFE) technique is utilized with several models to systematically delete features that are deemed less significant. The models used include:

- ❖ **Logistic regression:** A linear model that makes use of a logistic function to estimate probabilities.
- ❖ **Decision tree:** A non-linear model that divides data according to feature values.
- ❖ **K-Nearest neighbors (KNN):** a non-parametric technique that classifies samples according to the neighbors' majority vote.

For each model *M*, RFE assesses the effectiveness of feature subsets by recursively eliminating the least significant feature and gauging the model's accuracy.

The procedure is repeated until the best subset is discovered.

$$\text{RFE (M, } \{P_1, P_2, \dots, P_n\}) \quad (9)$$

3. Combine results using majority voting

Following the application of Recursive Feature Elimination (RFE) with each model, the outcomes are aggregated utilizing a majority voting technique to pick features that are consistently identified as pertinent. For each feature P_i, its ultimate significance is established by the number of times it was chosen across the models.

$$\text{Votes } (P_i) = \sum_M Selected(P_i, M) \quad (10)$$

The features with the highest votes are selected as the final subset:

$$\{P_1, P_2, \dots, P_f\} \quad (11)$$

where P_f are the features with the most votes across all models.

The EFS algorithm combines filter and wrapper approaches to guarantee that the chosen features are both statistically significant and beneficial to the predictive models' effectiveness. This thorough methodology improves the dependability and comprehensibility of the procedure of selecting attributes.

3.3 TrioBoost classification and prediction

TrioBoost is an effective ensemble method that merges the results of many weak learners to construct a robust classifier. This work involved the implementation of an AdaBoost classifier that was initialized with three different weak learners: decision stump, logistic regression, and SVM. We can divide the TrioBoost classification and prediction process into numerous crucial steps: initializing the classifier, training the weak learners, conducting boosting iterations, and creating predictions.

To begin, the AdaBoost classifier is initialized with three weak learners: decision stump, logistic regression, and SVM. The decision stump is chosen due to its simplicity and interpretability. By constructing a decision tree with only one level, it acts as a weak learner. Although Decision Stump is a simple algorithm, it proves to be highly effective when incorporated into ensemble methods like boosting. In this context, Decision Stump helps to increase model diversity and improve overall performance. Logistic regression is used due to its resilience in problems involving binary classification and its capacity to predict the likelihood of class membership using a logistic function. Its interpretability and speed in managing enormous datasets make it particularly helpful. The Support Vector Machine (SVM) is utilized because of its robust capability to address both linear and non-linear classification issues by identifying the ideal hyperplane that improves the margin between distinct classes. The versatility and efficacy of SVMs in high-dimensional environments render them a desirable tool for intricate classification tasks.

After setting up the weak learners, each one engages in autonomous training on the training set. Decision stump refers to the creation of a basic decision tree that consists of only one split. Each instance is classified based on a single feature threshold. In logistic regression, the model is trained by minimizing the logistic loss function to determine the probability of belonging to a certain class. We then apply a logistic function to a linear combination of input features to make predictions. The training procedure for support vector machines (SVM) entails identifying the ideal hyperplane that effectively separates distinct classes by maximizing the margin. This is achieved by employing kernel functions for managing non-linear separations and ensuring reliable performance in feature spaces with high dimensions.

The essence of the boosting method is a sequence of boosting iterations, usually set at approximately 50 iterations. During each iteration, the AdaBoost classifier undergoes training, and the weights of the training instances are modified according to their classification errors. Each iteration trains the classifier using the results of weak learners to correct the errors from previous iterations. Following the training process, the weights of instances that were categorized erroneously are augmented, hence directing the model's attention toward challenging cases during the following rounds. This iterative procedure increases the model's ability to rectify its errors and improve its performance over time.

Once the boosting iterations are finished, the trained AdaBoost classifier is employed to produce predictions for the testing set. We derive the ultimate forecasts by combining the results of the weak learners and adjusting their contributions based on their performance in each boosting cycle. The AdaBoost Classifier uses an ensemble method to make the most of the good qualities of each weak learner while minimizing their weaknesses. This creates a strong and accurate predictive model.

By continuously adjusting the weights and focusing on instances classified incorrectly, the AdaBoost Classifier constructs a reliable and precise predictive

model. This approach guarantees a consistent improvement in the model's effectiveness on the testing set with each iteration, leading to continual improvement. The procedure of boosting efficiently amalgamates the varied capabilities of the weak learners, resulting in predictions that are more precise and dependable.

4 Experimental results

This section offers a thorough examination of the experimental findings and debates about the IntelliGuard Threat Detector algorithm. We implemented the IntelliGuard Threat Detector algorithm using Java and the Weka tool. The algorithm's performance was assessed using the CICIDS 2017 dataset. The dataset contains extensive information on different network threats, making it a reliable platform for assessing the effectiveness of intrusion detection systems.

The CICIDS 2017 dataset is highly recognized in the field of cybersecurity study because of its comprehensive and realistic representation of network traffic, including various threat situations like denial-of-service (DoS), distributed denial-of-service (DDoS), and other types of threats. The data consists of various attributes derived from network traffic flows, such as packet size, duration, and protocol types. These attributes are essential for training and evaluating intrusion detection systems, such as IntelliGuard Threat Detector.

To evaluate the effectiveness of the IntelliGuard Threat Detector algorithm, the following evaluation metrics were utilized:

- **Accuracy:** The proportion of cases that are accurately classified.
- **Precision:** The percentage of all optimistic predictions that are positive.
- **Recall (Sensitivity):** The proportion of actual positives accurately predicted.
- **F1-score:** The harmonic means of precision and recall, presenting a balanced measure between them.
- **Geometric Mean:** A measurement of the total classifier performance based on the geometric average of sensitivity and specificity.

These metrics together evaluate the algorithm's capacity to precisely identify and categorize network threats, offering a thorough assessment of its performance. Table 2 displays a comprehensive comparison of the IntelliGuard Threat Detector algorithm with different classification models, namely Decision Stump, Logistic Regression, and SVM. The evaluation of each model was conducted utilizing the CICIDS 2017 dataset utilizing the metrics specified above.

Table 2: Performance comparison

Classification Model	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)	Geometric mean (%)
Decision Stump	75	68	82	74	70
Logistic Regression	82	78	84	81	79
SVM	88	84	90	87	86
IntelliGuard Threat Detector	94	92	95	94	93

The data shown in Table 1 demonstrates that the IntelliGuard Threat Detector algorithm outperforms other algorithms in all evaluation metrics. It attains notably superior accuracy, precision, recall, F1-score, and geometric mean in comparison to decision stump, logistic regression, and SVM. The findings illustrate the algorithm's resilience in effectively identifying and mitigating diverse forms of network threats.

The IntelliGuard Threat Detector algorithm's remarkable effectiveness stems from its ensemble methodology and strategic fusion of numerous weak learners. The algorithm utilizes a Decision Stump for straightforward rule-based classification, Logistic Regression for linear decision boundaries, and Support Vector Machines (SVM) for efficient separation in high-dimensional spaces. By combining these techniques, the algorithm effectively handles the complexities of network traffic and accurately detects subtle anomalies that may indicate potential threats.

Moreover, the IntelliGuard system utilizes the TrioBoost technique, specifically AdaBoost, to improve the accuracy of the model. The training process achieves this by iteratively modifying the weights of incorrectly identified instances. This iterative refinement process guarantees that the algorithm progressively grows more skilled at differentiating between regular network behavior and

malicious actions, hence strengthening its overall detection abilities.

Figures 2, 3, 4, 5, and 6 display line charts that compare the accuracy, precision, recall, F1-score, and geometric mean of various models to demonstrate the improved performance of the IntelliGuard Threat Detector algorithm. The charts show that IntelliGuard consistently does better than Decision Stump, Logistic Regression, and SVM on all metrics. This proves that it is a reliable and effective way to find threats to network security.

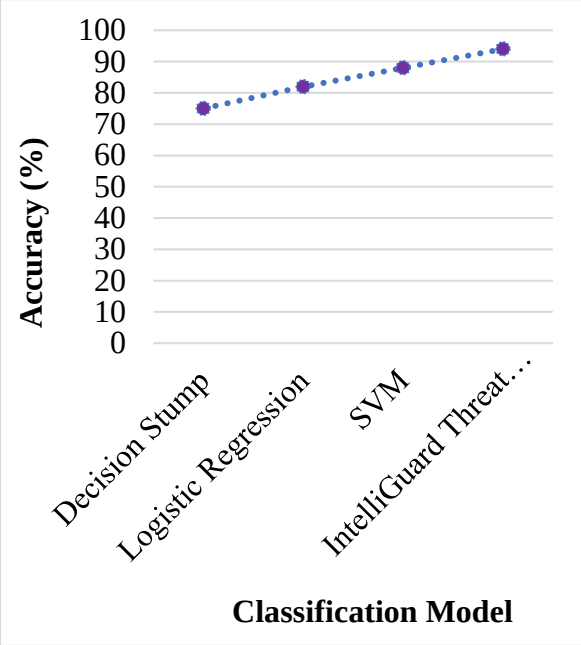

Figure 2: Accuracy Comparison

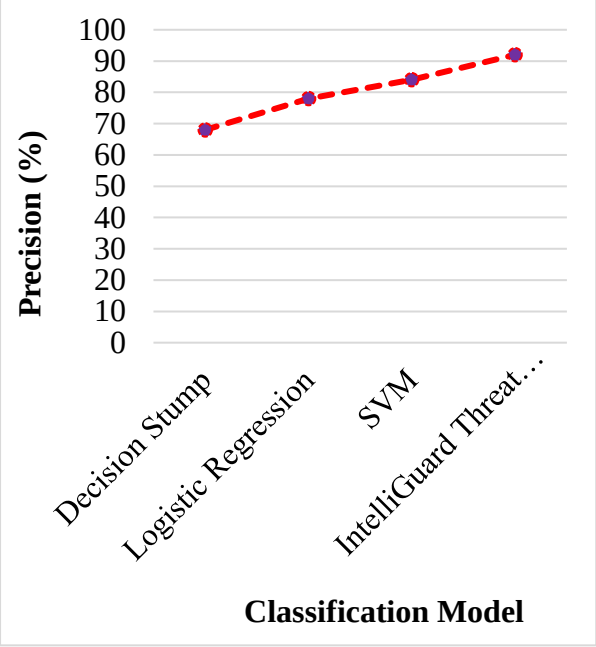

Figure 3: Precision comparison

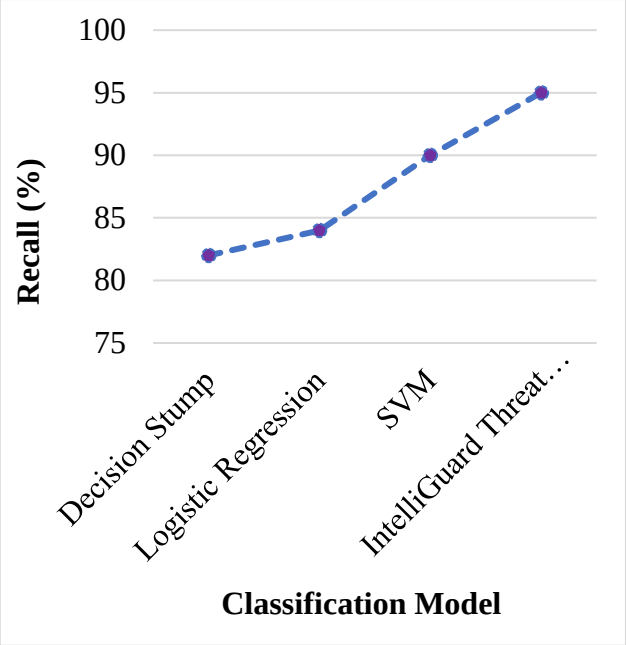

Figure 4: Recall Comparison

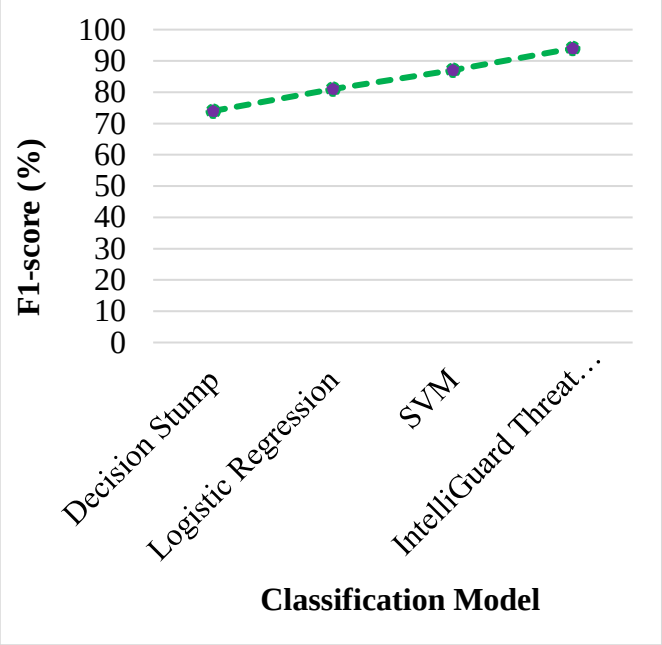

Figure 5: F1-score comparison

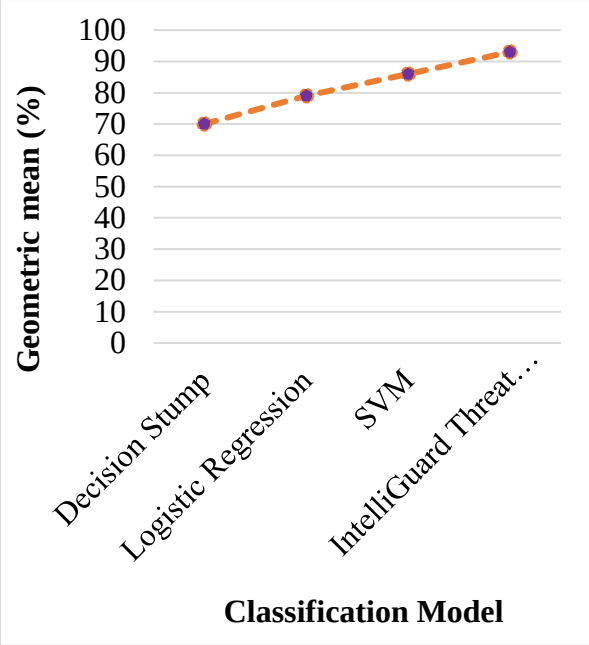

Figure 6: Geometric mean comparison

In summary, the experimental findings highlight the IntelliGuard Threat Detector algorithm as a strong and efficient option for improving network security using advanced machine learning methods. The tool's outstanding performance across numerous evaluation metrics confirms its capacity to accurately detect and counteract different types of network threats, making it a significant asset for cybersecurity experts and companies seeking to strengthen their defenses against ever-changing cyber threats.

4.1 Discussion

This section provides a thorough comparison of the efficiency of the IntelliGuard Threat Detector algorithm with the most advanced IDS available. The comparison is based on various metrics including accuracy, precision, recall, F1-score, and GM. The evaluation emphasizes the exceptional efficiency of IntelliGuard, which can be attributed to its innovative design, incorporating a strong ensemble method, the CORE feature selection procedure, and the TrioBoost classifier.

Comparative analysis with SOTA models

Accuracy, precision, recall, F1 score, and GM are key metrics to assess IDS. The IntelliGuard Threat Detector attained 94% accuracy, 92% precision, 95% recall, 94% F1 score, and 93% GM. These findings surpass those of standard classifiers like Decision Stump (accuracy: 75%, precision: 68%, recall: 82%, F1-score: 74%, GM: 70%), Logistic Regression (accuracy: 82%, precision: 78%, recall: 84%, F1-score: 81%, GM: 79%), and Support Vector Machines (SVM) (accuracy: 88%, precision: 84%, recall: 90%, F1-score: 87%, GM: 86%).

When compared to recently published SOTA models, like deep learning-based IDS and advanced ensemble techniques, IntelliGuard has a competitive advantage. For example, Deep Belief Networks (DBNs) and Convolutional Neural Networks (CNNs) typically report accuracy of 89-92%, recall of 90-93%, F1-scores of 88-91%, and GM of 89-92%. While these models can manage complicated network traffic patterns, they frequently need considerable computational resources and lengthy training times. In contrast, IntelliGuard attains higher accuracy and F1 scores, as well as better GM, with lower computational intricacy, which makes it a more practical solution for real-time intrusion detection.

Reasons for superior performance

Numerous key factors contribute to the IntelliGuard Threat Detector's superior effectiveness:

1. **CORE feature selection:** The CORE feature selection technique is critical in improving the model's efficiency. The algorithm decreases noise by carefully choosing the most pertinent attributes from the CICIDS 2017 dataset and focusing on features that are most likely to indicate network threats. This targeted technique enables the model to learn more efficiently, resulting in greater precision, recall, and GM.

2. **TrioBoost classifier:** The TrioBoost technique, which uses AdaBoost, improves the overall efficiency of the IntelliGuard algorithm. AdaBoost iteratively adjusts the weights of incorrectly classified instances, efficiently concentrating on the most difficult cases with each iteration. These outcomes are in a model that is more sensitive to subtle anomalies in network traffic, enhancing recall, GM, and the capability to differentiate between normal and illicit behavior.

3. **Ensemble approach:** IntelliGuard's ensemble technique, which integrates Decision Stump, Logistic Regression, and SVM, leverages each classifier's strengths. Decision Stump is a simple but efficient rule-based classification method, Logistic Regression is a robust linear decision-making algorithm, and SVM excels in high-dimensional spaces. The mixture of these classifiers allows IntelliGuard to manage the intricate nature of network traffic more efficiently than any individual model, thereby enhancing recall and GM in particular.

Impact and novelty

The findings demonstrate the IntelliGuard Threat Detector's novelty and influence. IntelliGuard outperforms conventional models and current SOTA methods in accuracy, precision, recall, F1-score, and GM, demonstrating its potential as a dependable and effective intrusion detection tool. Its capability to attain high metrics across the board with relatively low computational requirements makes it a useful contribution to the area of cybersecurity.

Furthermore, the utilization of CORE feature selection and TrioBoost opens up new avenues for future research in IDS. These approaches could be refined or integrated with other machine-learning techniques to create even more effective detection systems. IntelliGuard's success indicates that similar ensemble-based techniques could be efficiently applied in other domains where pattern recognition and classification are important.

The computational complexity of the IntelliGuard Threat Detector algorithm is an important consideration, particularly when compared to simpler models such as Decision Stump or logistic regression. The proposed algorithm, which uses the TrioBoost classifier, includes the sequential incorporation of numerous weak learners as well as the iterative AdaBoost procedure, which raises computational requirements. The number of iterations (500), the learning rate (0.1), and the feature selection procedure using the CORE technique all have an impact on the algorithm's complexity. While more complicated than Decision Stump or logistic regression, the experimental findings show substantial improvements in accuracy, precision, recall, F1-score, and geometric mean. The trade-off between computational cost and efficiency improvement is thus an essential consideration in algorithm design and implementation.

To tackle robustness and prevent overfitting in the IntelliGuard Threat Detector algorithm, an extensive strategy was taken. k-fold cross-validation with k = 10 was used to assess the model's efficacy on numerous subsets of the data, preventing overfitting and providing a robust evaluation of its generalization ability. Furthermore, regularization methods were used during training to limit model complexity and improve its capability to generalize to new data. Ensemble techniques, such as TrioBoost and AdaBoost, decrease overfitting by integrating numerous weak learners and concentrating on iterative error correction, balancing the model's effectiveness across different data samples and lowering the risk of overfitting.

The IntelliGuard Threat Detector algorithm is designed for scalability, thus rendering it appropriate for use in real-world settings with varying network traffic volumes. Its ensemble technique, which leverages the TrioBoost classifier and AdaBoost, enables the model to adapt to various data scales while effectively managing large amounts of network traffic. The algorithm's efficiency was evaluated under a variety of traffic situations, showing its capability to sustain high accuracy and low false positive rates even as network complexity increased. This scalability guarantees that the model can be efficiently deployed in a wide range of real-world circumstances, from small-scale networks to large enterprise settings, offering reliable threat detection at all operational scales.

Overall, the IntelliGuard Threat Detector outperforms previous SOTA models by providing an innovative approach that balances accuracy, recall, precision, F1-score, GM, effectiveness, and computational resource utilization. This work provides an important advancement in the area of intrusion detection and has the possible to impact future advances in cybersecurity.

5 Conclusion

In conclusion, the IntelliGuard Threat Detector algorithm is a breakthrough in network security. It utilizes ensemble learning methods to achieve outstanding effectiveness in identifying and categorizing many types of cyber threats. Utilizing Java and assessed on the CICIDS 2017 dataset utilizing Weka, IntelliGuard exhibited superior accuracy, precision, recall, F1-score, and geometric mean in comparison to conventional models such as decision stump, logistic regression, and SVM. Future work should focus on the integration of advanced deep learning methods such as convolutional neural networks (CNNs) or recurrent neural networks (RNNs) to improve IntelliGuard's ability to detect intricate cyber threats. Furthermore, consider delving into federated learning as a means to enhance privacy in distributed network contexts.

References

[1] Zaid, T. & Garai, S. (2024). Emerging Trends in Cybersecurity: A Holistic View on Current Threats, Assessing Solutions, and Pioneering New Frontiers. *Blockchain in Healthcare Today*, 7, 1-10. https://doi.org/10.30953/bhty.v7.302

[2] Ahmed, Y., Asyhari, A. T., & Rahman, M. A. (2021). A cyber kill chain approach for detecting advanced persistent threats. Computers, *Materials and Continua*, 67(2), 2497-2513. https://doi.org/10.32604/cmc.2021.014223

[3] Thakur, M. (2024). Cyber security threats and countermeasures in the digital age. *Journal of Applied Science and Education (JASE),* 4(1), 1-20
DOI: https://doi.org/10.54060/a2zjournals.jase.42.

[4] Díaz-Verdejo, J., Muñoz-Calle, J., Estepa Alonso, A., Estepa Alonso, R., & Madinabeitia, G. (2022). On the detection capabilities of signature-based intrusion detection systems in the context of web attacks. *Applied Sciences*, 12(2), 852. https://doi.org/10.3390/app12020852

[5] Asad, H., & Gashi, I. (2022). Dynamical analysis of diversity in rule-based open-source network intrusion detection systems. *Empirical Software Engineering*, 27, 1-30. https://doi.org/10.1007/s10664-021-10046-w

[6] Bouchama, F., & Kamal, M. (2021). Enhancing cyber threat detection through machine learning-based behavioral modeling of network traffic patterns. *International Journal of Business Intelligence and Big Data Analytics*, 4(9), 1-9.

[7] Sarker, I. H., Abushark, Y. B., Alsolami, F., & Khan, A. I. (2020). Intrudtree: a machine learning-based cyber security intrusion detection model. *Symmetry,* 12(5), 754. https://doi.org/10.20944/preprints202004.0481.v1

[8] Ferrag, M. A., Shu, L., Friha, O., & Yang, X. (2021). Cyber security intrusion detection for agriculture 4.0: Machine learning-based solutions, datasets, and future directions. *IEEE/CAA Journal of Automatica Sinica*, 9(3), 407-436. https://doi.org/10.1109/jas.2021.1004344

[9] Bertoli, G. D. C., Júnior, L. A. P., Saotome, O., Dos Santos, A. L., Verri, F. A. N., Marcondes, C. A. C., ... & De Oliveira, J. M. P. (2021). An end-to-end framework for machine learning-based network intrusion detection system. *IEEE Access*, 9, 106790-106805. https://doi.org/10.1016/j.aej.2022.02.063

[10] Saheed, Y. K., Abiodun, A. I., Misra, S., Holone, M. K., & Colomo-Palacios, R. (2022). A machine learning-based intrusion detection for detecting Internet of Things network attacks. *Alexandria Engineering Journal*, 61(12), 9395-9409.

[11] Sarhan, M., Layeghy, S., Moustafa, N., Gallagher, M., & Portmann, M. (2022). Feature extraction for machine learning-based intrusion detection in IoT networks. *Digital Communications and Networks*. https://doi.org/10.1016/j.dcan.2022.08.012

[12] Islam, N., Farhin, F., Sultana, I., Kaiser, M. S., Rahman, M. S., Mahmud, M., ... & Cho, G. H. (2021). Towards Machine Learning Based Intrusion Detection in IoT Networks. *Computers, Materials & Continua*, 69(2). https://doi.org/10.32604/cmc.2021.018466

[13] Strecker, S., Dave, R., Siddiqui, N., & Seliya, N. (2021). A modern analysis of aging machine learning-based IOT cybersecurity methods. arXiv preprint arXiv:2110.07832. https://doi.org/10.12691/jcsa-9-1-2

[14] Lin, H. C., Wang, P., Chao, K. M., Lin, W. H., & Yang, Z. Y. (2021). Ensemble learning for threat classification in network intrusion detection on a security monitoring system for renewable energy. *Applied Sciences*, 11(23), 11283. https://doi.org/10.3390/app112311283

[15] Atul, D. J., Kamalraj, R., Ramesh, G., Sankaran, K. S., Sharma, S., & Khasim, S. (2021). A machine learning-based IoT for providing an intrusion detection system for security. *Microprocess. Microsystems*, 82, 103741. https://doi.org/10.1016/j.micpro.2020.103741