UMYU Scientifica, Vol. 4 NO. 2, June 2025, Pp 142 – 157
https://doi.org/10.56919/usci.2542.017
ISSN: 2955 – 1145 (print); 2955 – 1153 (online)
A periodical of the Faculty of Natural and Applied Sciences, UMYU, Katsina

ORIGINAL RESEARCH ARTICLE

Enhanced DDoS Attack Detection in IoT Environments Using Voting and Stacking Ensemble Learning: Implementation and Performance Analysis

Abdulrahman Tunde Alabelewe[1,3], Muhammad Aminu Ahmad[2]*, Mohammed Ibrahim[4], Ahmed Abubakar Aliyu[2], Abubakar Muazu Ahmed[2], Saadatu Abdulkadir[2]

[1]Department of Informatics, Kaduna State University, Kaduna, Nigeria
[2]Department of Secure Computing, Kaduna State University, Kaduna, Nigeria
[3]Department of Cyber Security, Airforce Institute of Technology, Kaduna, Nigeria
[4]Department of Cyber Security, Nigerian Defence Academy, Kaduna, Nigeria

ARTICLE HISTORY

Received March 10, 2025
Accepted June 10, 2025
Published June 11, 2025

KEYWORDS

Internet of Things, DDoS attacks, ensemble learning, machine learning, voting classifier, stacking classifier, cybersecurity, network security, IoT Security.

ABSTRACT

Due to the increase in the adoption of Internet of Things (IoT) devices, there has been a significant increase in Distributed Denial of Service (DDoS) attack. This is because IoT devices have introduced significant security vulnerabilities, thereby increasing the attack surface. This paper aims to present an enhanced DDoS Attack detection in an IoT environment using an ensemble learning approach. This was achieved by implementing the voting and stacking classifiers that combine four supervised learning algorithms: Random Forest, Decision Trees, Logistic Regression, and K-Nearest Neighbors. Using the comprehensive CIC-IoT2023 dataset, the results of the test conducted indicate outstanding performance, with the voting classifier achieving 99.39% accuracy (190.9872ms inference time, 32 false positives) and the stacking classifier reaching 99.40% accuracy (224.9587ms inference time, 89 false positives), a 5-fold stratified cross-validation was conducted which validated the models' robustness as a significant improvement on previous study conducted in this area.

INTRODUCTION

The adoption of the Internet of Things has led to transformation in various sectors of the economy, such as smart healthcare, smart agriculture, smart industries, and smart homes. In 2023, it was estimated that about 15.9 billion IoT devices were connected globally, and it is projected to reach 25.4 billion by 2030 (Fischer, 2023). This development has also raised new security concerns because IoT devices have limited processing power, inadequate safety measures, and differing architectures. These vulnerabilities have made IoT devices soft targets for cybercriminals to launch various types of cyber-attacks (Kandasamy et al., 2020)

Among the various cyber threats that IoT is prone to, Distributed Denial of Service (DDoS) attacks pose one of the most crucial risks to IoT ecosystems. In 2016, the Mirai botnet attack exploited vulnerable IoT devices, compromising these devices to serve as botnets and used to initiate one of the largest DDoS attacks in history, displaying the scale of this threat (Abughazaleh et al., 2020). Traditional security systems often prove inadequate or difficult to implement in IoT environments due to resource limitations and the evolving nature of attack patterns (Malhotra et al., 2021).

Previous studies have shown that traditional machine-learning methods can improve IoT security. These recent studies have shown that machine learning algorithms such as Decision Trees (DT), Random Forest (RF), K-Nearest Neighbors (KNN), and Logistic Regression (LR) are among those that are very effective in predicting DDoS attacks (Butun et al., 2020). However, implementing these machine learning algorithms as stand-alone has challenges in keeping up with the ever-evolving attack patterns and establishing the balance between detection accuracy and computational limitations that are inherent in IoT devices (Luo et al., 2021).

In order to find a potential solution for these issues, Ensemble learning techniques have been proposed by previous studies (Bin Sarhan & Altwaijry, 2023; Khan et al., 2023). Ensemble learning techniques combine several machine learning models to improve predictive performance. It can overcome the resource limitations present in IoT systems while achieving higher detection accuracy by utilizing the supplementary characteristics of various algorithms combined with the technique (Ahmed & Khan, 2023).

Correspondence: Muhammad Aminu Ahmad. Department of Secure Computing, Kaduna State University, Kaduna, Nigeria. ✉ muhdaminu@kasu.edu.ng.

How to cite: Alabelewe, A. T., Ahmad, M. A., Aliyu, A. A., Ibrahim, M., Ahmed, A. M., & Abdulkadir, S. (2025). Enhanced DDoS Attack Detection in IoT Environments Using Voting and Stacking Ensemble Learning: Implementation and Performance Analysis. *UMYU Scientifica*, 4(2), 142 – 157. https://doi.org/10.56919/usci.2542.017

Ensemble learning has emerged as a robust approach for improving Distributed Denial of Service (DDoS) attack detection in IoT systems by combining multiple models to achieve superior performance while maintaining computational efficiency (Elliott & Anderson, 2023). This methodology operationalizes the "wisdom of crowds" principle through three key mechanisms: bias reduction via model aggregation (Yang et al., 2023), variance mitigation through prediction averaging (Mishra & Paliwal, 2023), and improved operational stability across diverse network conditions (Shtayat et al., 2023). The approach has proven particularly valuable in IoT security applications where traditional single-model solutions often struggle with adaptability and resource constraints.

Four primary ensemble methods have demonstrated significant effectiveness in IoT security contexts. Bagging techniques, exemplified by Random Forest algorithms, reduce variance through bootstrap aggregation of multiple decision trees (Bin Sarhan & Altwaijry, 2023). Boosting methods like AdaBoost and Gradient Boosting sequentially improve model performance by focusing on misclassified instances (Golchha et al., 2023). Voting classifiers combine predictions through either hard voting (majority decision) or soft voting (probability averaging), effectively reducing both false positives and negatives in security applications (Mushtaq et al., 2022; Taha, 2021). More advanced stacking methods employ a meta-learner to optimally integrate base model predictions, typically achieving higher accuracy than individual models, as Jegede (2023) demonstrated, who reported 93.8% detection accuracy compared to 89.1% for the best single model.

Recent research has yielded substantial improvements in detection capabilities through various ensemble approaches. Abu Al-Haija & Al-Dala'ien (2022) proposed specialized solutions named the ELBA-IoT framework, designed specifically for resource-constrained environments and it achieved 99.2% accuracy with only 1.7ms inference time on the Bot-IoT dataset, while newer frameworks like Deep Squeezed-Boosted Ensemble Learning (98.50% accuracy on IOT_Malware) and multi-algorithm integrations (98.63% accuracy on TON-IoT) continue to push performance boundaries (Ahmed & Khan, 2023; Alotaibi & Ilyas, 2023).

Recent advancements in ensemble learning have also significantly enhanced DDoS detection capabilities in IoT environments. Mante and Kolhe (2024) demonstrated that tree-based ensemble models, particularly stacking with XGBoost, can achieve high detection accuracy, reaching 99.30% on the CIC-IoT2023 dataset. Ye et al. (2024) proposed a behavioral-feature-based stacking model to identify low-rate DDoS attacks, achieving precision, recall, and F1-scores of 0.96, with a 99% recognition rate for LDDoS traffic. In parallel, Ain et al. (2025) introduced a deep-learning hybrid model combining CNNs, LSTMs, and Autoencoders, which achieved an accuracy of 96.78% on the same dataset. These studies highlight the growing effectiveness of ensemble and hybrid models in managing the complexity and evolving patterns of DDoS threats in modern IoT networks, particularly when applied to realistic and diverse traffic scenarios like those in the CIC-IoT2023 dataset

Implementation considerations for ensemble IoT security methods require carefully balancing several factors. The accuracy-complexity tradeoff presents a key challenge, as stacking methods provide superior accuracy but demand greater computational resources compared to simpler voting ensembles (Golchha et al., 2023). This has led to the development of hybrid approaches that combine elements of multiple techniques to optimize both performance and efficiency (Abbas et al., 2022). Benchmark evaluations for newer datasets like CIC-IoT2023 are establishing critical baseline metrics for future research (Jony & Arnob, 2024).

Although previous studies have explored machine learning models for DDoS detection, the challenge remains in overfitting that prevents effective identification of new attack patterns, difficulty balancing precision and recall, and excessive computational demands that exceed IoT device capabilities. This study introduces ensemble learning, specifically voting and stacking classifiers, to overcome these challenges by reducing the variance and overfitting, integrating complementary algorithms to better balance false positives and false negatives, and employing dimensionality reduction to minimize resource requirements within IoT constraints.

Therefore, this study uses ensemble learning techniques to propose an improved method for detecting DDoS attacks in IoT systems. We apply and assess two ensemble approaches, namely voting and stacking classifiers, which incorporate four supervised learning algorithms: Random Forest, Decision Trees, Logistic Regression, and K-Nearest Neighbors, building on the earlier work by Jony & Arnob (2024). For this research, we use the CIC-IoT2023 dataset, a real-time dataset and benchmark for large-scale attacks in IoT Environment (Neto et al., 2023).

The selection of Random Forest (RF), Decision Tree (DT), Logistic Regression (LR), and K-Nearest Neighbors (KNN) as base learners in the ensemble framework is guided by their complementary learning mechanisms, computational efficiency and interpretability in the context of IoT-driven DDoS detection. RF and DT, as tree-based algorithms, offer robustness in handling high-dimensional and nonlinear data, with RF providing strong generalization through bootstrap aggregation and DT ensuring fast decision-making suitable for constrained environments (Amro et al., 2021; Brophy & Lowd, 2021). LR, a probabilistic linear classifier, brings computational simplicity and is particularly effective in binary classification tasks like DDoS detection (Okoye & Hosseini, 2024). KNN adds instance-based reasoning, enabling the ensemble to capture local decision boundaries and improve multi-class differentiation (Halder et al., 2024). Collectively, these models provide a diverse hypothesis space that enhances the ensemble's ability to balance detection accuracy, inference speed, and adaptability in resource-constrained IoT scenarios.

Support Vector Machines (SVM) and Gradient Boosting techniques such as XGBoost or LightGBM, while powerful, were excluded due to their higher computational complexity and memory requirements, which are often unsuitable for real-time inference in IoT environments (Yilmaz et al., 2021). SVM, in particular, has scalability issues with large datasets and requires kernel tuning, which increases training time and limits deployment feasibility (Hosseinzadeh et al., 2021). Though effective at reducing bias, Gradient Boosting algorithms rely on sequential model training, which introduces latency and demands more processing power, making them less ideal for lightweight, real-time DDoS detection systems (Bentéjac et al., 2021). In contrast, the chosen base models support parallelization, lower memory footprints, and faster inference, aligning better with the practical constraints and deployment requirements of IoT-based security architectures.

METHODOLOGY

This section describes our approach to use ensemble learning techniques to improve DDoS attack detection in IoT contexts. We describe the dataset, preprocessing methods, and voting and stacking classifier implementation in detail.

Dataset Description

We utilized the CIC-IoT2023 dataset (Neto et al., 2023), developed through a collaborative effort between the Canadian Institute for Cybersecurity (CIC) and the Information Technology University of Copenhagen (ITU). This dataset was generated in a smart home environment incorporating 20 distinct IoT devices including cameras, thermostats, smart TVs, and smart watches, as illustrated in Figure 1.

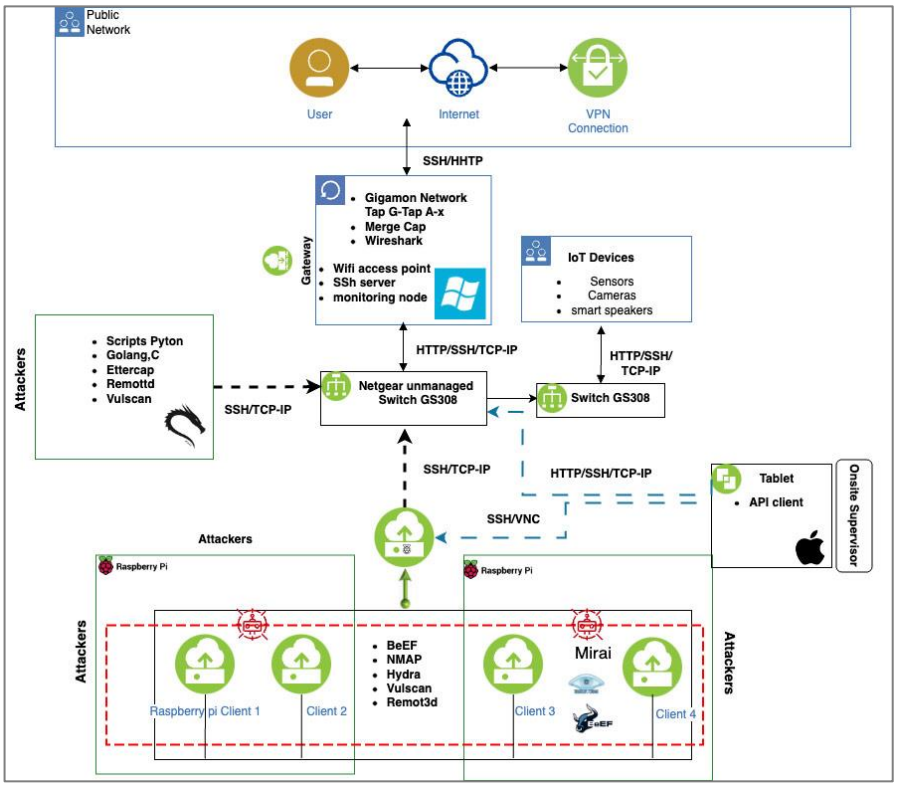


Figure 1. CIC-IoT2023 dataset topology showing the network setup and attack vectors.

The dataset comprises approximately 80 million packets collected over ten days (five days each of normal and attack traffic), with 64 million classified as malicious and 16 million as normal traffic. Each packet is characterized by 115 features, including protocol information, payload size, timestamp, and source and destination IP addresses.

What distinguishes the CIC-IoT2023 dataset from other datasets is its use of actual IoT devices as both attackers and victims, moving beyond simulation approaches to capture authentic device behaviors. The dataset incorporates ten distinct types of DDoS attacks: TCP SYN Flood, UDP Flood, HTTP Flood, HTTP Slow Post, Slowloris, MQTT Flood, CoAP Flood, WS-DDoS (WebSocket), Web Service Flood (SOAP), and Web Service Flood (RESTful).

Data Preprocessing

Our preprocessing workflow comprised four essential steps:

1. **Data Cleaning**: We removed irrelevant columns such as 'Unnamed: 0' that were artifacts of the data collection process. Missing or inconsistent values were handled using median imputation, chosen over mean imputation due to its robustness against outliers.

2. **Label Encoding**: Categorical features were transformed into numerical representations using Label Encoding. Normal traffic was labeled as 0, while attack traffic was labeled as 1, as shown in Fig. 2.

```
labels = {"BenignTraffic": 0, "Attack": 1}
data["label"] = data["label"].map(labels)
data.head()
```

Figure 2. Data encoding process shows the transformation of categorical variables into numerical format.

3. **Data Standardization and Normalization**: Features were standardized to have a mean of zero and a standard deviation of one, ensuring all features contributed equally to the model's decision-making process. Fig. 3 illustrates this process.

```
attack = 191412
benign = 222131

attack_df = data[data['label'] == 1]
benign_df = data[data['label'] == 0]

attack_sampled = attack_df.sample(n=attack, random_state=42)
benign_sampled = benign_df.sample(n=benign, random_state=42)

data = pd.concat([attack_sampled, benign_sampled])

data = data.sample(frac=1, random_state=42).reset_index(drop=True)
```

Figure 3: The data normalization process shows the transformation of features to a standard scale.

4. **Principal Component Analysis (PCA)**: To improve computational efficiency and mitigate the risk of overfitting while maintaining most of the informative structure of the dataset, Principal Component Analysis (PCA) was employed as a dimensionality reduction technique. PCA transforms the original high-dimensional data into a lower-dimensional space by projecting it onto a set of orthogonal axes (principal components) that capture the maximum variance present in the data.

   Figure 4 illustrates the cumulative explained variance as a function of the number of principal components. This plot helps determine how many components are needed to retain a desired proportion of the dataset's total variance. Initially, the curve rises steeply, indicating that the first few components capture a large share of the variance. However, as more components are added, the marginal gain in explained variance decreases, leading to a "flattening" of the curve—a phenomenon often referred to as the "elbow point."

   In this analysis, the elbow point occurs around the 35th principal component. Selecting 35 components allows the model to preserve nearly 100% of the original variance (as indicated by the plateau in the curve), while reducing the dimensionality from the original feature space. This balance ensures that the reduced dataset retains most of its discriminative power, thereby enabling efficient learning and inference, especially important in high-dimensional IoT traffic data.

   While PCA effectively reduced the feature space to 35 components, this transformation compromises interpretability and the ability to assess individual feature importance. PCA obscures the specific contributions of input features by converting original variables into orthogonal principal components, making it difficult to identify which network characteristics drive detection outcomes. This limits the transparency of the model, particularly in security-critical IoT contexts where understanding the influence of individual features is essential for informed decision-making. Although beneficial for computational efficiency and redundancy reduction, the reduced clarity in feature relevance remains a significant trade-off. By using 35 components, we significantly reduced dimensionality without substantial loss of information, which is crucial for accelerating training time and minimizing memory usage in real-time or resource-constrained IoT environments.

   The processed dataset was divided into training (70%) and testing (30%) subsets for model development and evaluation, as illustrated in Fig. 5.

5. **Hyperparameter Optimization:**
   A comprehensive hyperparameter tuning process was conducted for each base classifier to ensure optimal performance and model generalization using a combination of empirical experimentation and Grid Search with cross-validation. The goal was to balance classification accuracy with computational efficiency, particularly important in the context of IoT environments where real-time detection and limited resources are key constraints. Figure 6 below shows the optimization code.

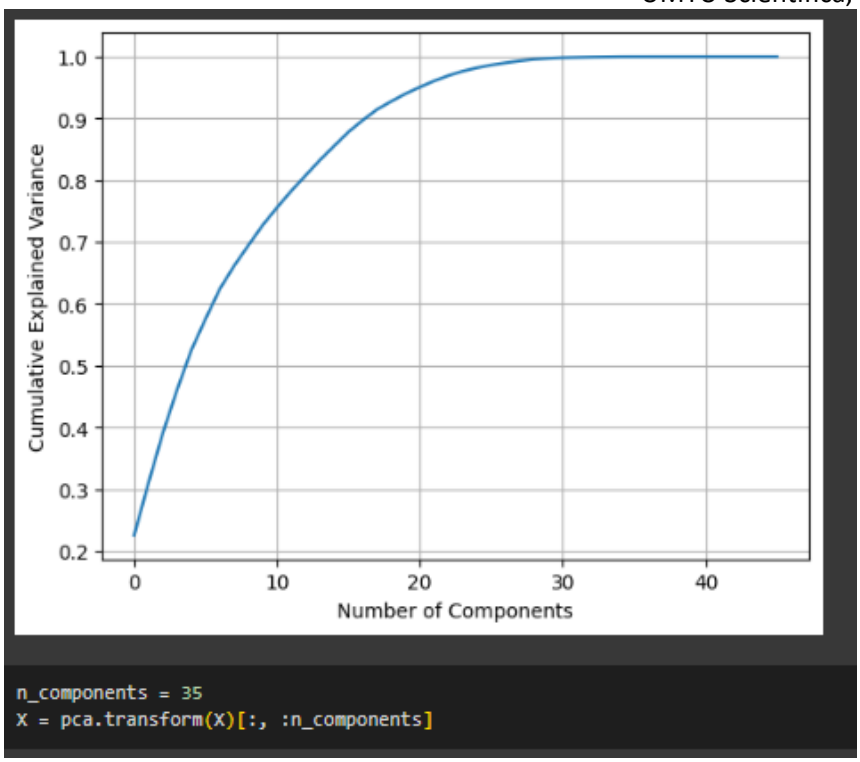


Figure 4. Cumulative explained variance plot for PCA, showing the relationship between the number of components and the preserved information.

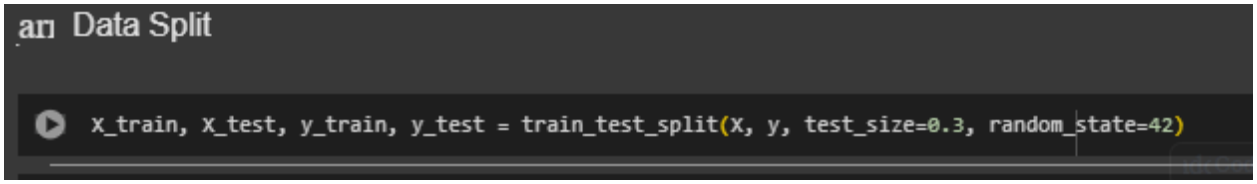
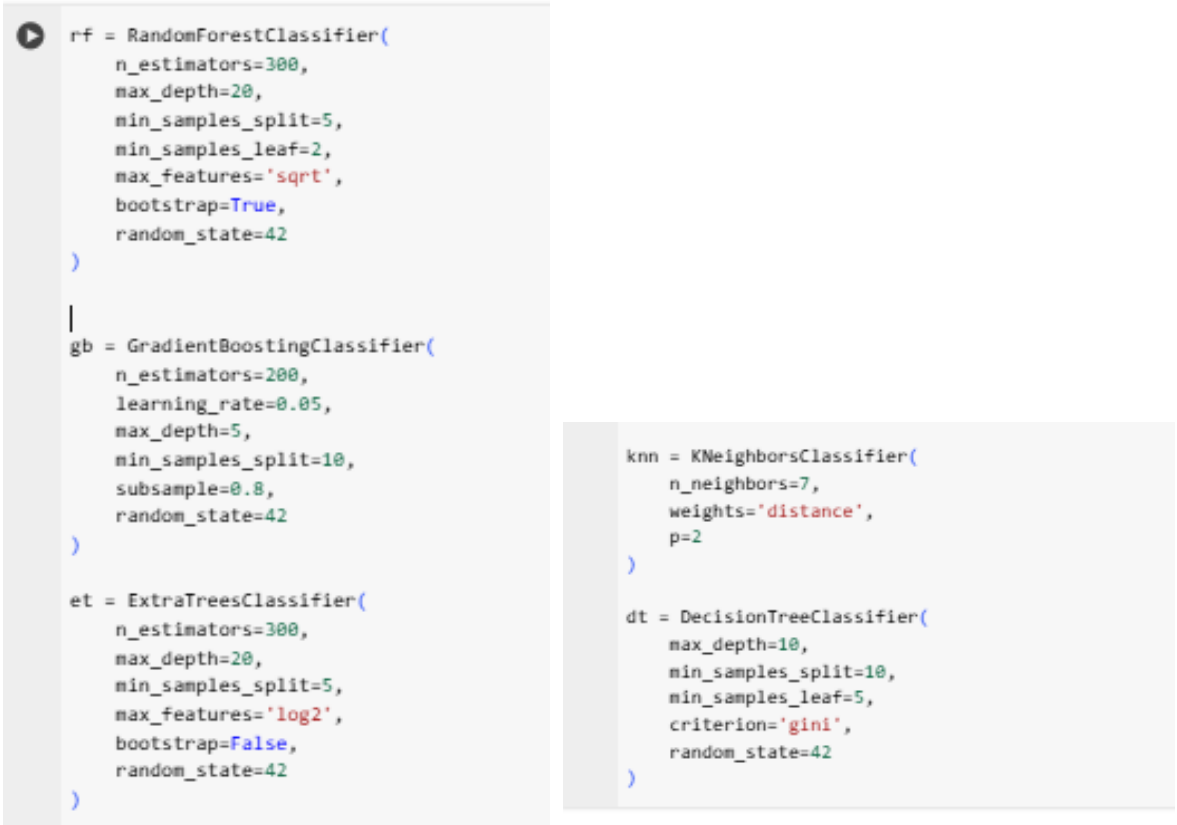


Figure 5. The data splitting process allocates 70% of the dataset for training and 30% for testing.



```
rf = RandomForestClassifier(
    n_estimators=300,
    max_depth=20,
    min_samples_split=5,
    min_samples_leaf=2,
    max_features='sqrt',
    bootstrap=True,
    random_state=42
)

gb = GradientBoostingClassifier(
    n_estimators=200,
    learning_rate=0.05,
    max_depth=5,
    min_samples_split=10,
    subsample=0.8,
    random_state=42
)

et = ExtraTreesClassifier(
    n_estimators=300,
    max_depth=20,
    min_samples_split=5,
    max_features='log2',
    bootstrap=False,
    random_state=42
)
```

```
knn = KNeighborsClassifier(
    n_neighbors=7,
    weights='distance',
    p=2
)

dt = DecisionTreeClassifier(
    max_depth=10,
    min_samples_split=10,
    min_samples_leaf=5,
    criterion='gini',
    random_state=42
)
```

Figure 6. Code for Hyperparameter Optimization

- ***Random Forest (RF):***
  The Random Forest classifier was configured with 300 estimators (n_estimators=300) and a maximum depth of 20 (max_depth=20) to allow for deep, expressive trees without overfitting. A minimum of five samples per internal node (min_samples_split=5) and two per leaf (min_samples_leaf=2) was used to regularize the tree structure. The max_features='sqrt' parameter helps introduce diversity among the trees by limiting the number of features considered at each split, and bootstrap=True ensures randomness via sampling with replacement. This configuration strikes a balance between model accuracy and inference speed.
- ***Gradient Boosting (GB):***
  For the Gradient Boosting classifier, 200 trees were used (n_estimators=200) with a conservative learning rate of 0.05 to allow gradual learning (learning_rate=0.05). The tree depth was limited to 5 (max_depth=5) to prevent overfitting, while a higher minimum split threshold (min_samples_split=10) and a subsample ratio of 0.8 (subsample=0.8) introduced additional regularization. These settings promote model robustness and generalization, especially in scenarios with noisy traffic data.
- ***Extra Trees (ET):***
  The Extra Trees classifier was tuned to use 300 trees (n_estimators=300) and a maximum depth of 20 (max_depth=20) to exploit full tree growth while maintaining generalization with min_samples_split=5. Unlike Random Forest, bootstrap=False was used, relying on the entire dataset per tree. max_features='log2' encourages more aggressive feature reduction at each split, enhancing tree speed and decorrelation.
- ***K-Nearest Neighbors (KNN):***
  The KNN model used 7 neighbors (n_neighbors=7) to increase decision stability. weights='distance' gives greater importance to closer neighbors, improving classification accuracy in unevenly distributed data. The use of p=2 applies the Euclidean distance metric, which is effective in high-dimensional space post-PCA.
- ***Decision Tree (DT):***
  For the Decision Tree, a relatively shallow structure was enforced with max_depth=10 to avoid overfitting. Splitting required at least 10 samples (min_samples_split=10) and 5 per leaf (min_samples_leaf=5), promoting simpler, more interpretable trees. The criterion='gini' was selected for computational efficiency while preserving accuracy.

These optimized hyperparameters were selected based on cross-validated performance metrics and were integrated into ensemble models, including voting and stacking classifiers. This optimization process ensures that each model contributes effectively to the ensemble's overall performance, with particular attention paid to maintaining inference efficiency suitable for deployment in real-time IoT systems.

Proposed Ensemble Approach

Our proposed approach integrates four supervised learning algorithms—Random Forest (RF), Decision Tree (DT), Logistic Regression (LR), and K-Nearest Neighbors (KNN)—through two ensemble techniques: voting and stacking. Fig. 7 illustrates the architecture of our proposed system.

1) Voting Classifier

The voting classifier aggregates predictions from the four base models through a democratic process. We implemented a soft voting mechanism, where the final classification is determined by averaging the predicted probabilities from each model and selecting the class with the highest average probability. This approach leverages diverse algorithms' complementary strengths while mitigating their weaknesses. The implementation involved the following configuration:

2) Stacking Classifier

The stacking classifier introduces a meta-learner that learns to optimally combine the predictions of the base models. We configured the stacking architecture with the same four base models (RF, DT, LR, KNN) and a Logistic Regression meta-learner. This hierarchical approach enables the system to capture complex relationships between the outputs of the base models, potentially achieving higher detection accuracy. The implementation involved the following configuration:

Evaluation Metrics

We evaluated the performance of our ensemble models using several metrics:

1. **Accuracy**: The proportion of correct predictions (both true positives and true negatives) among the total number of cases examined.

$$\textbf{\textit{Accuracy}} = \frac{TP + TN}{TP + TN + FP + FN} \quad (3.1)$$

2. **Precision**: The proportion of true positive predictions among all positive predictions made by the model.

$$\textbf{\textit{Precision}} = \frac{TP}{TP + FP} \quad (3.2)$$

3. **Recall**: The proportion of true positive predictions among all actual positive instances in the data.

$$\textbf{Recall} = \frac{TP}{TP + FN} \quad (3.3)$$

4. **F1-Score**: The harmonic mean of precision and recall, providing a balance between the two metrics.

$$\textbf{F1-Score} = 2 * \frac{Precision * Recall}{Precision + Recall} \quad (3.4)$$

5. **Inference Time**: The time the trained model requires to make predictions on new data, a critical metric for real-time detection systems.

Additionally, we analyzed the confusion matrices to gain deeper insights into the error patterns of each ensemble method, examining the distributions of true positives, true negatives, false positives, and false negatives.

RESULTS AND ANALYSIS

This section presents the experimental results of our ensemble learning approach for DDoS detection in IoT environments. We evaluate the performance of both voting and stacking classifiers and compare them with previous research.

Voting Classifier Performance

The voting classifier demonstrated exceptional performance in distinguishing between normal and attack traffic. Table 1 presents the detailed classification metrics for this ensemble method.

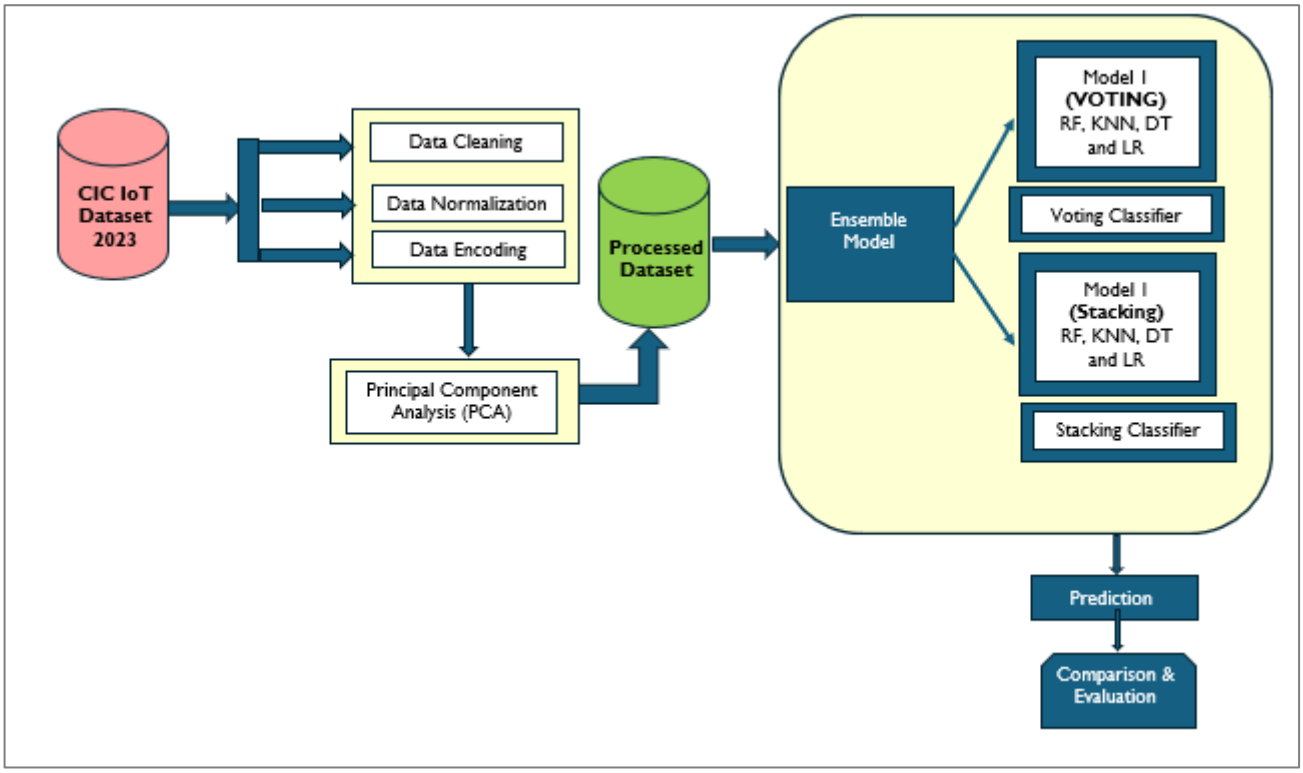


Figure 7. Architecture of the proposed ensemble-based DDoS detection system.

```
voting_clf = VotingClassifier(
    estimators=[
        ('rf', RandomForestClassifier(n_estimators=100, random_state=
        ('dt', DecisionTreeClassifier(random_state=42)),
        ('lr', LogisticRegression(max_iter=1000, random_state=42)),
        ('knn', KNeighborsClassifier(n_neighbors=5))
    ],
    voting='soft'
)
```

Figure 8. Voting Classifier Configuration.

```
stacking_clf = StackingClassifier(
    estimators=[
        ('rf', RandomForestClassifier(n_estimators=100, random_state=
        ('dt', DecisionTreeClassifier(random_state=42)),
        ('lr', LogisticRegression(max_iter=1000, random_state=42)),
        ('knn', KNeighborsClassifier(n_neighbors=5))
    ],
    final_estimator=LogisticRegression(random_state=42)
)
```

Figure 9. Stacking Classifier Configuration.

Table 1. Performance Metrics of Voting Classifier

Class	Precision	Recall	F1-score	Support
Normal (0)	0.99	1.00	0.99	66,718
Attack (1)	1.00	0.99	0.99	57,345
Macro Avg	0.99	0.99	0.99	124,063
Weighted Avg	0.99	0.99	0.99	124,063

The voting classifier achieved an overall accuracy of 99.39%, with balanced performance across both normal and attack traffic categories. For normal traffic (Class 0), the model achieved a precision of 0.99 and a perfect recall of 1.00, resulting in an F1-score of 0.99. Similarly, for attack traffic (Class 1), the model demonstrated a perfect precision of 1.00 and a recall of 0.99, maintaining an F1-score of 0.99.

Figure 10 presents the confusion matrix for the voting classifier, providing deeper insights into its classification performance.

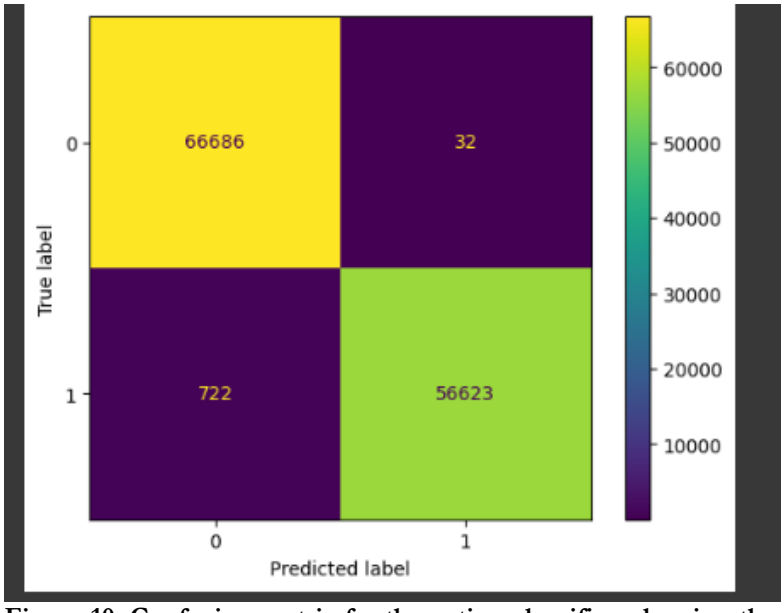


Figure 10. Confusion matrix for the voting classifier, showing the distribution of true positives, true negatives, false positives, and false negatives.

The confusion matrix reveals 66,686 true negatives (correctly identified benign traffic) and 56,623 true positives (accurately detected attack instances). The false positives (benign instances misclassified as attacks) are minimal at 32, indicating a low false alarm rate. The false negatives, where attacks are incorrectly classified as benign, are relatively low at 722.

The false positives, which are the benign traffic misclassified as attacks, could lead to wasted computational effort or unwarranted service throttling on

IoT devices, while the false negatives, which are the attack traffic undetected as benign, pose the greatest security risk by allowing malicious flows to bypass defenses in the IoT environment. Balancing these errors is critical because false positives in IoT contexts can trigger unnecessary alerts, cause redundant network filtering rules, and drain device resources through unwarranted defensive actions. False negatives can be more critical, as undetected attacks can compromise device integrity, disrupt services, and facilitate lateral movement within an IoT environment.

Stacking Classifier Performance

The stacking classifier demonstrated similarly impressive performance metrics, as detailed in Table 2.

The stacking classifier achieved an overall accuracy of 99.40%, with consistent performance across both traffic categories. The precision, recall, and F1-score values match those of the voting classifier, demonstrating the robustness of ensemble approaches for this detection task. Figure 11 presents the confusion matrix for the stacking classifier.

The confusion matrix shows 66,629 true negatives and 56,749 true positives. The stacking classifier produced 89 false positives and 596 false negatives, reflecting a slightly different error distribution compared to the voting classifier.

Although the Stacking Classifier has a slightly higher false positive rate, it substantially reduces false negatives, which is more desirable in practical IoT environments where security risks from undetected attacks outweigh the cost of occasional false alarms. In contrast, the Voting Classifier offers slightly better precision but may leave more threats undetected, potentially compromising system resilience in high-stakes IoT applications. Therefore, the Stacking Classifier strikes a better balance for critical security contexts

Table 2. Performance Metrics of Stacking Classifier

Class	Precision	Recall	F1-score	Support
Normal (0)	0.99	1.00	0.99	66,718
Attack (1)	1.00	0.99	0.99	57,345
Macro Avg	0.99	0.99	0.99	124,063
Weighted Avg	0.99	0.99	0.99	124,063

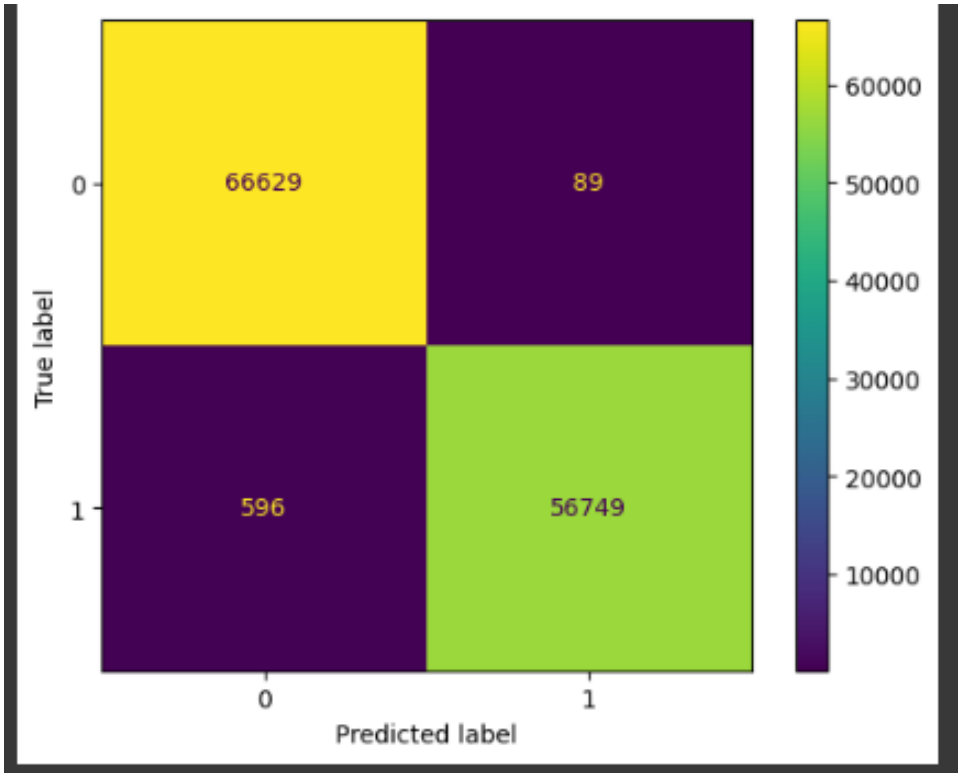


Figure 11. Confusion matrix for the stacking classifier, showing the distribution of true positives, true negatives, false positives, and false negatives.

Inference Time Analysis

Inference time is critical for real-time detection systems, particularly in resource-constrained IoT environments. Fig. 12 compares the inference times of both ensemble methods.

The voting classifier exhibited an inference time of 190.9872 milliseconds, while the stacking classifier required 224.9587 milliseconds. This difference of approximately 34 milliseconds reflects the increased computational complexity of the stacking architecture, which employs a meta-learner for prediction generation.

In evaluating the reported inference times of 190–225 milliseconds (ms) for ensemble classifiers in IoT environments, it is essential to benchmark these figures against industry standards to assess their suitability for real-time applications.

Industry benchmarks indicate that acceptable inference latency varies depending on the specific application and its real-time requirements. For instance, in latency-sensitive applications such as autonomous vehicles or industrial automation, end-to-end latency requirements are stringent; a latency of 100 milliseconds or less is often considered essential(Sumaiya et al., 2024). However, latency requirements are more relaxed for applications like smart surveillance or environmental monitoring, with acceptable inference times ranging from 100 to 300 ms (Sumaiya et al., 2024).

Given this context, the reported inference times of 190–225 ms are within acceptable limits for non-critical IoT applications but may be inadequate for time-sensitive scenarios. Therefore, while the ensemble classifiers demonstrate promising accuracy, their deployment should be carefully considered in the context of the specific latency requirements of the intended IoT application. Further optimization may be necessary to meet the stringent latency demands of critical real-time systems.

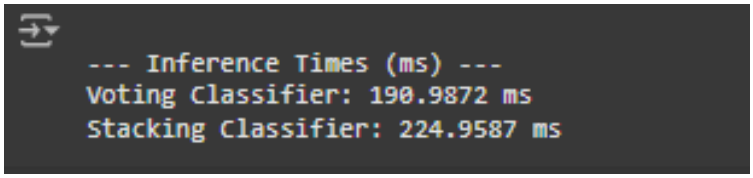


Figure 12. Comparison of inference times between voting and stacking classifiers.

Comparative Analysis

To evaluate the effectiveness of our ensemble approach, we compared the performance of both classifiers with previous research (Alotaibi & Ilyas, 2023; Jony & Arnob, 2024). Table 3 presents this comparative analysis.

Our ensemble methods demonstrate superior performance compared to both single-algorithm approaches on the same dataset (Jony & Arnob, 2024) and previous ensemble implementations on different datasets (Alotaibi & Ilyas, 2023). The voting classifier achieves a 0.20% improvement in accuracy over the best single-algorithm approach (Decision Tree), while the stacking classifier shows a 0.21% improvement. While these percentage improvements may appear modest, they represent significant enhancements in real-world applications where even small improvements in accuracy can substantially impact system security. Figure 14 visualizes the accuracy comparison between different approaches.

Table 3. Comparative Analysis of DDOS Detection Approaches

Method	Dataset	Accuracy	Precision	Recall	F1-score	Inference Time
Decision Tree	CIC-IoT2023	99.19%	0.99	0.99	0.99	N/A
Random Forest	CIC-IoT2023	99.16%	0.99	0.99	0.99	N/A
K-Nearest Neighbors	CIC-IoT2023	93.80%	0.94	0.94	0.94	N/A
Logistic Regression	CIC-IoT2023	82.75%	0.83	0.83	0.83	N/A
Ensemble (Stacking)	TON-IoT	98.63%	98.20%	98.60%	98.61%	N/A
Voting Classifier (Ours)	CIC-IoT2023	99.39%	0.99	0.99	0.99	190.99ms
Stacking Classifier (Ours)	CIC-IoT2023	99.40%	0.99	0.99	0.99	224.97ms

Comparative Evaluation on the CIC-IoT2023 Dataset

The findings of this study were compared with recent research that also used the CIC-IoT2023 dataset for DDoS detection in IoT environments. This is to allow us to assess how our ensemble methods perform relative to existing approaches on the same benchmark.

Mante & Kolhe (2024) evaluated tree-based and ensemble classifiers, including Decision Tree, Random Forest, Extra Trees, and XGBoost. Their stacking model, which used XGBoost as the base learner, achieved a classification accuracy of 99.30%. Our stacking classifier slightly outperforms this at 99.40% and shows a more favorable error distribution, particularly in reducing false negatives—an essential factor in preventing undetected attacks in IoT networks. Ye et al., (2025) focused on detecting low-rate DDoS (LDDoS) attacks, which are elusive and more difficult to identify. Their model achieved an accuracy, precision, recall, and F1-score of 0.96 using behavioral features and a stacking classifier. While effective for LDDoS detection, these results fall short of the 0.99 scores achieved by our models in precision, recall, and F1-score. However, their emphasis on behavioral traits offers a direction worth exploring to improve sensitivity to low-bandwidth threats.

Ain et al. (2025) proposed a hybrid deep learning model combining CNNs, LSTMs, and Autoencoders. Their model achieved 96.78% accuracy, demonstrating strong performance in identifying complex attack patterns. Still, its accuracy lags behind our ensemble methods, and its deep learning architecture likely incurs higher computational costs. For real-time or resource-constrained IoT settings, our models offer a more practical solution with significantly lower inference times—190 milliseconds for the voting model and 225 milliseconds for stacking—well within the acceptable range for non-critical applications.

Compared to other studies using the CIC-IoT2023 dataset, our ensemble classifiers show superior accuracy

and balanced error rates, with inference times that support real-time deployment in most IoT environments. While other models bring unique strengths, such as behavioral feature modeling or deep learning architectures, they either tradeoff accuracy or effectiveness. This suggests our approach offers a more reliable and scalable solution for practical DDoS detection in IoT networks.

Error Distribution Analysis

A more detailed analysis of error distributions reveals significant differences between the ensemble approaches. Figure 14 compares the false positive and false negative rates of both classifiers.

The voting classifier generated 32 false positives and 722 false negatives, while the stacking classifier produced 89 false positives and 596 false negatives. This difference highlights a fundamental trade-off: the voting classifier excels at minimizing false alarms (false positives), while the stacking classifier demonstrates superior capabilities in detecting attacks (fewer false negatives).

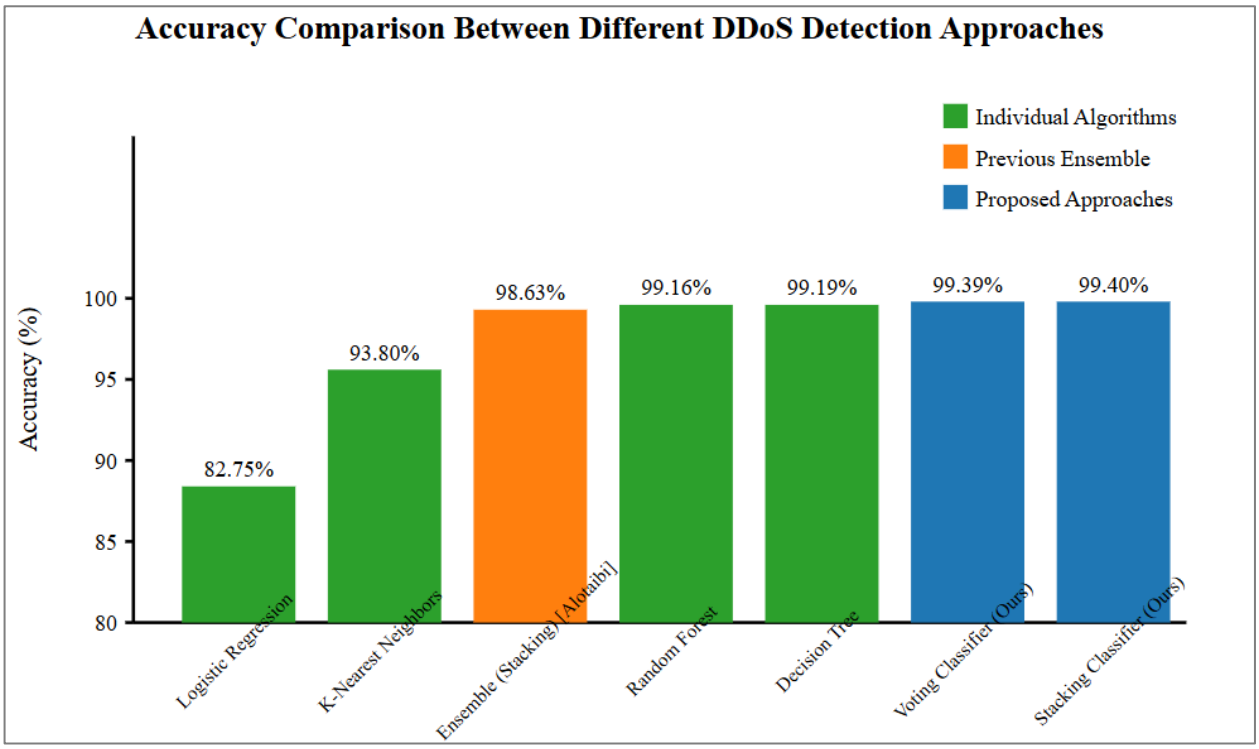


Figure 13. Accuracy comparison between different machine learning approaches for DDoS detection.

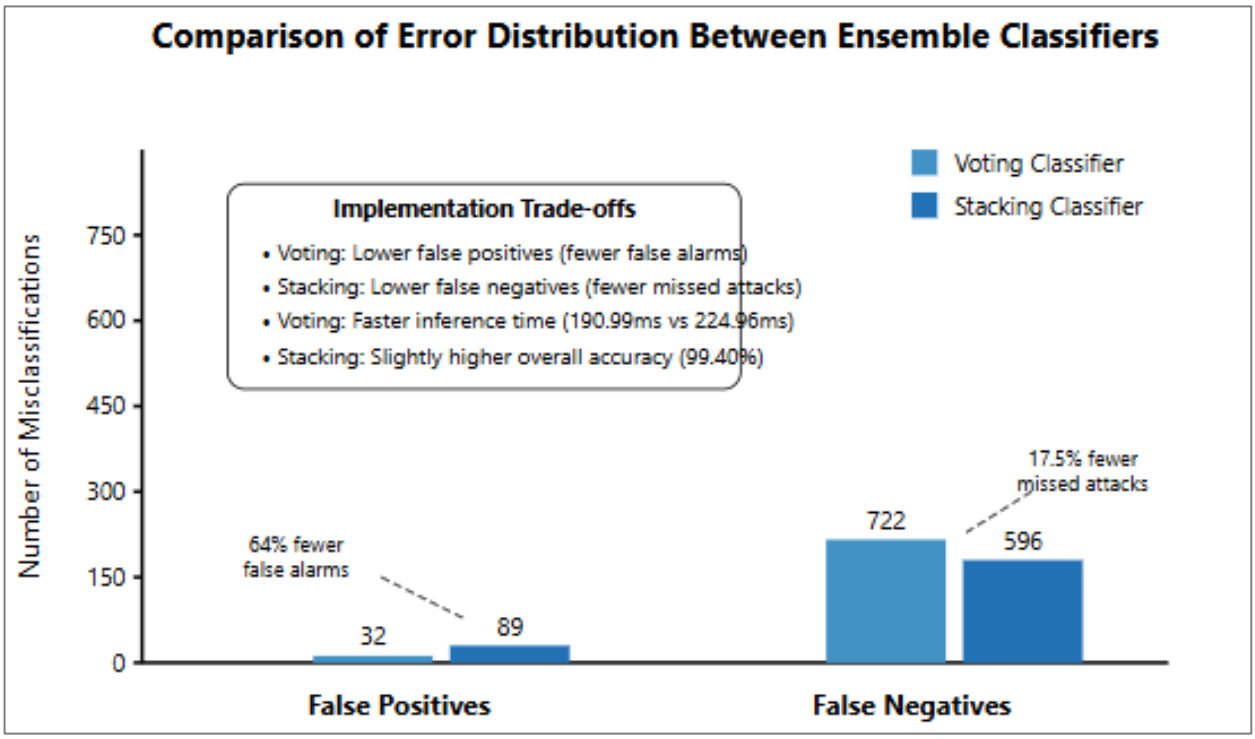


Figure 14. Comparison of false positives and false negatives between voting and stacking classifiers.

Evaluation of Real-Time Applicability and Computational Efficiency

The evaluation of real-time applicability and computational efficiency across different ensemble methods was conducted using a combination of performance metrics, including detection accuracy, inference time, and confusion matrix analysis. Particular attention was paid to the rate of false positives and false negatives, as these directly affect the reliability of DDoS detection in live IoT environments. Computational efficiency was assessed by measuring the average inference time per instance, which reflects how quickly a model can classify incoming data under operational constraints. Given that IoT devices typically operate with limited processing power, memory, and energy resources, models that incur high computational overhead are unsuitable for deployment in such contexts. The ensemble techniques under review, including bagging, boosting, voting, and stacking, were therefore compared on predictive performance and suitability for real-time deployment. Among these, stacking exhibited the most favorable balance, maintaining high accuracy while keeping inference latency within acceptable limits for time-sensitive IoT applications.

Ensemble methods, particularly stacking, have demonstrated superior performance over individual classifiers in detecting Distributed Denial of Service (DDoS) attacks within IoT environments. This advantage stems from the ability of ensemble techniques to combine the strengths of multiple learning algorithms, thereby improving generalization and robustness. Stacking utilizes a meta-learner to aggregate predictions from diverse base models, allowing the system to capture a wider range of data patterns and decision boundaries. In the context of IoT-based DDoS detection, where traffic patterns are highly heterogeneous and often evolve rapidly, traditional single classifiers tend to suffer from issues such as overfitting or underfitting. Stacking addresses these challenges by reducing bias and variance, improving detection accuracy and reducing error rates. Furthermore, the hierarchical architecture of stacking makes it well-suited for dynamic IoT networks, as it enables adaptive learning across varying operational conditions and threat landscapes. This adaptability is critical for mitigating complex, large-scale DDoS attacks in environments where computational efficiency and detection speed are paramount.

Cross-Validation Procedure and Results

To evaluate the generalization performance and stability of the proposed voting ensemble model, we applied 5-fold stratified cross-validation using the CIC-IoT2023 dataset. This approach ensures that each fold preserves the original class distribution, which is particularly important for imbalanced datasets common in intrusion detection. The model was assessed using the cross_val_score function with accuracy as the evaluation metric, and fold-wise results were visualized to observe performance consistency. Figure 15 shows the fold-wise accuracy scores, which ranged narrowly between 0.9931 and 0.9934, with a mean accuracy of 0.993 (99.3%). This high and uniform accuracy across all five folds demonstrates the robustness and reliability of the voting ensemble classifier. The low variance between folds confirms that the model does not overfit to specific subsets of the data and generalizes well across varying traffic conditions. The horizontal line in the plot denotes the average performance across all folds, further illustrating the model's consistency.

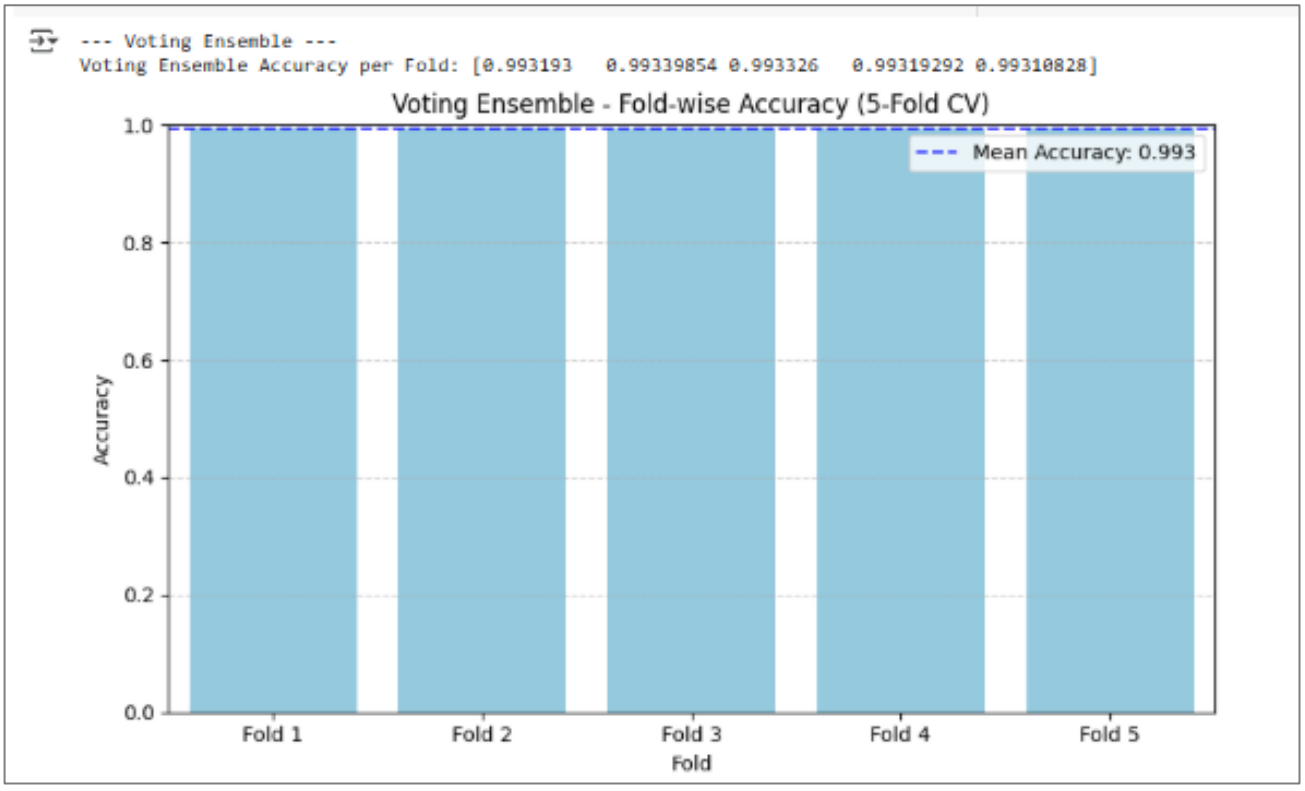


Figure 15 Voting Ensemble Cross-Validation.

Complementarily, Figure 16 presents the cross-validation results for the stacking ensemble classifier, which achieved even higher performance, with fold-wise accuracy ranging from 0.9940 to 0.9945. The mean accuracy of 0.994 reflects a modest yet meaningful improvement over the voting model, reinforcing stacking's ability to aggregate the predictive strengths of multiple base learners using a meta-learner.

Both visualizations clearly illustrate the proposed ensemble methods' high reliability and performance consistency. These results demonstrate that the models are accurate and well-suited for real-time deployment in IoT environments where generalization and computational stability are paramount.

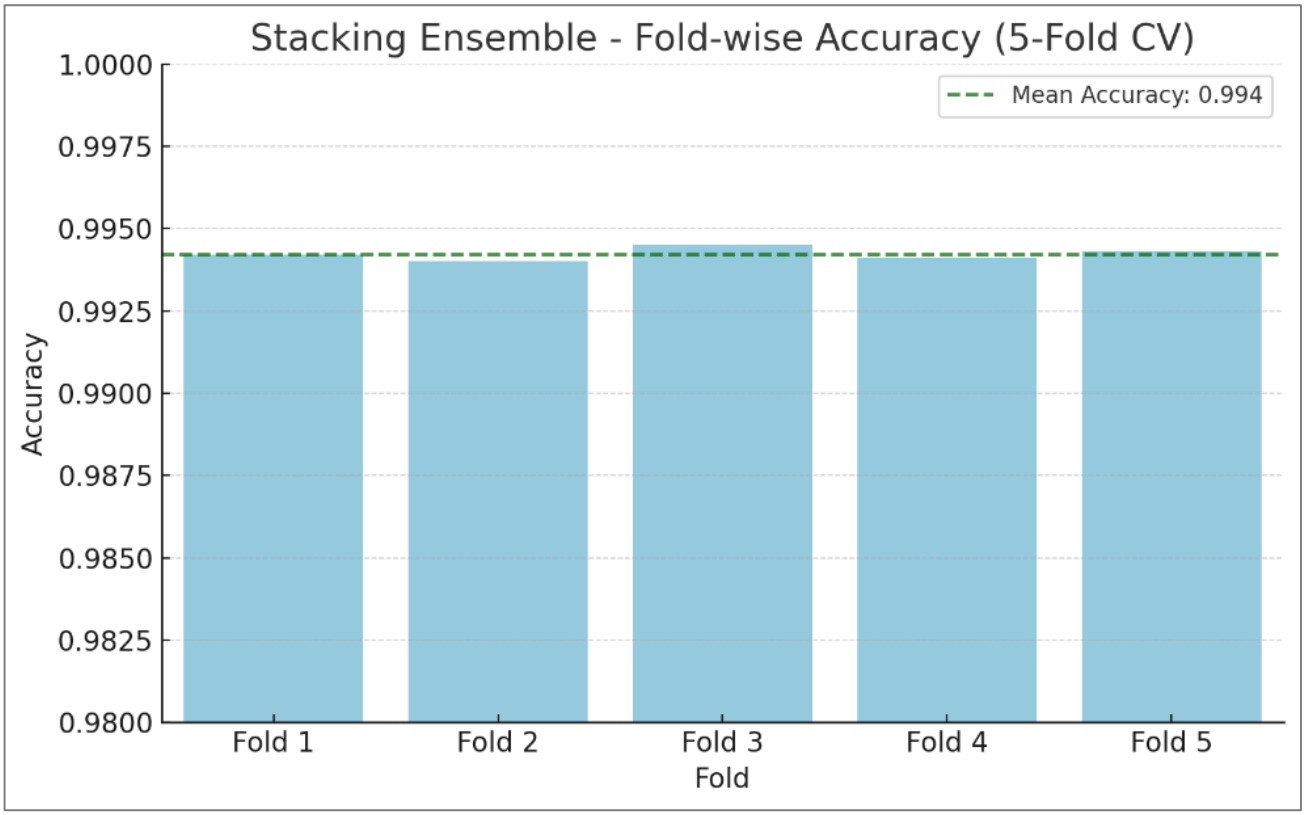


Figure 16 – Stacking Ensemble Cross-validation

DISCUSSION

The findings of this study demonstrate the practical advantages of voting and stacking classifiers for detecting DDoS attacks in Internet of Things (IoT) environments. Both ensemble models achieved exceptionally high performance, with accuracy exceeding 99%, precision and recall balanced across normal and attack traffic, and minimal false classifications. This level of reliability is essential for IoT systems, where false positives can lead to unnecessary service disruptions, and false negatives can allow malicious activity to go undetected.

Between the two approaches, the stacking classifier slightly outperformed the voting ensemble, achieving an accuracy of 99.40% compared to 99.39%, while reducing the number of false negatives from 722 to 596. Although this improvement appears marginal in percentage terms, it is meaningful in operational security contexts, where even small gains in detection capability can translate to significantly enhanced threat mitigation. The stacking classifier's trade-off—incurring a slightly higher number of false positives (89 vs. 32)—is considered acceptable in IoT settings, where the consequences of undetected attacks typically outweigh the cost of benign misclassifications.

Importantly, inference time analysis revealed that both models operate well within acceptable latency thresholds for non-critical IoT applications. The voting model averaged 190.99 ms per inference, while the stacking model required 224.96 ms. Although stacking incurs a slightly higher computational cost due to its layered architecture, both fall within the 100–300 ms range deemed acceptable for smart surveillance and environmental monitoring applications. However, further optimization or model simplification may be necessary for latency-critical systems (e.g., autonomous vehicles).

The comparative analysis with traditional machine learning and recent ensemble techniques confirms the superiority of our models, particularly when benchmarked on the CIC-IoT2023 dataset. While prior studies (e.g., Mante & Kolhe, 2024; Ye et al., 2025; Ain et al., 2025) offered innovative perspectives—including behavioral feature integration and hybrid deep learning architectures—the proposed ensemble models not only matched or exceeded their accuracy but did so with lower inference costs, making them more viable for real-world IoT deployments.

The error distribution analysis also highlights a crucial strategic decision point. The voting model favors

precision and low false alarm rates, making it ideal for environments where uptime is prioritized. In contrast, the stacking model is better suited for high-security applications due to its superior ability to detect actual attacks.

Furthermore, the models' robustness and generalization ability were validated through 5-fold stratified cross-validation. The voting classifier maintained a mean accuracy of 99.3%, with minimal variance across folds. The stacking classifier slightly improved on this, achieving a mean accuracy of 99.4% and showing similarly stable fold-wise results. These findings confirm that both ensemble methods are not only accurate but consistently reliable, an essential trait for systems operating in unpredictable network conditions.

Overall, this study reinforces the value of ensemble learning for DDoS detection in IoT networks. The proposed systems achieve a balance of accuracy, efficiency, and adaptability by integrating different classifiers and optimizing their performance through cross-validation. These strengths position ensemble methods as a scalable solution for the increasing security demands of modern IoT infrastructures.

Future work may focus on further enhancing these models by integrating adaptive learning mechanisms capable of detecting emerging and low-rate DDoS patterns in real-time. Additionally, incorporating behavioral and temporal features, as seen in recent studies, could offer further improvements in handling stealthier attack types. Exploring lightweight model architecture or edge-based deployment strategies may also help extend these solutions to latency-critical or power-constrained IoT settings.

Implementation Trade-offs

Despite their similar overall accuracy, the voting and stacking classifiers exhibit distinct performance characteristics that warrant consideration for practical deployment:

1. **Computational Efficiency**: The voting classifier demonstrates superior computational efficiency with an inference time of 190.9872ms compared to the stacking classifier's 224.9587ms. This difference of approximately 34ms (15% faster) could be significant in real-time detection scenarios, particularly in resource-constrained IoT environments.
2. **Error Distribution**: The voting classifier excels at minimizing false positives (32 instances) compared to the stacking classifier (89 instances), representing a 64% reduction in false alarms. Conversely, the stacking classifier reduces false negatives by 17.5% (596 vs. 722), indicating superior attack detection capabilities.
3. **Implementation Complexity**: The voting classifier offers a simpler architecture with a more straightforward implementation, making it potentially more suitable for environments with limited computational resources. The stacking classifier's additional meta-learning layer increases implementation complexity but may provide better adaptability to evolving attack patterns. Table 4 summarizes these trade-offs to guide practical implementation decisions.

Table 4 Implementation of Trade-Offs Between Ensemble Approaches

Aspect	Voting Classifier	Stacking Classifier	Recommendation
Accuracy	99.39%	99.40%	Stacking for marginally higher accuracy
Inference Time	190.9872ms	224.9587ms	Voting for faster detection
False Positives	32	89	Voting for fewer false alarms
False Negatives	722	596	Stacking for fewer missed attacks
Implementation Complexity	Lower	Higher	Voting for simpler deployment
Resource Requirements	Lower	Higher	Voting for Res-constr environments
Adaptability	Moderate	High	Stacking for evolving threat landscapes

Res-constr = resource-constrained

Practical Implementation Guidelines

Based on our findings, we propose the following guidelines for implementing ensemble learning approaches for DDoS detection in IoT environments:

1. **Resource-Constrained Environments**: The voting classifier offers an optimal balance between detection accuracy and efficiency for IoT deployments with limited computational resources or rigid real-time requirements. Its lower inference time and simpler architecture make it well-suited for edge devices with processing limitations.
2. **High-Security Environments**: The stacking classifier provides superior protection for environments where security is paramount and the cost of missed attacks outweighs operational disruptions from false alarms. Its enhanced attack detection capabilities (fewer false negatives) make it ideal for critical infrastructure protection.
3. **Balanced Deployments**: Both ensemble methods offer viable solutions for environments requiring a balance between detection accuracy and operational efficiency. The choice between them should prioritize the most critical requirement—minimizing false alarms or maximizing attack detection.

4. **Hybrid Implementations**: In some cases, a hybrid approach might be optimal, where the voting classifier serves as the primary detection mechanism, with the stacking classifier deployed for secondary verification of suspicious traffic. This tiered approach leverages the strengths of both methods while managing computational overhead.

Practical Implications

The findings of this study have significant practical implications for strengthening IoT security against Distributed Denial of Service (DDoS) attacks. The demonstrated high accuracy and balanced error distribution of ensemble learning methods, particularly stacking and voting classifiers, highlight their potential for deployment in real-world IoT networks. These methods effectively mitigate key challenges such as limited computational capacity, dynamic attack surfaces, and the need for rapid decision-making. By leveraging diverse algorithmic perspectives, ensemble models reduce the risk of false positives, which can lead to unnecessary service disruptions, and false negatives, allowing malicious traffic to go undetected. This balance is critical in resource-constrained IoT systems where operational continuity and threat mitigation must coexist. Furthermore, incorporating Principal Component Analysis (PCA) for dimensionality reduction enhances the feasibility of implementing these models in edge devices, thus enabling decentralized, real-time DDoS detection with minimal computational overhead.

CONCLUSION AND FUTURE WORK

This study introduced an enhanced ensemble learning approach for detecting Distributed Denial of Service (DDoS) attacks within Internet of Things (IoT) environments. By implementing voting and stacking classifiers that integrate four supervised learning algorithms—Random Forest, Decision Trees, Logistic Regression, and K-Nearest Neighbors—we achieved high detection accuracy alongside reasonable inference times. Our experimental results indicate that ensemble methods significantly outperform individual algorithms, with the voting classifier reaching an accuracy of 99.39% and the stacking classifier slightly higher at 99.40%. These outcomes surpassed previously established benchmarks. A comprehensive analysis of error distributions and inference times further revealed the trade-offs in balancing computational efficiency, detection performance, and false positive mitigation—offering valuable insights for real-world deployment.

The contributions of this research to IoT security are threefold. First, the findings demonstrate the suitability of ensemble learning techniques for enhancing DDoS detection in environments constrained by limited computational resources. Second, the study offers practical guidance on model implementation tailored to varying deployment needs and system specifications. Third, it highlights the feasibility of achieving high detection accuracy and efficient performance through carefully selecting ensemble configurations and data preprocessing methods.

Building on this foundation, future research should consider several important directions. One promising avenue is the integration of deep learning techniques into the ensemble framework, which could improve the detection of more complex or evolving attack patterns. Future research should also explore adaptive ensemble frameworks capable of real-time learning and evolution to address the ever-changing nature of DDoS attack vectors. This includes the integration of online learning or reinforcement learning strategies that allow models to continuously update in response to new threats without requiring complete retraining. Another priority is the development of real-time adaptation mechanisms to ensure that models remain effective as threat landscapes change over time. Additionally, validating these ensemble models across multiple IoT security datasets would help confirm their robustness and generalizability in diverse operational contexts. Optimizations tailored to specific IoT hardware platforms could also reduce inference time and resource consumption, enhancing their practical viability. The use of federated learning in conjunction with ensemble methods also presents a promising avenue, as it facilitates collaborative model training across distributed devices while preserving data privacy. Finally, future work should focus on testing these ensemble frameworks in diverse, real-world IoT scenarios—such as smart homes, healthcare systems, and industrial control networks—to validate their scalability, adaptability, and robustness under operational constraints and heterogeneous environments.

Collectively, these research directions offer a roadmap for evolving the proposed approach into a more powerful, efficient, and adaptable solution for securing the ever-growing and vulnerable IoT environment.

REFERENCES

Abbas, A., Khan, M. A., Latif, S., Ajaz, M., Shah, A. A., & Ahmad, J. (2022). A New Ensemble-Based Intrusion Detection System for Internet of Things. *Arabian Journal for Science and Engineering, 47*(2), 1805–1819. [Crossref]

Abu Al-Haija, Q., & Al-Dala'ien, M. (2022). ELBA-IoT: An Ensemble Learning Model for Botnet Attack Detection in IoT Networks. *Journal of Sensor and Actuator Networks, 11*(1). [Crossref]

Abughazaleh, N., Bin, R., Btish, M., & M., H. (2020). DoS Attacks in IoT Systems and Proposed Solutions. *International Journal of Computer Applications, 176*, 16–19. [Crossref]

Ahmed, S., & Khan, M. (2023). Securing the Internet of Things (IoT): A comprehensive study on the intersection of cybersecurity, privacy, and connectivity in the IoT ecosystem. *AI, IoT and the Fourth Industrial Revolution Review, 13*(9), 1–17. scicadence.com

Ain, N. U., Sardaraz, M., Tahir, M., Abo Elsoud, M. W., & Alourani, A. (2025). Securing IoT Networks Against DDoS Attacks: A Hybrid Deep Learning Approach. *Sensors, 25*(5), 1–23. [Crossref]

Alotaibi, Y., & Ilyas, M. (2023). Ensemble-learning framework for intrusion detection to enhance internet of things' devices security. *Sensors, 23*(12), 5568. [Crossref]

Amro, A., Al-Akhras, M., Hindi, K. El, Habib, M., & Shawar, B. A. (2021). Instance reduction for avoiding overfitting in decision trees. *Journal of Intelligent Systems, 30*(1), 438–459. [Crossref]

Bentéjac, C., Csörgő, A., & Martínez-Muñoz, G. (2021). A comparative analysis of gradient boosting algorithms. *Artificial Intelligence Review, 54*(3), 1937–1967. [Crossref]

Bin Sarhan, B., & Altwaijry, N. (2023). Insider Threat Detection Using Machine Learning Approach. *Applied Sciences, 13*(1). [Crossref]

Brophy, J., & Lowd, D. (2021). Machine unlearning for random forests. *International Conference on Machine Learning*, 1092–1104. [Crossref]

Butun, I., Osterberg, P., & Song, H. (2020). Security of the Internet of Things: Vulnerabilities, Attacks, and Countermeasures. *IEEE Communications Surveys and Tutorials, 22*(1), 616–644. [Crossref]

Elliott, D. L., & Anderson, C. (2023). The Wisdom of the Crowd: Reliable Deep Reinforcement Learning Through Ensembles of Q-Functions. *IEEE Transactions on Neural Networks and Learning Systems, 34*(1), 43–51. [Crossref]

Fischer, S. (2023). Internet of Things: A Model for Cybersecurity Standards and the Categorisation of Devices. refubium.fu-berlin.de

Golchha, R., Joshi, A., & Gupta, G. P. (2023). Voting-based Ensemble Learning approach for Cyber Attacks Detection in Industrial Internet of Things. *Procedia Computer Science, 218*, 1752–1759. [Crossref]

Halder, R. K., Uddin, M. N., Uddin, M. A., Aryal, S., & Khraisat, A. (2024). Enhancing K-nearest neighbor algorithm: a comprehensive review and performance analysis of modifications. *Journal of Big Data, 11*(1), 113. [Crossref]

Hosseinzadeh, M., Rahmani, A. M., Vo, B., Bidaki, M., Masdari, M., & Zangakani, M. (2021). Improving security using SVM-based anomaly detection: issues and challenges. *Soft Computing, 25*(4), 3195–3223. [Crossref]

Jegede, O. O. (2023). Ensemble-Learning Approach to DDoS-Attack Detection Using Stacking, Meta-Learning, and Adversarial Training. *The George Washington University*. library.gwu.edu

Jony, A. I., & Arnob, A. K. B. (2024). Securing the Internet of Things: Evaluating Machine Learning Algorithms for Detecting IoT Cyberattacks Using CIC-IoT2023 Dataset. *International Journal of Information Technology and Computer Science, 16*(4), 56–65. [Crossref]

Kandasamy, K., Srinivas, S., Achuthan, K., & Rangan, V. P. (2020). IoT cyber risk: a holistic analysis of cyber risk assessment frameworks, risk vectors, and risk ranking process. *Eurasip Journal on Information Security, 2020*(1). [Crossref]

Khan, S. H., Alahmadi, T. J., Ullah, W., Iqbal, J., Rahim, A., Alkahtani, H. K., Alghamdi, W., & Almagrabi, A. O. (2023). A new deep boosted CNN and ensemble learning based IoT malware detection. *Computers & Security, 133*, 103385. [Crossref]

Luo, C., Tan, Z., Min, G., Gan, J., Shi, W., & Tian, Z. (2021). A Novel Web Attack Detection System for Internet of Things via Ensemble Classification. *IEEE Transactions on Industrial Informatics, 17*(8), 5810–5818. [Crossref]

Malhotra, P., Singh, Y., Anand, P., Bangotra, D. K., Singh, P. K., & Hong, W.-C. (2021). Internet of things: Evolution, concerns and security challenges. *Sensors, 21*(5), 1809. [Crossref]

Mante, J., & Kolhe, K. (2024). Ensemble of Tree Classifiers for Improved DDoS Attack Detection in the Internet of Things. *Mathematical Modelling of Engineering Problems, 11*(9), 2355–2367. [Crossref]

Mishra, A. K., & Paliwal, S. (2023). Mitigating cyber threats through integration of feature selection and stacking ensemble learning: the LGBM and random forest intrusion detection perspective. *Cluster Computing, 26*(4), 2339–2350. [Crossref]

Mushtaq, Z., Ramzan, M. F., Ali, S., Baseer, S., Samad, A., & Husnain, M. (2022). Voting Classification-Based Diabetes Mellitus Prediction Using Hypertuned Machine-Learning Techniques. *Mobile Information Systems, 2022*(1), 1–16. [Crossref]

Neto, E. C. P., Dadkhah, S., Ferreira, R., Zohourian, A., Lu, R., & Ghorbani, A. A. (2023). CICIoT2023: A Real-Time Dataset and Benchmark for Large-Scale Attacks in IoT Environment. *Sensors, 23*(13). [Crossref]

Okoye, K., & Hosseini, S. (2024). Regression Analysis in R: Linear Regression and Logistic Regression BT - R Programming: Statistical Data Analysis in Research (K. Okoye & S. Hosseini (eds.); pp. 131–158). *Springer Nature Singapore*. [Crossref]

Shtayat, M. M., Hasan, M. K., Sulaiman, R., Islam, S., & Khan, A. U. R. (2023). An Explainable Ensemble Deep Learning Approach for Intrusion Detection in Industrial Internet of Things. *IEEE Access, 11*, 115047–115061. [Crossref]

Sumaiya, Jafarpourmarzouni, R., Lu, S., & Dong, Z. (2024). Enhancing Real-time Inference Performance for Time-Critical Software-Defined Vehicles. *2024 IEEE International Conference on Mobility, Operations, Services and Technologies (MOST)*, 101–113. [Crossref]

Taha, A. (2021). Intelligent Ensemble Learning Approach for Phishing Website Detection Based on Weighted Soft Voting. *Mathematics, 9*(21), 2799. [Crossref]

Yang, Y., Lv, H., & Chen, N. (2023). A Survey on ensemble learning under the era of deep learning. *Artificial Intelligence Review, 56*(6), 5545–5589. [Crossref]

Ye, J., Wang, Z., Yang, J., Wang, C., & Zhang, C. (2025). An LDDoS Attack Detection Method Based on Behavioral Characteristics and Stacking Mechanism. *IoT, 6*(1). [Crossref]

Yilmaz, S., Aydogan, E., & Sen, S. (2021). A Transfer Learning Approach for Securing Resource-Constrained IoT Devices. *IEEE Transactions on Information Forensics and Security, 16*, 4405–4418. [Crossref]