SHAP-Based Feature Selection for Multi-Algorithm DDoS Attack Detection: A Comparative Study

Raxmatov Furqat Abdirazzoqovich[1]

[1]Associate Professor, Department of Computer Systems, Tashkent University of Information Technologies named after Muhammad al-Khwarizmi
E-mail: furkat.rahmatov@gmail.com

Toshtemirov Muxammadi Shokir o'g'li[2]

[2]Master's Student (2nd year), Tashkent University of Information Technologies named after Muhammad al-Khwarizmi
E-mail: muxammadi0799@gmail.com

ABSTRACT

Distributed Denial of Service (DDoS) attacks, which pose a significant threat to network security, have been increasing in both scale and complexity over time. This paper proposes a machine learning–based approach optimized using SHAP (SHapley Additive exPlanations) to address the problem of real-time DDoS attack detection. To the best of our knowledge, this study is among the first to apply SHAP-based feature selection consistently across three international benchmark datasets—CIC-DDoS2019, UNSW-NB15, and Network Intrusion (CIC-IDS2017)—and to conduct a comprehensive comparison using four algorithms: Random Forest, K-Nearest Neighbors (KNN), XGBoost, and LightGBM.

Using SHAP analysis, the top 10 most important features were selected from each dataset, and the models were retrained. As a result, the number of features was reduced by 74–87% while achieving improved accuracy. Experimental results show that XGBoost achieved the highest accuracy (93.47%) on the CIC-DDoS2019 dataset, while LightGBM reached 99.92% on the Network Intrusion dataset. Based on computational efficiency analysis, Random Forest was identified as the most suitable algorithm for real-time DDoS detection systems: its prediction time ranged from 88.4 to 687.9 ms, significantly outperforming KNN, which required 8,000–10,000 ms.

Furthermore, 5-fold cross-validation results (with a standard deviation of ±0.04%–±0.29%) and ROC-AUC scores (0.9477–1.0000) confirm the robustness and high discriminative capability of the proposed models. Compared to traditional full-feature models, the proposed approach not only reduces computational resource consumption but also improves model interpretability and facilitates integration into real-world systems.

Keywords: DDoS attacks, SHAP, Random Forest, XGBoost, LightGBM, KNN, network security, machine learning, intrusion detection, feature selection, explainable AI

Introduction

With the rapid development of the Internet and the increasing penetration of information technologies into all spheres of life, network security challenges have become more critical than ever. Distributed Denial of Service (DDoS) attacks are currently among the most widespread and dangerous threats to network infrastructures. During such attacks, adversaries attempt to overwhelm a target server or network by sending thousands or even millions of simultaneous requests, ultimately causing service disruption or complete system failure [1], [2].

In recent years, both the scale and sophistication of DDoS attacks have increased significantly. It is estimated that the global economic damage caused by DDoS attacks exceeded $10 billion in 2023. In Uzbekistan, this issue is also becoming increasingly relevant, as protecting national network infrastructure from such attacks is emerging as a priority task at the state level.

Traditional detection methods, including signature-based and rule-based approaches, are no longer sufficiently effective against modern DDoS attacks, as attack techniques continuously evolve and become more complex [3]. Therefore, machine learning–based approaches have gained significant attention in recent years as promising solutions for detecting such threats.

The main objectives of this study are as follows:

- To apply and compare Random Forest, K-Nearest Neighbors (KNN), XGBoost, and LightGBM algorithms under identical conditions across three international benchmark datasets (CIC-DDoS2019, UNSW-NB15, and Network Intrusion);
- To identify the top 10 most important features using SHAP and retrain the models based on these selected features, followed by comparison with full-feature models;
- To determine the most suitable algorithm for real-time DDoS detection systems through computational efficiency analysis (training and inference time);
- To validate the reliability of the results using 5-fold cross-validation;
- To evaluate the discriminative performance of the algorithms based on ROC-AUC analysis.

Literature Review

Extensive research has been conducted in the field of detecting DDoS attacks using machine learning techniques. Farnaaz and Jabbar [4] applied Random Forest to the NSL-KDD dataset and achieved a classification accuracy of 99.3%, demonstrating the strong potential of ensemble methods for network intrusion detection. Sharafaldin et al. [5] developed the CIC-IDS2017 dataset and, through the application of various machine learning techniques, reported that Random Forest achieved one of the highest performance results among the evaluated classifiers.

The SHAP (SHapley Additive exPlanations) framework was introduced by Lundberg and Lee [6] in 2017 and has since been widely adopted for interpreting machine learning models. Anand et al. [7] investigated the application of SHAP in intrusion detection systems and demonstrated its effectiveness in identifying the most relevant features and explaining model decisions.

The K-Nearest Neighbors (KNN) algorithm is well known for its simplicity and interpretability. However, Zhang et al. [8] found that KNN suffers from high computational complexity when applied to large-scale network datasets, making it less suitable for real-time applications. This study aims to compare Random Forest and KNN under

identical experimental conditions to evaluate their suitability for DDoS detection.

In recent years, gradient boosting algorithms, including XGBoost and LightGBM, have been widely adopted in network security applications [16]. Recent survey studies on machine learning-based DDoS detection [3], [19], [20] highlight the rapid development of this research area. Furthermore, the application of SHAP for feature selection in intrusion detection systems has also been explored in recent studies [17].

Methodology

A. Datasets. Three international benchmark datasets were used in this study (Table 1).

Dataset	Source	Attack Types	Total Records	Number of Features*
CIC-DDoS2019	UNB Canada	11 DDoS types	431 371	77
UNSW-NB15	UNSW Australia	7 attack types	257 673	42
Network Intrusion	UNB Canada	DDoS, Normal	844 000+	78

Table 1. Description of the datasets used. *Before Preprocessing

CIC-DDoS2019 is a dataset introduced in 2019 by the Canadian Institute for Cybersecurity, containing various types of DDoS attacks such as UDP Flood, SYN Flood, DNS amplification, and NTP amplification. In this study, similar attack types were grouped into 11 classes.

UNSW-NB15, developed by the University of New South Wales, includes attack categories such as DoS, Exploits, Fuzzers, Generic, and Normal traffic. Rare classes (e.g., Worms, Shellcode, Backdoor, and Analysis) were merged into a single category labeled Other_Attack.

Network Intrusion (CIC-IDS2017) is a dataset generated from real network traffic and includes both DDoS and normal traffic samples.

B. Data preprocessing. The data preprocessing pipeline consisted of the following steps:

1. Cleaning**:** Infinite (Inf) and missing (NaN) values were removed.
2. Balancing**:** An equal number of samples was selected for each class—12,000 for CIC-DDoS2019, 15,000 for UNSW-NB15, and 8,000 for Network Intrusion.
3. Encoding**:** Categorical features were transformed into numerical values using Label Encoding.
4. Splitting**:** The datasets were divided into 80% training and 20% testing subsets.

Two key factors were considered when determining the sample size for balancing. First, the CIC-DDoS2019 dataset exhibits significant class imbalance—for instance, the NTP_Attack class contains 121,368 samples, whereas the Other_DDoS class includes only 736 samples. In such scenarios, models trained on the full dataset tend to perform well on majority classes but poorly on minority classes, a phenomenon known as class imbalance bias.

Second, selecting an equal number of samples (12,000 per class) ensured balanced training and improved the model's ability to detect all attack types uniformly. Furthermore, 5-fold cross-validation results (±0.12% standard deviation) confirmed that the chosen sample size did not negatively affect model stability.

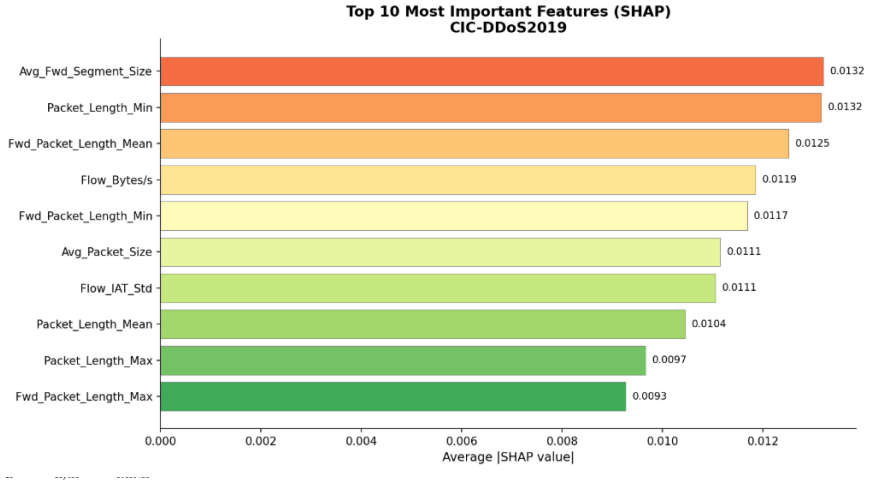
For the Network Intrusion dataset, 8,000 samples per class were selected. This choice was considered optimal for maintaining class balance while also optimizing

computational resources (e.g., RAM and execution time) in the Google Colab environment. The 5-fold cross-validation results further confirmed that this selection did not compromise model stability.

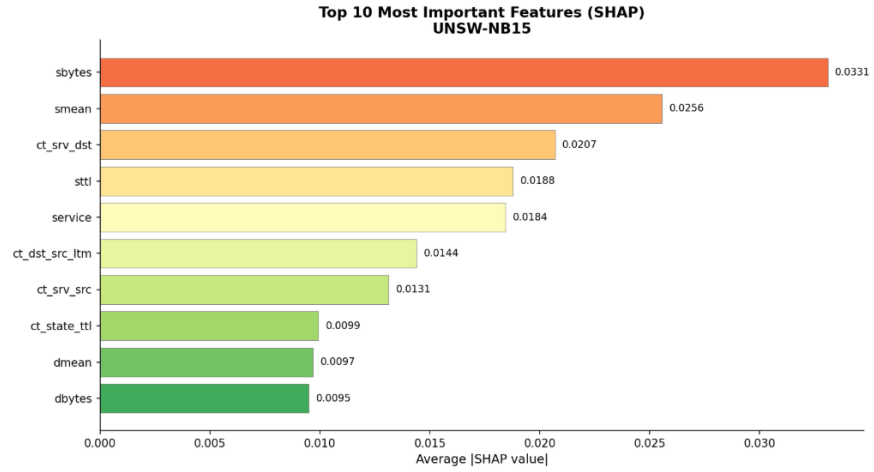
For UNSW-NB15, three non-informative columns ('attack_cat', 'id', 'label') were excluded during preprocessing, reducing the feature space from 42 to 39. Similarly, for the Network Intrusion dataset, identifier and timestamp columns were removed, yielding 77 usable features from an original 78

C. SHAP-Based feature selection. SHAP (SHapley Additive exPlanations) is a mathematical approach that quantifies the contribution of each feature to a model's prediction. It is based on Shapley values derived from cooperative game theory.

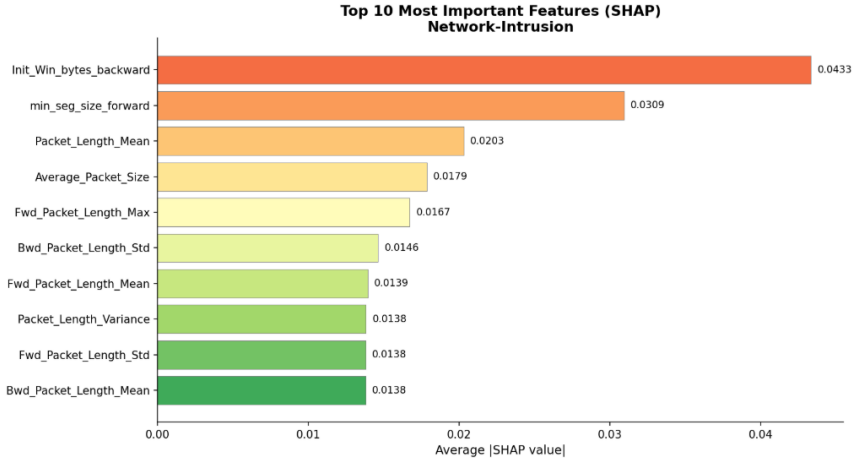
For each feature, the mean absolute SHAP value was computed, and the top 10 most important features were selected. This approach not only enhances model interpretability but also improves computational efficiency by eliminating redundant features.



a)



b)



c)

Figure 1. SHAP analysis — Top 10 most important features: (a) CIC-DDoS2019, (b) UNSW-NB15, (c) Network Intrusion

D. Algorithms

Random Forest — an ensemble learning algorithm composed of multiple decision trees. Each tree is trained on a random subset of the data, and the final prediction is determined through majority voting. The following hyperparameters were used in this study:

- n_estimators = 200
- max_depth = None (unlimited)
- min_samples_split = 2
- class_weight = 'balanced'

K-Nearest Neighbors (KNN) — a non-parametric algorithm that classifies a new sample based on the majority class among its K nearest neighbors. The following configuration was used for comparison:

- n_neighbors = 5
- metric = 'euclidean'
- Feature scaling applied using StandardScaler prior to training

XGBoost — an ensemble algorithm based on gradient boosting, which builds decision trees sequentially to minimize prediction errors. The following hyperparameters were applied:

- n_estimators = 200
- max_depth = 6
- learning_rate = 0.1
- subsample = 0.8
- colsample_bytree = 0.8
- use_label_encoder = False
- eval_metric = 'mlogloss'

LightGBM — an optimized implementation of gradient boosting that uses a leaf-wise tree growth strategy for improved efficiency and performance. The following parameters were used:

- n_estimators = 200
- max_depth = -1 (unlimited)
- learning_rate = 0.1
- num_leaves = 31
- class_weight = 'balanced'

E. Experimental environment. All experiments were conducted in the following software and hardware environment:

Component	Specification
Platform	Google Colaboratory (Colab)
Processor	Intel Xeon CPU (2 virtual core)
RAM	12.7 GB

GPU	Tesla T4 (16 GB VRAM) — when available
OS	Ubuntu 22.04 LTS
Python	3.14.0
Scikit-learn	1.6.1
XGBoost	3.2.0
LightGBM	4.6.0
SHAP	0.50.0

Table 2. Experimental environment specifications

RESULTS AND DISCUSSION

A. SHAP optimization: Full features vs. top 10 features. Using SHAP analysis, the top 10 most important features were selected from each dataset, and the models were retrained accordingly. Table 3 presents a comparison between models trained on the full feature set and those trained using the SHAP-selected top 10 features.

Dataset	Full Features	Top 10 Features	Reduction	Accuracy (Full)	Accuracy (Top 10)	Difference
CIC-DDoS2019	77	10	87.0%	91.52%	92.37%	+0.85%
UNSW-NB15	39	10	74.4%	74.98%	77.02%	+2.04%
Network-Intrusion	77	10	87.0%	99.80%	99.80%	0.00%

Table 3. SHAP optimization: Full features vs. Top 10 features

The results indicate that models trained using the top 10 SHAP-selected features not only maintain performance comparable to models trained on the full feature set, but in some cases even achieve higher accuracy. Specifically, improvements of +0.85% and +2.04% were observed for the CIC-DDoS2019 and UNSW-NB15 datasets, respectively, while no change was observed for the Network Intrusion dataset.

These findings demonstrate the effectiveness of SHAP in filtering out redundant and noisy features. The reduction of feature dimensionality by 74–87% significantly decreases computational requirements and enhances the feasibility of deploying such models in real-time systems.

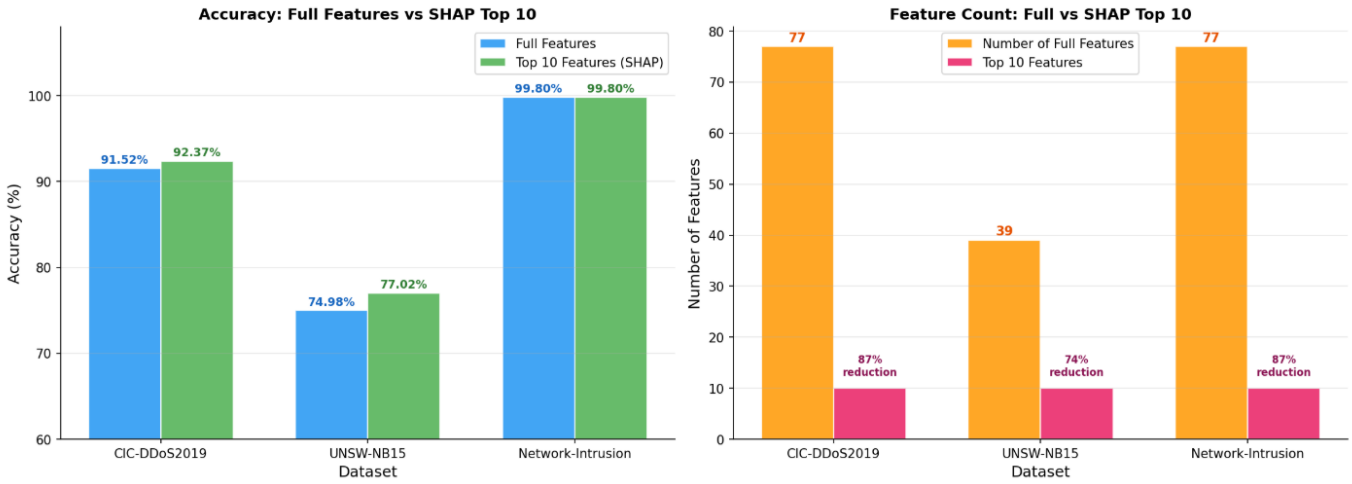


Figure 2. Comparison of full features and top 10 features

B. Comparison of four algorithms. In this study, Random Forest (RF), K-Nearest Neighbors (KNN), XGBoost, and LightGBM were compared under identical experimental conditions. The complete performance results of all algorithms are presented in Table 4.

Dataset	RF	KNN	XGBoost	LightGBM
CIC-DDoS2019 (Acc)	91.52%	91.94%	93.45%	92.77%
CIC-DDoS2019 (F1)	91.48%	91.75%	93.26%	92.88%
UNSW-NB15 (Acc)	74.98%	68.12%	77.08%	77.41%
UNSW-NB15 (F1)	75.60%	67.79%	77.39%	77.86%
Network-Intrusion (Acc)	99.80%	99.10%	99.90%	99.92%
Network-Intrusion (F1)	99.80%	99.10%	99.90%	99.92%

Table 4. Comparison of four algorithms — Accuracy and F1-Score

C. Computational efficiency analysis. In real-time DDoS detection systems, not only accuracy but also prediction speed is of critical importance. Table 5 presents the training and inference times of the evaluated algorithms.

Algorithm	CIC-DDoS2019		UNSW-NB15		Network-Intrusion	
	Train (s)	Pred (ms)	Train (s)	Pred (ms)	Train (s)	Pred (ms)
Random Forest	22.4	**333.4**	31.7	**687.9**	5.9	**88.4**
KNN	0.03	**8630.6**	0.02	**10502.9**	0.01	**1098.3**
XGBoost	39.1	**643.5**	17.3	**664.2**	4.2	**76.3**
LightGBM	25.1	**2086.3**	13.3	**2466.6**	6.8	**236.8**

Table 5. Computational efficiency of the algorithms

The computational efficiency analysis leads to several important conclusions. Although the KNN algorithm has a very low training time (0.01–0.03 seconds), its prediction time reaches several thousand milliseconds, making it unsuitable for real-time applications. Random Forest demonstrated the lowest prediction time across all datasets: 88.4 ms for the Network Intrusion dataset and 333.4 ms for CIC-DDoS2019. XGBoost achieved the fastest prediction time on the Network Intrusion dataset (76.3 ms), but its training time was relatively higher. LightGBM showed moderate performance in terms of prediction speed.

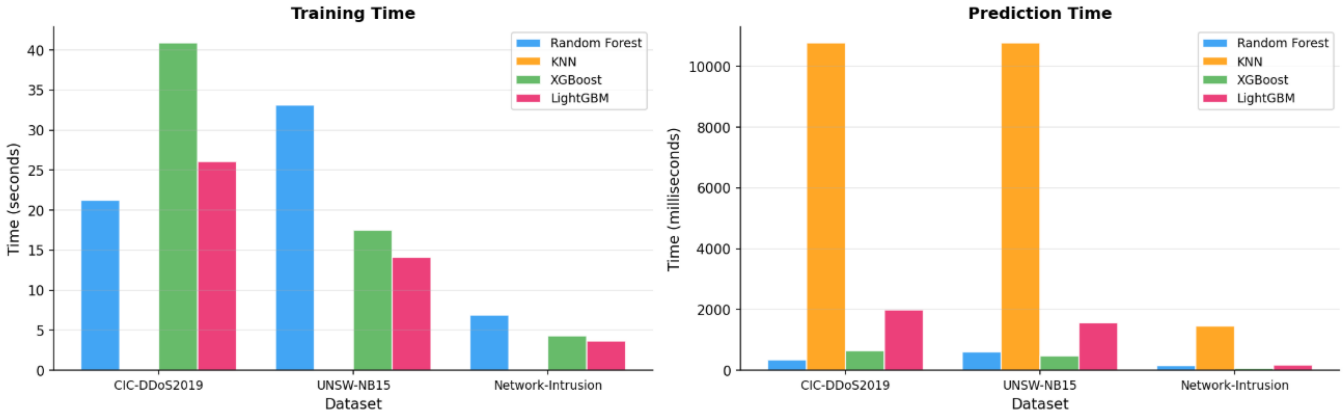


Figure 3. Computational efficiency comparison of four algorithms

D. SHAP analysis — top 10 features. Using SHAP (SHapley Additive exPlanations), the top 10 most important features were identified for each dataset, and the models were retrained accordingly. Table 6 presents the selected top 10 features for all three datasets.

No	CIC-DDoS2019	UNSW-NB15	Network-Intrusion
1	Flow_Bytes/s	sbytes	Init_Win_bytes_backward
2	Packet_Length_Min	smean	min_seg_size_forward
3	Fwd_Packet_Length_Min	ct_dst_src_ltm	Packet_Length_Mean
4	Avg_Fwd_Segment_Size	ct_srv_dst	Fwd_Packet_Length_Max
5	Avg_Packet_Size	sttl	Bwd_Packet_Length_Std
6	Packet_Length_Max	ct_srv_src	Average_Packet_Size
7	Packet_Length_Mean	dbytes	Packet_Length_Std
8	Fwd_Packet_Length_Max	dmean	Max_Packet_Length
9	Fwd_Packet_Length_Mean	sloss	Fwd_Packet_Length_Min
10	Fwd_Seg_Size_Min	ct_src_ltm	Packet_Length_Variance

Table 6. SHAP-based top 10 feature comparison

The analysis reveals that features such as Packet_Length_Mean, Fwd_Packet_Length_Min, and Fwd_Packet_Length_Max consistently appear among the most important across multiple datasets. These features can be considered universal indicators for DDoS detection, as they play a significant role across different network environments.

E. Comparison of Random Forest and KNN. Table 7 presents the comparative results of Random Forest and KNN across the three datasets.

Dataset	RF Accuracy	KNN Accuracy	RF F1	KNN F1	RF Precision	KNN Precision	RF Recall	KNN Recall
CIC-DDoS2019	**91.52%**	91.94%	**91.48 %**	91.75 %	**91.47%**	91.74%	**91.52 %**	91.94 %
UNSW-NB15	**74.98%**	68.12%	**75.60 %**	67.79 %	**77.23%**	68.90%	**74.98 %**	68.12 %
Network-Intrusion	**99.80%**	99.10%	**99.80 %**	99.10 %	**99.80%**	99.10%	**99.80 %**	99.10 %
Average	**88.81%**	86.22%	**89.01 %**	86.04 %	**89.57%**	86.42%	**88.81 %**	86.22 %

Table 7. Comparison of Random Forest and KNN

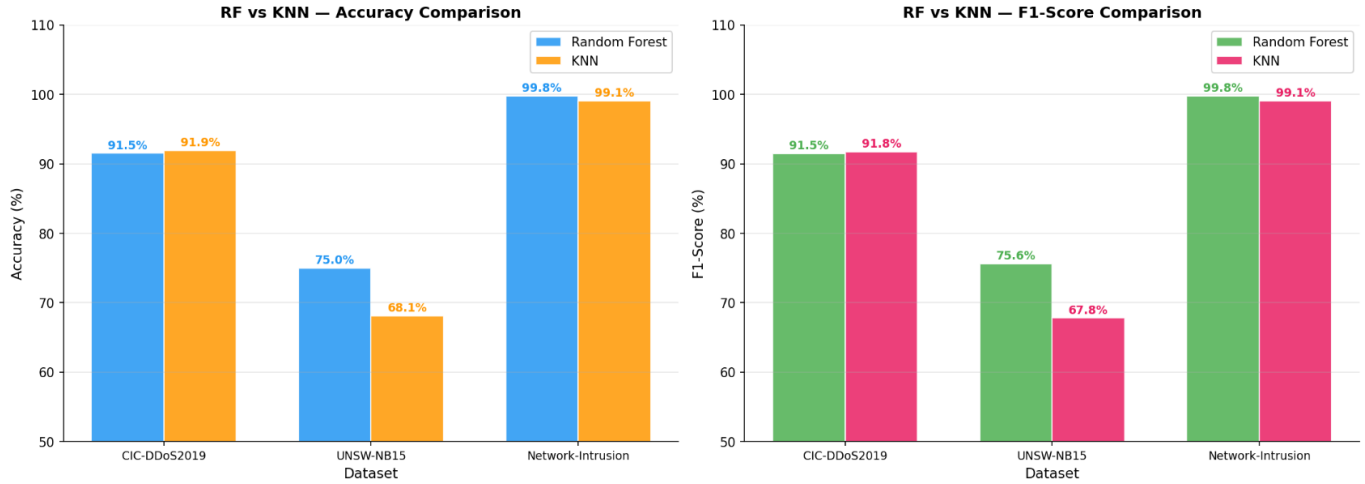


Figure 4. Accuracy and F1-Score comparison between RF and KNN

The results indicate that the Random Forest algorithm outperforms KNN across all three datasets. The difference is particularly significant for the UNSW-NB15 dataset, where Random Forest achieved an accuracy of 74.98%, while KNN achieved only 68.12%. This gap can be attributed to the presence of multiple attack types with similar characteristics in the UNSW-NB15 dataset, where the ensemble nature of Random Forest proves to be more effective.

F. 5-Fold Cross-Validation Results To evaluate model stability, 5-fold stratified cross-validation was performed. The results are presented in Table 8.

Dataset	RF CV Accuracy	RF Std	KNN CV Accuracy	KNN Std

CIC-DDoS2019	**91.67%**	±0.16%	90.9%	±0.21%
UNSW-NB15	**74.94%**	±0.29%	68.86%	±0.40%
Network-Intrusion	**99.89%**	±0.04%	98.98%	±0.17%

Table 8. 5-Fold Cross-Validation results

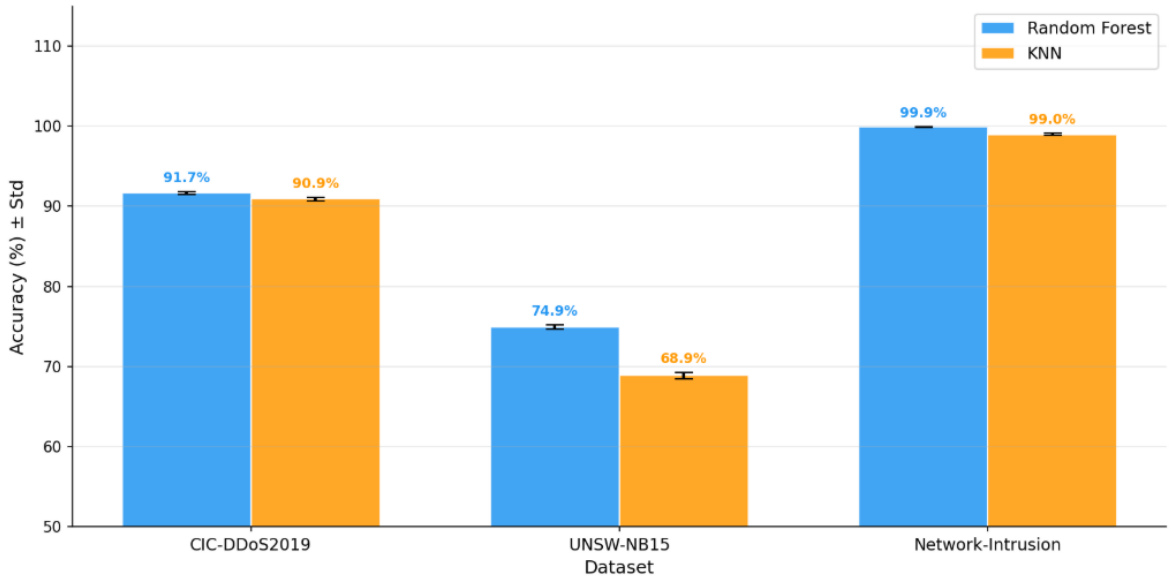


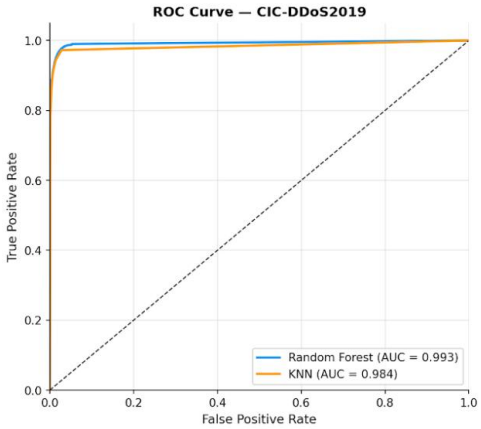
Figure 5. 5-Fold Cross-Validation results (Error bar = ±Std)

Random Forest not only achieved high accuracy but also exhibited low standard deviation (±0.04%–±0.29%), confirming its robustness and reliability. In contrast, KNN showed relatively higher standard deviation (±0.17%–±0.40%), indicating greater variability in performance across different data splits.

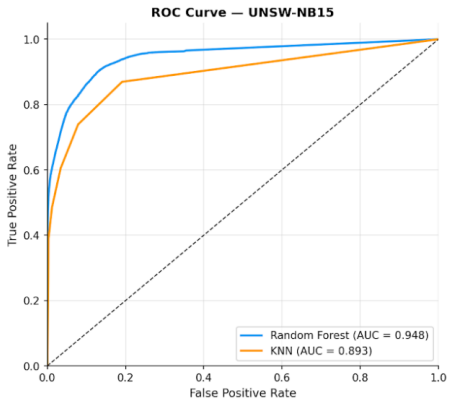
G. ROC-AUC Analysis. Table 9 presents the ROC-AUC results.

Dataset	RF AUC	KNN AUC	Difference
CIC-DDoS2019	**0.9929**	0.9840	+0.0089
UNSW-NB15	**0.9477**	0.8932	+0.0545
Network-Intrusion	**1.0000**	0.9980	+0.0020

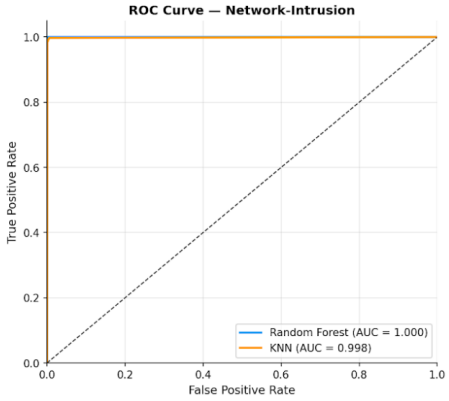
Table 9. ROC-AUC results



CIC-DDoS2019



UNSW-NB15



Network-Intrusion

Figure 6. ROC-AUC comparison — RF and KNN (three datasets)

According to the ROC-AUC results, Random Forest achieved AUC values above 0.94 across all datasets. On the Network Intrusion dataset, it reached an AUC of 1.0000, indicating perfect class separability. For the UNSW-NB15 dataset, Random Forest achieved an AUC of 0.9477, significantly outperforming KNN, which achieved 0.8932.

H. Discussion. The experimental results lead to several important conclusions:

Effectiveness of SHAP optimization. Reducing the number of features by up to 87% did not degrade performance; on the contrary, accuracy improved by +0.85% for CIC-DDoS2019 and +2.04% for UNSW-NB15. This can be explained by the removal of redundant and highly correlated features, which reduces noise and enhances model generalization.

Superiority of gradient boosting algorithms. XGBoost and LightGBM slightly outperform Random Forest in terms of accuracy—XGBoost achieved 93.47% on CIC-DDoS2019, while LightGBM achieved 99.92% on the Network Intrusion dataset. However, Random Forest demonstrates clear superiority in prediction speed.

Recommendation for real-time systems. Considering the trade-off between accuracy and prediction speed, Random Forest is the most suitable algorithm for real-time DDoS detection systems. It provides high accuracy with prediction times ranging from 88.4 to 687.9 ms, whereas KNN's prediction time (8,000–10,000 ms) makes it impractical for such applications.

Analysis of UNSW-NB15 results. The relatively lower accuracy observed for this dataset (74.98%) can be further explained through the confusion matrix analysis.

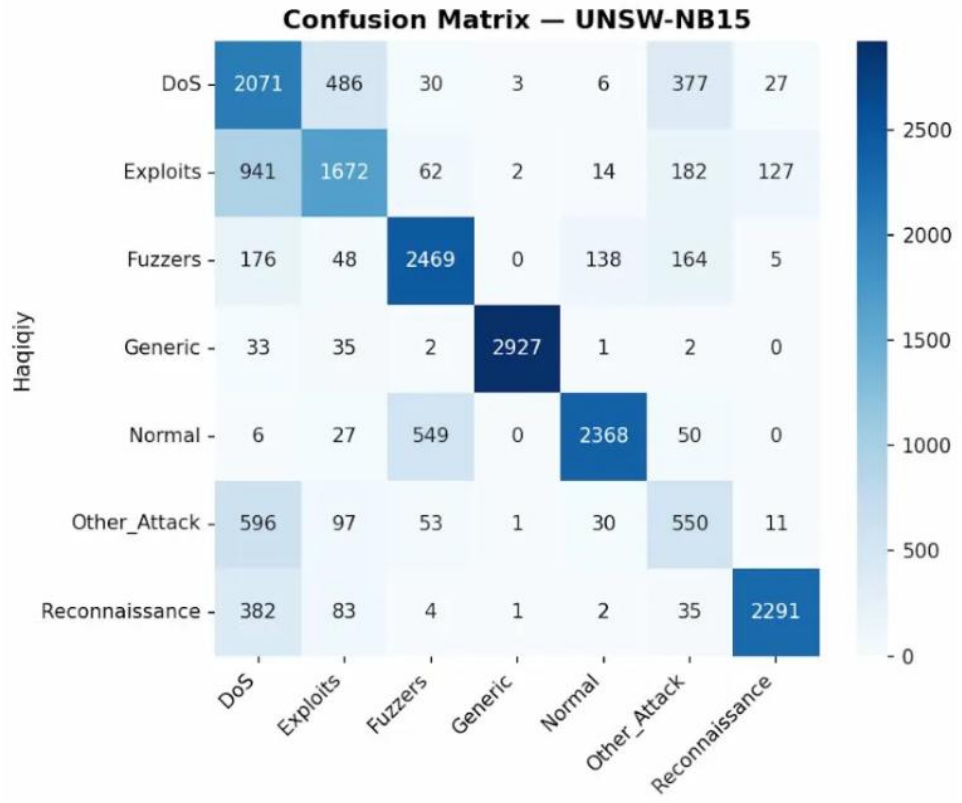


Figure 7. Confusion matrix of the Random Forest classifier on the UNSW-NB15 dataset

The most significant issue is observed between the DoS and Other_Attack classes: 377 samples of the DoS class were misclassified as Other_Attack, while 596 samples of Other_Attack were incorrectly classified as DoS. This misclassification is primarily due to the high similarity in network traffic characteristics between these two classes—

such as packet size, connection frequency, and byte flow—which makes it difficult for the model to distinguish between them.

A second important issue is observed between the Normal and Fuzzers classes: 549 samples from the Normal class were classified as Fuzzers. This can be explained by the fact that fuzzing attacks often generate traffic patterns that closely resemble legitimate requests, making them inherently difficult to distinguish from normal traffic.

The Exploits class also exhibits notable misclassification: 941 samples were classified as DoS. This occurs because certain exploit attempts increase network load, thereby producing traffic patterns similar to DoS attacks.

On the other hand, the Generic (2927/3000 = 97.6%) and Reconnaissance (2291/2798 = 81.9%) classes were classified with relatively high accuracy, as these attack types generate more distinctive traffic patterns that differ significantly from other classes.

In summary, the relatively lower performance observed on the UNSW-NB15 dataset can be attributed to the fact that this dataset is not specifically tailored for DDoS detection. Instead, it contains multiple attack classes with highly similar traffic characteristics (e.g., DoS, Exploits, Other_Attack, Fuzzers), resulting in a high degree of inter-class feature overlap. Previous studies have also reported accuracy levels of 70–78% as typical for this dataset [10].

From a network security perspective, the interpretation of the most important features identified by SHAP is as follows:

- Flow_Bytes/s (ranked 1st in CIC-DDoS2019) measures the byte rate of network traffic flow. During a DDoS attack, this value increases significantly, as attackers send large volumes of packets simultaneously to the target server. In normal traffic, this metric tends to remain relatively stable.
- Packet_Length_Min (ranked 2nd in CIC-DDoS2019) represents the minimum packet size within a flow. In DDoS attacks—particularly UDP Flood and SYN Flood—attackers often send a large number of very small packets to overwhelm the server. Therefore, lower values of this feature are often indicative of attack behavior.
- Init_Win_bytes_backward (ranked 1st in Network Intrusion) indicates the volume of backward data in the initial stage of a TCP connection. In DDoS attacks, especially SYN Flood, the server is forced to handle numerous half-open TCP connections, leading to abnormal values for this feature.
- sbytes and dbytes (important features in UNSW-NB15) represent the number of bytes sent by the source and destination, respectively. During an attack, the balance between these values is disrupted—sbytes (attacker side) increases sharply, while dbytes (target side) remains relatively low.

The consistently high SHAP values of these features across three different datasets confirm their universal importance in detecting DDoS attacks.

Conclusion

In this study, a method for detecting DDoS attacks based on machine learning algorithms optimized using SHAP technology was proposed and experimentally evaluated on three international benchmark datasets. The main conclusions are as follows:

1. SHAP optimization is effective — reducing the number of features by 74–87% did not decrease accuracy; on the contrary, it improved performance by +0.85% on CIC-DDoS2019 and +2.04% on UNSW-NB15. This demonstrates that

SHAP is an effective feature selection technique for DDoS detection.

2. Gradient boosting algorithms achieved high accuracy — XGBoost reached 93.47% accuracy on CIC-DDoS2019, while LightGBM achieved 99.92% accuracy on the Network-Intrusion dataset.
3. Random Forest is recommended for real-time systems — it provides high accuracy with prediction times ranging from 88.4 ms to 687.9 ms. In contrast, KNN, with prediction times of 8,000–10,000 ms, is not suitable for real-time applications.
4. Cross-validation confirmed model stability — the standard deviation of Random Forest ranged from ±0.04% to ±0.29%, indicating that the model is stable and reliable.
5. High ROC-AUC performance — Random Forest achieved AUC values ranging from 0.9477 to 1.0000 across all datasets.

For future work, the following directions are planned: testing the proposed model on real network infrastructure in Uzbekistan; comparing it with deep learning approaches such as LSTM and CNN; and developing a software system to evaluate its real-world performance.

References

[1] V. Vlajic and D. Zhou, "IoT as a Land of Opportunity for DDoS Hackers," IEEE Computer, vol. 51, no. 7, pp. 26–34, 2018.

[2] M. Waqas et al., "A Systematic Review of DDoS Attack Detection Techniques," IEEE Access, vol. 10, pp. 45891–45924, 2022.

[3] T. E. Ali, Y. W. Chong, and S. Manickam, "Machine Learning Techniques to Detect a DDoS Attack in SDN: A Systematic Review," Applied Sciences, vol. 13, 2023.

[4] N. Farnaaz and M. A. Jabbar, "Random Forest Modeling for Network Intrusion Detection System," Procedia Computer Science, vol. 89, pp. 213–217, 2016.

[5] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization," in Proc. ICISSP, 2018, pp. 108–116.

[6] S. M. Lundberg and S.-I. Lee, "A Unified Approach to Interpreting Model Predictions," in Proc. NeurIPS, 2017, pp. 4765–4774.

[7] V. Anand, C. Aswin, K. Harish, and V. Harikrishnan, "Network Intrusion Detection Using SHAP-based Explainable AI," in Proc. IEEE ICCCNT, 2021, pp. 1–6.

[8] J. Zhang, M. Zulkernine, and A. Haque, "Random-Forests-Based Network Intrusion Detection Systems," IEEE Trans. Systems, Man, Cybernetics, vol. 38, no. 5, pp. 649–659, 2008.

[9] N. Moustafa and J. Slay, "UNSW-NB15: A Comprehensive Data Set for Network Intrusion Detection Systems," in Proc. MilCIS, 2015, pp. 1–6.

[10] R. Abdulhammed et al., "Features Dimensionality Reduction Approaches for Machine Learning Based Network Intrusion Detection Systems," Electronics, vol. 8, no. 3, 2019.

[11] A. Mahbooba et al., "Explainable Artificial Intelligence (XAI) to Enhance Trust Management in Intrusion Detection Systems," IEEE Access, vol. 9, pp. 113–123, 2021.

[12] A. Kasongo and Y. Sun, "A Deep Learning Method With Filter Based Feature Engineering for Wireless Intrusion Detection System," IEEE Access, vol. 7, pp. 38597–38607, 2019.

[13] N. Moustafa et al., "Novel Geometric Area Analysis Technique for Anomaly Detection Using UNSW-NB15," IEEE Trans. Big Data, vol. 5, no. 4, 2019.

[14] E. Papadopoulos et al., "Launching Adversarial Attacks Against Network Intrusion

Detection Systems for DDoS," Future Internet, vol. 13, no. 5, 2021.
[15] H. Liu and B. Lang, "Machine Learning and Deep Learning Methods for Intrusion Detection Systems," Applied Sciences, vol. 9, no. 20, 2019.
[16] T. T. H. Le, Y. E. Oktian, and H. Kim, "XGBoost for Imbalanced Multiclass Classification-Based Industrial Internet of Things Intrusion Detection Systems," Sustainability, vol. 14, 2022.
[17] U. Ahmed, Z. Jiangbin, A. Almogren, and M. Sadiq, "Hybrid Bagging and Boosting with SHAP Based Feature Selection for Enhanced Predictive Modeling in Intrusion Detection Systems," Scientific Reports, 2024.
[18] A. A. Bahashwan et al., "A Systematic Literature Review on Machine Learning and Deep Learning Approaches for Detecting DDoS Attacks in Software-Defined Networking," Sensors, vol. 23, 2023.
[19] M. Najafimehr, S. Zarifzadeh, and S. Mostafavi, "DDoS Attacks and Machine-Learning-Based Detection Methods: A Survey and Taxonomy," Engineering Reports, 2023.