Online version at https://journal.lenterailmu.com/index.php/josapen

JOSAPEN

E-ISSN: 3031-2272 (Online)

Explainable Machine Learning for DDoS Attack Detection with Physical Network Validation

Muhammad Azzam Anshori [1] **, *Rahyul Amri*** [2,*]

[1,2]*Informatics Engineering Study Program, Faculty of Engineering, Universitas Riau, Pekanbaru, Indonesia*

ARTICLE INFO

Article history:
Received xx August 2026
Revised 26 August 2026
Accepted 26 August 2026

Keywords:
DDoS Detection
Machine Learning
Explainable AI
SHAP
CIC-DDoS2019

ABSTRACT

Distributed Denial-of-Service (DDoS) attacks remain one of the most disruptive threats to network infrastructure, yet many machine learning (ML)-based detection studies report only offline benchmark performance without verifying whether that performance holds under real network conditions. This study evaluates two explainable ML classifiers, XGBoost and Random Forest, for DDoS detection and examines whether their near-perfect offline accuracy translates into reliable physical-network operation. The study combines offline benchmarking on the CIC-DDoS2019 dataset (293,485 flows) with physical-network validation using a working Intrusion Detection System (IDS) prototype under a controlled SYN-flood attack. Session-disjoint stratified sampling prevented flow-level leakage across attack sessions, while SHapley Additive exPlanations (SHAP) interpreted global and local feature importance. Offline, both classifiers achieved near-perfect performance (accuracy 99.99% for XGBoost, 99.98% for Random Forest; F1 = 0.9999; ROC-AUC up to 1.0000), with no statistically significant difference between them (McNemar's exact test, p = 0.2188), though XGBoost achieved approximately 3.69 times higher inference throughput (1,819,816 flows/s). SHAP identified Min Packet Length, Fwd Packet Length Min, Inbound, Protocol, and Init_Win_bytes_forward as the most influential features. In physical deployment, however, the IDS prototype flagged 6.26% of captured flows (6,935 of 110,762) as ATTACK during the SYN-flood test, and a separate 397-flow ambient-benign subset yielded a 6.80% false positive rate (95% Wilson CI: 4.72–9.71%), with short-duration SSDP/UPnP-style UDP control traffic accounting for 70% of observed false positives. This gap shows that near-perfect offline accuracy does not guarantee low false positives in real deployment, indicating that offline benchmarks alone are insufficient for validating IDS readiness.

1. Introduction

In the current digital era, information and communication technology has advanced rapidly and become the primary backbone of various strategic sectors such as banking, education, government, and healthcare services. However, the high dependency on internet network infrastructure is accompanied by increasingly complex and frequent cyber threats. One of the most destructive and continuously evolving cybersecurity threats is the Distributed Denial-of-Service (DDoS) attack [1].

DDoS attacks flood targets, such as servers, applications, or network bandwidth, with fake traffic originating from distributed botnets, rendering services inaccessible to legitimate users [2].

To mitigate this threat, an Intrusion Detection System (IDS) plays a crucial role as the frontline of network defense [3]. Traditional rule-based or signature-based IDS mechanisms have significant limitations in identifying new attack variants (zero-day attacks) and often fail to cope with the fast dynamics of network traffic [4]. This has driven the adoption of Machine Learning (ML) for detecting network traffic anomalies [5]. Various ML algorithms such as Decision Tree, Random Forest, Support Vector Machine

* *Corresponding author:* Rahyul Amri
email: rahyulamri@lecturer.unri.ac.id

(SVM), K-Nearest Neighbors (KNN), and Extreme Gradient Boosting (XGBoost) have been shown to achieve very high network traffic classification accuracy [6], [7].

Nevertheless, the application of ML in Intrusion Detection Systems still faces two fundamental challenges. The first is the "black-box" problem, i.e., the limited transparency of model decisions [8]. Complex ensemble algorithms often classify traffic without providing a rational explanation of which features indicate an attack, which for security analysts hinders mitigation and creates doubt in corrective actions [9]. The Explainable Artificial Intelligence (XAI) approach, particularly Shapley Additive Explanations (SHAP), offers a robust theoretical solution for global and local interpretation of ML predictions [10], [11].

The second challenge is the gap between model performance evaluation on static datasets (offline benchmarking) and practical implementation on real-time physical networks [12], where robustness against ambient traffic such as DHCP, mDNS, and SSDP remains largely unvalidated. Prior studies have combined ensemble methods with SHAP [13] or compared classical algorithms on CIC-DDoS variants [6], [14], [15], [16], [17], [18], but are generally limited to offline benchmarking and have not measured model robustness against ambient traffic on real networks. This gap is what distinguishes the present study through quantitative validation across two deployment scenarios (VM-to-VM and physical network).

Based on the above problems, this study evaluates and compares the performance of the XGBoost and Random Forest algorithms in detecting DDoS attacks using the standardized public dataset CIC-DDoS2019. The main contributions of this study are fourfold. First, a session-disjoint comparison of XGBoost and Random Forest for binary DDoS detection is performed, explicitly grouping flows by attack session prior to splitting to reduce flow-level leakage risk. Second, SHAP-based analysis is applied to identify the flow-level statistical characteristics most strongly associated with the selected classifier's decisions. Third, model-inference throughput is benchmarked for both classifiers under controlled, documented computational conditions, reporting the relative efficiency of the evaluated configurations rather than an intrinsic architectural claim. Fourth, and most importantly, the selected classifier is validated in a physical Wi-Fi network under controlled SYN-flood traffic and ambient background traffic; this experiment reveals that near-perfect benchmark performance does not necessarily imply robust performance on previously underrepresented benign traffic, producing a measurable false-positive rate concentrated in UDP-based control protocols such as SSDP and mDNS. Physical-network validation is therefore treated not merely as a deployment confirmation step, but as a diagnostic tool for exposing the distributional limitations of benchmark-trained DDoS classifiers.

2. Method

2.1 Research Stages

This study was conducted systematically following structured experimental stages, starting from problem identification, data collection, data preprocessing, ML model training, performance evaluation, SHAP XAI interpretation, up to real-time IDS prototype testing on a physical network. The research workflow is presented in Figure 1.

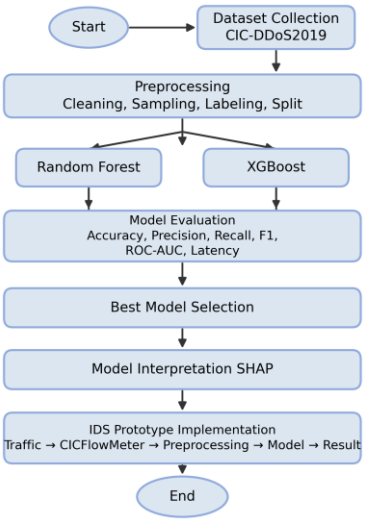

Fig. 1 - The research workflow

2.2 Research Hardware and Dataset

Experiments were conducted using hardware and software with standardized specifications to ensure valid and reproducible computational results, summarized in Table 1 and Table 2.

Table 1 - Research Hardware Specifications

Component	Detailed Specification
Processor (CPU)	AMD Ryzen 7 6800H (8 Cores, 16 Threads, up to 4.7 GHz)
Memory (RAM)	8 GB DDR5 4800 MHz
Storage	512 GB SSD NVMe PCIe 4.0
Network Adapter	Wi-Fi 6 (802.11ax) & Realtek PCIe GbE Family Controller

Table 2 - Software and Libraries Used

Software/Library	Version	Main Function
Windows 11 / Ubuntu 22.04 LTS	22H2 / 22.04	Host and VM operating system
Python	3.11.9	Main programming language for ML experiments
Scikit-Learn	1.8.0	Preprocessing, train-test split, RF baseline
XGBoost	3.2.0	Extreme Gradient Boosting implementation
SHAP	0.51.0	Explainable AI (XAI) interpretation framework
CICFlowMeter	4.0	Network flow feature extraction (80 attributes)
Kali Linux & hping3	2024.1	Attack machine and SYN/UDP flood generator
Oracle VM VirtualBox	7.2.10	Isolated virtualization testing environment

The dataset used is the CIC-DDoS2019 benchmark dataset from the Canadian Institute for Cybersecurity (CIC), which has been widely adopted in recent DDoS detection research [19], [20]. This dataset covers various TCP/UDP protocol-based DDoS attack scenarios, such as SYN flood, UDP flood, MSSQL, NetBIOS, LDAP, DNS, NTP, SNMP, and SSDP, as well as normal (benign) traffic.

2.3 Data Preprocessing

Data cleaning began with removing invalid values (missing, NaN, and infinity values) produced during network extraction [14]. Identity features that are constant or cause data leakage, such as Unnamed: 0, Flow ID, Source IP, Source Port, Destination IP, and Timestamp, were removed [21]. The retained flow statistical

features do not constitute data leakage, since they are computed from traffic characteristics observable in real time during inference, not from the label itself.

The original CIC-DDoS2019 corpus spans more than 50 million labeled flow rows across the eleven attack-type files used in this study, which is computationally prohibitive to load and process in full under the available memory budget; a fixed per-file sampling quota was therefore set for the ATTACK class, allocated proportionally to each attack type's share of the total attack population so that the resulting subset preserved the original relative distribution across attack types and thereby avoided arbitrary or cherry-picked selection, while the BENIGN class was retained in full since its population was already substantially smaller than the ATTACK quota. Both the per-file sampling and the subsequent split (below) were performed with a fixed random seed (random_state = 42) to support exact reproducibility. After cleaning, proportional stratified sampling was applied, resulting in a final population of 293,485 flow rows (50,446 BENIGN; 243,039 ATTACK). To prevent flow-level leakage from correlated traffic within the same attack session, flows were first grouped into sessions by concatenating source IP, destination IP, and timestamp floored to a 1-minute window—i.e., flows sharing the same source-destination IP pair within the same 1-minute interval were treated as one session—an operational grouping for leakage control, not a protocol-level session (15,782 unique sessions, average 18.60 flows/session), and the dataset was split at the session level using GroupShuffleSplit (random_state = 42) targeting an 80/20 ratio, yielding a training set of 231,327 samples (12,625 sessions) and a testing set of 62,158 samples (3,157 sessions), with verified zero session overlap between the two partitions. This grouping controls for leakage within a single 1-minute window but does not guarantee independence between sessions belonging to the same broader attack campaign that spans multiple consecutive windows or hosts; a small residual correlation between train and test partitions may therefore remain, and campaign-level or file-level disjoint splitting would provide a stricter leakage control in future work. Because splitting was performed at the session rather than sample level, the resulting sample-level ratio (78.8%/21.2%) deviates slightly from the 80/20 target.

2.4 Machine Learning Algorithms and Hyperparameter Configuration

This study compares two ensemble learning algorithms: XGBoost and Random Forest. XGBoost builds decision trees sequentially (iterative boosting), minimizing the loss function through gradient descent and L1/L2 regularization [16]. Random Forest works based on bagging, building many independent decision trees that produce predictions based on majority voting [17].

The XGBoost score (pre-sigmoid) is an aggregation of all trees built iteratively, formulated in Equation (1), where K denotes the number of trees and $f_k(x_i)$ is the prediction contribution of the k-th tree for the i-th data point.

$$\hat{y}_i = \sum_{k=1}^{K} f_k(x_i) \quad (1)$$

Each split in the Random Forest decision tree is chosen based on the lowest Gini impurity value, formulated in Equation (2), where c denotes the number of classes and p_i is the proportion of class-i samples in the evaluated node.

$$Gini = 1 - \sum_{i=1}^{c} {p_i}^2 \quad (2)$$

The hyperparameter configuration uses common empirical settings shown in Table 3 and Table 4, determined from official documentation recommendations and preliminary experiments to balance accuracy and inference efficiency, without extensive hyperparameter search.

Table 3 - XGBoost Hyperparameter Configuration

Parameter	Value	Function Description
n_estimators	150	Number of decision trees (boosting iterations)
max_depth	6	Maximum depth of each decision tree
learning_rate (eta)	0.15	Feature weight shrinkage factor
subsample	0.8	Row sample ratio for training each tree
colsample_bytree	0.8	Column feature ratio for training each tree
objective	binary: logistic	Binary classification probability objective
eval_metric	logloss	Loss evaluation metric during iterations
random_state	42	Random seed for reproducibility

Table 4 - Random Forest Hyperparameter Configuration

Parameter	Value	Function Description
n_estimators	150	Number of independent decision trees
max_depth	18	Maximum depth limit of decision trees
max_features	sqrt	Number of random features evaluated per split
class_weight	balanced	Automatic sample weight adjustment
random_state	42	Random seed for reproducibility

2.5 Classification Evaluation Metrics and Inference Benchmark

Model performance was evaluated using a Confusion Matrix classifying predictions into True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN) [22]. Standard evaluation metrics were computed through Equations (3)–(6).

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (3)$$

$$Precision = \frac{TP}{TP+FP} \quad (4)$$

$$Recall = \frac{TP}{TP+FN} \quad (5)$$

$$F1 - Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (6)$$

In addition, the ROC-AUC metric [23] was measured, along with computational inference benchmarks covering Throughput (flow/second) and Latency (second/flow) [24], measured specifically at the model.predict() stage after flow features were available, excluding CICFlowMeter feature extraction time since it is identical for both classifiers.

2.6 Model Interpretation (Explainable AI: SHAP)

To open the "black-box" nature of the ensemble models, the Shapley Additive Explanations (SHAP) method based on cooperative game theory was applied [10]. The SHAP value measures the marginal contribution of each feature xi to the model output f(x) compared to the mean expected value E[f(x)], formulated in Equation (7) [25].

$$f(x) = g(z') = \varphi_0 + \Sigma_i \varphi_i z_i' \quad (7)$$

Here, g(z′) is the explanation model based on the

simplified binary representation z′, φ0 is the base (expected) model output, and φi denotes the SHAP value of the i-th feature. SHAP analysis was performed globally using the SHAP Summary Plot and Feature Importance Bar Plot [13], computed using TreeExplainer applied to a stratified subset of 500 samples drawn from the test set to preserve class proportion while keeping computation tractable.

2.7 IDS Prototype Architecture and Testbed Scheme

The IDS prototype was built using Python, integrating the CICFlowMeter real-time feature extraction engine with the best-trained ML model. The prototype captures live packets from a physical network interface, groups packets into 5-tuple flows, converts flow statistics into an 80-feature matrix, and directly performs binary classification (ATTACK vs BENIGN), as illustrated in Figure 2. Testing was conducted in two environments summarized in Table 5.

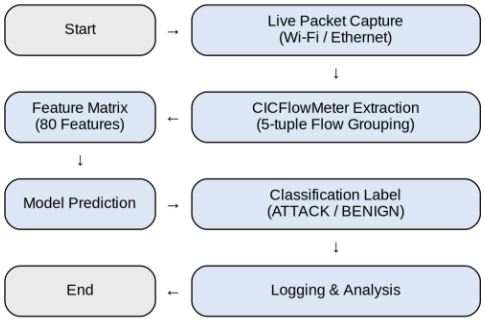

Fig. 2 - IDS Prototype Implementation Flow, from Live Packet Capture to Classification Output

In the Virtual Environment (VM-to-VM) scenario, two Kali Linux attacker VMs targeted a single Ubuntu target VM (also serving as the IDS server) on one host, with NIC 1 in Host-only mode to isolate test traffic for SYN-flood detection testing.

In the Physical Network Environment scenario, both attacker VMs stayed on the same host, but NIC 1 was switched to Bridged mode (MediaTek Wi-Fi 6 MT7921, Promiscuous Mode: Allow All) so attack traffic left the host and reached a second physical laptop over the local Wi-Fi network, testing the prototype against ambient background traffic and controlled SYN-flood attacks generated with hping3 [18].

Table 5 - Device Specifications for the VM-to-VM and Physical Testing Schemes

Device/VM	Role	Specification	NIC Config
Laptop 1 (ASUS TUF A15)	VirtualBox host, all VMs	AMD Ryzen 7 6800H, 8 GB DDR5, 512 GB NVMe SSD	Wi-Fi 6 & Realtek PCIe GbE
Target/IDS Server VM	Attack target & IDS host (VM-to-VM only)	Ubuntu 64-bit, 2048 MB RAM, 2 vCPU	NIC1: Host-only; NIC2: NAT
Attacker VM 1	Attacker 1 (SYN/UDP flood)	Kali Linux 2026.2, 2048 MB RAM, 2 vCPU	Host-only / Bridged (MT7921)
Attacker VM 2	Attacker 2 (simultaneous)	Kali Linux 2026.2, 2048 MB RAM, 2 vCPU	Host-only / Bridged (MT7921)
Laptop 2 (HP 14s-fq1xxx)	Direct physical target (physical scheme only)	AMD Ryzen 5 5500U, 16 GB RAM, 487.3 GB SSD, Win11	Realtek RTL8822CE 802.11ac

3. Results and Discussion

3.1 Dataset Characteristics and Preprocessing Results

The data cleaning and proportional stratified sampling stages produced a final dataset of 293,485 flow rows (session-aware, GroupShuffleSplit). The stratified sample split is summarized in Table 6 and Table 7.

Table 6 - Final Research Dataset Class Distribution

Class Category	Samples (Flow)	Percentage
BENIGN (Normal Traffic)	50,446	17.19%
ATTACK (DDoS Attack)	243,039	82.81%
Total Dataset Population	293,485	100.00%

Table 7 - Stratified Training and Testing Data Split

Data Subset	Total Samples	BENIGN	ATTACK
Training Set (78.8%)	231,327	40,870	190,457
Testing Set (21.2%)	62,158	9,576	52,582

3.2 Model Classification Performance Evaluation

Model training was highly efficient for XGBoost (2.86 s) versus Random Forest (12.71 s). Classification performance on the test data (62,158 samples) is presented in Table 8 and Table 9.

Table 8 - ML Model Classification Performance Evaluation

Model	Acc.	Prec.	Rec.	F1	AUC
XGBoost	0.9999	0.9999	0.9999	0.9999	1.0000
Random Forest	0.9998	0.9999	0.9998	0.9999	0.9999

Table 9 - Detailed Classification Performance per Target Class

Model	Class	Prec.	Rec.	F1	N
XGBoost	BENIGN (0)	0.9995	0.9996	0.9995	9,576
XGBoost	ATTACK (1)	0.9999	0.9999	0.9999	52,582
Random Forest	BENIGN (0)	0.9991	0.9996	0.9993	9,576
Random Forest	ATTACK (1)	0.9999	0.9998	0.9999	52,582

To examine consistency across the original attack categories, a post-hoc analysis was conducted on detection performance per original attack-type label prior to binary encoding. This does not represent multiclass performance; the model remains a binary classifier, and original labels were used solely for this per-category consistency check. All 13 original traffic categories achieved a minimum recall of 99.96%; ten categories (TFTP, DrDoS_NetBIOS, DrDoS_SSDP, DrDoS_LDAP, DrDoS_UDP, DrDoS_NTP, DrDoS_MSSQL, Syn, UDP-lag, WebDDoS) reached 100% recall, while the two attack categories below 100% recall (DrDoS_DNS and DrDoS_SNMP), together with the BENIGN baseline for comparison, are detailed in Table 10. Two categories had very few test samples, WebDDoS (n = 1) and UDP-lag (n = 2); both reached 100% recall despite the small samples, so the 95% Wilson score CI is 20.7–100% for WebDDoS and 34.2–100% for UDP-lag, and these two estimates should be treated as indicative rather than reliable evidence of per-class generalization.

Table 10 - XGBoost Detection Performance by Traffic Category (Including Benign Baseline)

Traffic Type	N Test	N Correct	Recall
BENIGN	9,576	9,572	99.96%
DrDoS_DNS	7,998	7,995	99.96%
DrDoS_SNMP	4,827	4,825	99.96%

Both models show exceptionally high accuracy (99.98–99.99%) with an F1-Score of 0.9999, consistent with the high separability reported for CIC-DDoS2019 [14]. The Confusion Matrix is in Figure 3 and the ROC Curve in Figure 4. Given the imbalanced class composition (BENIGN 17.19% vs ATTACK 82.81%), the Matthews Correlation Coefficient (MCC) and Balanced Accuracy were also computed: XGBoost (0.9994; 99.97%) and Random Forest (0.9992; 99.97%), both well above the 82.81% majority-class baseline, indicating the high performance is not primarily attributable to class imbalance. The ROC-AUC of 1.0000 should be interpreted with caution, as it originates from a highly separable benchmark dataset and may not necessarily reflect production network conditions.

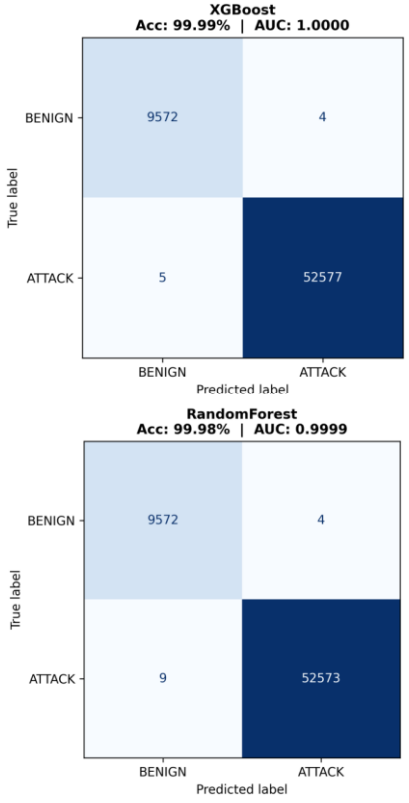

Fig. 3 - (a) Confusion Matrix of the XGBoost Model, (b) Confusion Matrix of the RandomForest Model

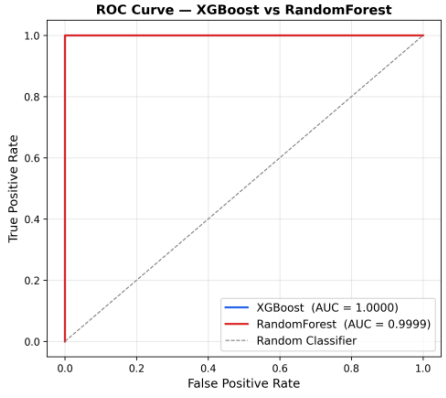

Fig. 4 - ROC Curve of the XGBoost and RF Models

3.3 Inference Benchmark and Computational Efficiency

Inference efficiency is important for live IDS applications; however, this benchmark measures only the ML inference stage after flow-level features is available and does not represent end-to-end IDS latency. Results are presented in Table 11. Latency per flow was calculated from batch prediction time divided by number of flows, so this metric is more precisely characterized as batch-normalized model-inference latency rather than a per-flow end-to-end measurement.

Table 11 - Model Computational Inference Efficiency Benchmark

Model	Mean (s)	Std Dev	Best (s)	Thr. (flow/s)	Time/Flow
XGBoost	0.034156	±0.004156	0.023837	1,819,816	0.55 µs
Random Forest	0.125906	±0.008785	0.113088	493,687	2.03 µs

Although both models have comparable classification accuracy, XGBoost is markedly faster at the inference stage, achieving a batch-normalized throughput of up to 1,819,816 flows/second (0.55 µs/flow), 3.69× higher than Random Forest (493,687 flow/s); this benchmark measures only the model.predict() stage and excludes CICFlowMeter feature extraction (identical across both classifiers), so the speedup reflects inference-stage efficiency rather than full end-to-end pipeline latency. Given comparable classification performance, XGBoost is therefore preferable for live IDS deployment when computational cost matters. This aligns with XGBoost's shallower trees (max_depth = 6) versus Random Forest's deeper ensemble (max_depth = 18, Tables 3–4), reflecting the evaluated configurations rather than an inherent algorithmic property.

3.4 Generalization Gap Analysis

To assess the generalization gap, training and testing performance were compared, summarized in Table 12.

Table 12 - Train vs Test Performance Comparison (Generalization Gap)

Model	Metric	Train	Test	Gap
XGBoost	F1-Score	1.0000	0.9999	0.0001
XGBoost	Accuracy	1.0000	0.9999	0.0001
XGBoost	ROC-AUC	1.0000	1.0000	0.0000
Random Forest	F1-Score	1.0000	0.9999	0.0001
Random Forest	Accuracy	1.0000	0.9998	0.0002
Random Forest	ROC-AUC	1.0000	0.9999	0.0001

The negligible gap (≤0.0002) indicates no substantial train–test degradation under this session-disjoint protocol (Section 2.3), which reduces—though not fully eliminates—flow-level leakage relative to a random split. McNemar's exact test on the full test set (n = 62,158) found only 6 discordant predictions (n01 = 1, n10 = 5), p = 0.2188 (p > 0.05), indicating no statistically significant difference between the two classifiers; this non-significant result does not by itself establish equivalence. The learning curve is shown in Figure 5. This result supports within-dataset generalization to unseen CIC-DDoS2019 sessions more convincingly than a random split would, though it does not establish generalization to a different traffic distribution — a distinction examined further alongside the representational gap observed during physical-network testing (Section 3.6).

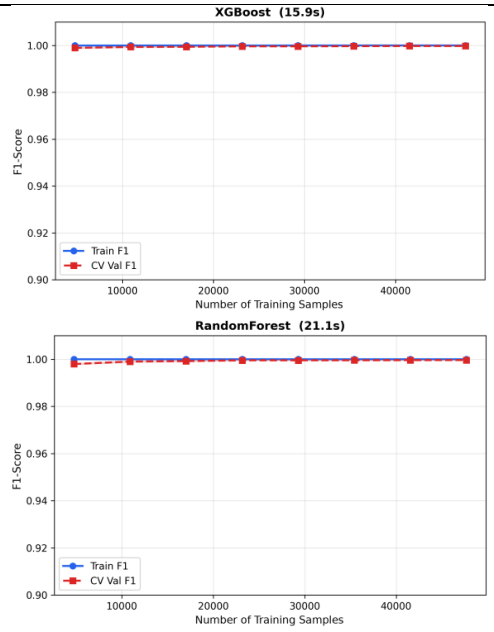

Fig. 5 - (a) F1-Score Learning Curve of the XGBoost Model, (b) F1-Score Learning Curve of the RandomForest Model

3.5 Feature Importance Interpretation Analysis (SHAP)

Since XGBoost was identified as preferable for live deployment based on inference efficiency (Section 3.3), SHAP interpretation was conducted specifically on the trained XGBoost model. The SHAP Summary Plot and Feature Importance Bar Plot are presented in Figure 6.

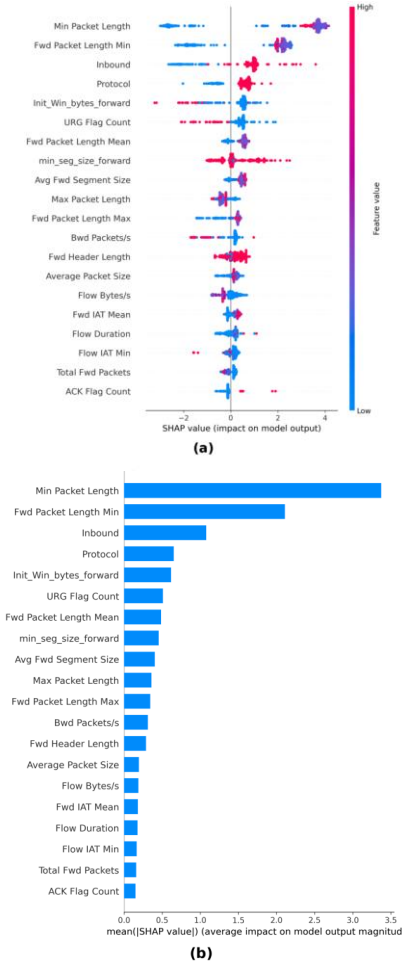

Fig. 6 - (a) SHAP Summary Plot and (b) SHAP Feature Importance Bar Plot of the XGBoost Model's Feature Contributions

Min Packet Length is the most dominant predictor, with a higher minimum packet length consistently associated with an increased ATTACK prediction; this association is a model-learned statistical pattern rather than direct evidence of a specific attack mechanism, though plausible given that reflection/amplification-based volumetric DDoS traffic has been empirically shown to exhibit characteristic packet-size distributions distinct from benign traffic [26]. Fwd Packet Length Min provides the second-largest contribution, following a similar directional pattern. Inbound (traffic direction) is the third most influential feature, with high Inbound values pushing predictions toward ATTACK. Protocol-related information also contributes substantially to the classifier's decision boundary, with UDP-associated observations tending to receive higher attack attribution in the evaluated sample, consistent with the predominance of UDP-based attack traffic in CIC-DDoS2019; this reflects a learned statistical association rather than a claim that the UDP protocol itself is causally responsible. Init_Win_bytes_forward completes the top five. SHAP values identify statistical associations exploited by the model rather than causal relationships between features and underlying attack mechanisms.

3.6 Live-Traffic IDS Prototype Testing on a Physical Network and False Positive Analysis

IDS prototype testing was conducted directly on a local physical Wi-Fi interface to test model reliability beyond laboratory simulation. Testing evaluated natural ambient background traffic and controlled hping3 SYN-flood attack traffic. False-positive results per ambient traffic category are presented in Table 13.

Table 13 - FPR Evaluation per Ambient Traffic Category (Physical Network)

Traffic Category	Total	FP	FPR
SSDP/UPnP (239.255.255.250:1900)	19	19	100.00%
mDNS (224.0.0.251:5353)	42	3	7.14%
HTTP :80 (Controlled Traffic)	284	0	0.00%
HTTPS :443 (Background Web)	31	1	3.23%
Broadcast Subnet/Global	15	2	13.33%
Uncommon Port/Host (53800, 1.1.1.1:53, 27017)	6	2	33.33%
Ambient-Traffic Subset FPR	397	27	6.80%

Table 14 - XGBoost Predicted Traffic Classification During the Controlled SYN-Flood Experiment, per Minute

Min.	Time	Dur.(s)	Flow	Pred.ATK	Pred.BEN	%ATK
1	17:02	26	987	120	867	12.16%
2	17:03	60	29,308	831	28,477	2.84%
3	17:04	60	21,786	1,438	20,348	6.60%
4	17:05	60	23,499	1,684	21,815	7.17%
5	17:06	60	23,439	1,894	21,545	8.08%
6	17:07	34	11,743	968	10,775	8.24%
Total	5 min	300	110,762	6,935	103,827	6.26%

Note. The experiment ran for exactly 300 seconds (17:02:34–17:07:34 WIB); the Duration (s) column shows each row's coverage within its clock-minute bucket. Minutes 1 and 6 are partial (26 s and 34 s respectively) because the test started and ended mid-minute, which is why their flow counts and %ATTACK values diverge from the full-minute rows.

The physical test identified an important phenomenon: the prototype captured 110,762 flows and classified 6,935 as ATTACK (6.26%) during the five-minute SYN-flood window, with the per-

minute ATTACK share ranging from 2.83% to 12.16% (Table 14) — a proportion distinct from the ambient-traffic FPR below since it includes genuine attack traffic rather than confirmed-benign flows alone. Because the experiment contained simultaneous background traffic and no independent packet/flow-level ground truth was performed for this window, these predictions are reported as classification outputs rather than a direct accuracy estimate. Ambient flows were categorized as known-benign by experimental control (no attack tool active during capture), not per-flow verification; FPR was calculated as FP/(FP+TN). Evaluation revealed an overall FPR of 6.80% (95% Wilson CI: 4.72–9.71%, n = 397), where SSDP/UPnP (FPR 100%, n = 19) remained almost entirely misclassified as ATTACK, while mDNS (FPR 7.14%, n = 42) was classified correctly in the large majority of cases. Several per-category sample sizes are small (e.g., SSDP/UPnP: 95% CI 83.18–100%; Uncommon Port/Host: 33.33%, n = 6, 95% CI 9.68–70.00%), so category-level percentages should be interpreted with caution; the overall ambient FPR (n = 397) is the more statistically stable summary.

This residual misclassification is consistent with the limited representation in the training dataset (CIC-DDoS2019) of short-duration UDP control packets. The model appears to associate short-duration UDP packets with small-scale UDP floods, an effect most pronounced for SSDP/UPnP, whose packets resemble the volumetric UDP-flood pattern on two dominant SHAP features (Min Packet Length and Protocol); mDNS shares some of these characteristics but is distinguished correctly in most instances. HTTP/HTTPS traffic had much lower FPR (0.00% and 3.23%) since its pattern more closely resembles legitimate training data. These findings motivate a hybrid rule-based filtering mechanism with context-aware preprocessing for known local-control traffic such as SSDP/UPnP, rather than blanket exclusion, since such traffic could be attacker-leveraged, consistent with standard rule-based network-filtering practice.

Table 15 - Comparison of This Study with Prior Studies

Study	Model	Dataset	XAI	Phys.	Distinguishing Position
Erdas [1]	RF/DT/ KNN/ SVM	CICIDS2017/ DDoS2019	No	No	No interpretability
Assadhan et al. [13]	LSTM+ SHAP	CIC-DDoS2019	Yes	No	Complex model; offline
Najar & Naik [14]	CNN (Inception)	CIC-DDoS2019	No	No	High accuracy, no XAI
Madoune et al. [27]	RF + XGBoost ensemble	SDN traffic	No	No	99.93% ensemble accuracy, no XAI
This Study	XGBoost & RF + SHAP	CIC-DDoS2019	Yes	Yes	Efficiency, XAI, real-world validation

Table 15 positions this study relative to four representative prior and contemporaneous works. Erdas et al. [1] achieved competitive performance using classical algorithms but provided neither interpretability nor physical-network validation. Assadhan et al. [13] incorporated SHAP-based interpretability but relied on comparatively complex architectures evaluated purely offline. Najar and Naik [14] achieved high CNN-based detection accuracy on the CICDDoS2019 dataset but did not incorporate explainability. Madoune et al. [27] reported a high RF+XGBoost ensemble accuracy (99.93%) on SDN traffic, but the work lacked explainability and was not validated on a physical network. This study jointly combines high-efficiency ensemble classification, SHAP-based explainability, and direct live-traffic validation across separate attacker and victim devices on a physical network, addressing both the black-box limitation and the offline-to-physical implementation gap identified in Section 1.

4. Conclusion

The experimental results demonstrate that both XGBoost and Random Forest achieve near-perfect binary DDoS classification performance on CIC-DDoS2019 under a session-disjoint evaluation protocol, with XGBoost obtaining an F1-score of 0.9999 and ROC-AUC of 1.0000 on the held-out test set. XGBoost also achieved a substantially higher model-inference throughput of 1,819,816 flows/second (0.55 µs/flow, 3.69× higher than Random Forest). SHAP analysis further identified Min Packet Length, Fwd Packet Length Min, Inbound, Protocol, and Init_Win_bytes_forward as the features most strongly associated with the XGBoost model's predictions.

However, the physical-network experiment qualifies these benchmark results. As detailed in Section 3.6, live testing revealed a measurable false-positive rate concentrated in ambient UDP control traffic (notably SSDP/UPnP), an estimate that should be treated as preliminary given the limited size of the benign subset evaluated. The negligible train–test gap under session-disjoint evaluation shows excellent within-dataset generalization, but this does not by itself establish generalization to operational network conditions. The practical implication is that DDoS intrusion detection evaluation should not rely on benchmark accuracy alone: session-disjoint testing, explainability analysis, inference benchmarking, and physical-network validation each provide complementary, partially independent evidence about model reliability, and no single metric is sufficient on its own.

This study has several limitations: the training dataset originates solely from CIC-DDoS2019 and does not yet cover the full variety of modern normal traffic; physical testing was conducted at a two-host laboratory scale as a proof-of-concept rather than a production-scale deployment; the SYN-flood physical validation covered a single attack type rather than the full range of DDoS vectors in the dataset; the inference-speed benchmark reflects one host configuration, untested across heterogeneous hardware; the SHAP interpretability analysis was computed on a stratified subset of 500 test samples rather than the full test set, a size chosen for computational tractability, so the identified feature-importance patterns should be read as representative of general model behavior rather than an exhaustive account of rarer feature interactions; and the evaluation focused on binary classification, so multi-class attack-type identification was not evaluated.

Future work should enrich the dataset with ambient UDP control traffic samples (DHCP, mDNS, SSDP, NTP), apply hybrid rule-based pre-filtering, test the IDS on edge devices (Raspberry Pi, Nvidia Jetson), and perform post-attack network forensics. A naive static rule (e.g., allow-listing traffic by destination port or multicast address alone) risks creating a predictable blind spot that an attacker could exploit to evade detection; any such pre-filtering should instead be context-aware, incorporating signals such as packet rate, source diversity, burst duration, and host role rather than protocol or port identity alone. It should also validate robustness via k-fold cross-validation, evaluate a reduced feature set from top-ranked SHAP

features for lower inference cost, and apply automated hyperparameter optimization (e.g., RandomizedSearchCV or Optuna) instead of the empirical settings used here.

REFERENCES

[1] S. Erdas, A. E. Akkaya, and A. A. Aydin, "Machine learning based hybrid DDoS attack prediction," *European Journal of Technique*, vol. 15, no. 2, pp. 45–56, 2025, doi: 10.36222/ejt.1670798.

[2] B. K. Mohammed and H. H. Khayoon, "A critical theoretical comparison of DoS and DDoS attacks: Detection and defense perspectives," in *Proc. 2026 2nd Int. Conf. Computing and Emerging Sciences, ACM*, 2026, pp. 1–8. doi: 10.1145/3797491.3797497.

[3] A. H. Ali *et al.*, "Unveiling machine learning strategies and considerations in intrusion detection systems: A comprehensive survey," *Front. Comput. Sci.*, vol. 6, p. 1387354, 2024, doi: 10.3389/fcomp.2024.1387354.

[4] A. Momand, S. U. Jan, and N. Ramzan, "A systematic and comprehensive survey of recent advances in intrusion detection systems using machine learning: Deep learning, datasets, and attack taxonomy," *J. Sens.*, vol. 2023, p. 6048087, 2023, doi: 10.1155/2023/6048087.

[5] N. Sharma and B. Arora, "Machine learning and deep learning models for anomaly intrusion detection in networks: A systematic review," *SN Comput. Sci.*, vol. 6, p. 832, 2025, doi: 10.1007/s42979-025-04352-z.

[6] T. H. Sow and M. Adda, "Enhancing IDS performance through a comparative analysis of Random Forest, XGBoost, and Deep Neural Networks," *Machine Learning with Applications*, vol. 22, p. 100738, 2025, doi: 10.1016/j.mlwa.2025.100738.

[7] O. Achbarou, T. Datsi, O. Bourkoukou, and A. My El Kiram, "Enhanced intrusion detection system using feature selection and hybrid learning models for high performance and efficiency in an IoT environment," *Journal of Engineering Research*, 2025, doi: 10.1016/j.jer.2025.10.016.

[8] N. Moustafa, N. Koroniotis, M. Keshk, A. Y. Zomaya, and Z. Tari, "Explainable intrusion detection for cyber defences in the internet of things: Opportunities and solutions," *IEEE Communications Surveys & Tutorials*, vol. 25, no. 3, pp. 1775–1807, 2023, doi: 10.1109/COMST.2023.3280465.

[9] M. T. Islam, M. K. Syfullah, M. G. Rashed, and D. Das, "Bridging the gap: Advancing the transparency and trustworthiness of network intrusion detection with explainable AI," *International Journal of Machine Learning and Cybernetics*, vol. 15, no. 11, pp. 5337–5360, 2024, doi: 10.1007/s13042-024-02242-z.

[10] P. Hermosilla, S. Berríos, and H. Allende-Cid, "Explainable AI for forensic analysis: A comparative study of SHAP and LIME in intrusion detection models," *Applied Sciences*, vol. 15, no. 13, p. 7329, 2025, doi: 10.3390/app15137329.

[11] V. Z. Mohale and I. C. Obagbuwa, "A systematic review on the integration of explainable artificial intelligence in intrusion detection systems to enhancing transparency and interpretability in cybersecurity," *Front. Artif. Intell.*, vol. 8, 2025, doi: 10.3389/frai.2025.1526221.

[12] J. Hesford, "Expectations versus reality: Evaluating intrusion detection systems in practice," in *Proc. 55th Annual IEEE/IFIP Int. Conf. Dependable Systems and Networks — Supplemental Volume (DSN-S)*, 2025, pp. 56–62. doi: 10.1109/dsn-s65789.2025.00042.

[13] B. Assadhan, A. Bashaiwth, and H. Binsalleeh, "Enhancing explanation of LSTM-based DDoS attack classification using SHAP with pattern dependency," *IEEE Access*, vol. 12, pp. 90707–90725, 2024, doi: 10.1109/access.2024.3421299.

[14] A. A. Najar and S. M. Naik, "A robust DDoS intrusion detection system using convolutional neural network," *Computers and Electrical Engineering*, vol. 117, 2024, doi: 10.1016/j.compeleceng.2024.109277.

[15] H. Sharif, "A machine learning-based approach for the detection of DDoS attacks on the Internet of Things using CICDDoS2019 dataset - PortMap," *Lahore Garrison University Research Journal of Computer Science and Information Technology*, vol. 8, no. 2, pp. 19–30, 2024, doi: 10.54692/lgurjcsit.2024.082569.

[16] Y. Hu, K. Xiao, L. Luo, and L. Chen, "An XGBoost-based intrusion detection framework with interpretability analysis for IoT networks," *Applied Sciences*, vol. 16, no. 2, p. 980, 2026, doi: 10.3390/app16020980.

[17] Z. Zhang, S. Kong, T. Xiao, and A. Yang, "A network intrusion detection method based on bagging ensemble," *Symmetry (Basel).*, vol. 16, no. 7, p. 850, 2024, doi: 10.3390/sym16070850.

[18] S. Sambangi, L. Gondi, and S. Aljawarneh, "A feature similarity machine learning model for DDoS attack detection in modern network environments for Industry 4.0," *Computers & Electrical Engineering*, vol. 100, p. 107955, 2022, doi: 10.1016/j.compeleceng.2022.107955.

[19] D. Akgun, S. Hizal, and U. Cavusoglu, "A new DDoS attacks intrusion detection model based on deep learning for cybersecurity," *Comput. Secur.*, vol. 118, 2022, doi: 10.1016/j.cose.2022.102748.

[20] J. Shaikh, Y. A. Butt, and H. F. Naqvi, "Effective intrusion detection system using deep learning for DDoS attacks," *Asian Bulletin of Big Data Management*, vol. 4, no. 1, pp. 168–183, 2024, doi: 10.62019/abbdm.v4i1.113.

[21] M. A. Bouke and A. Abdullah, "An empirical study of pattern leakage impact during data preprocessing on machine learning-based intrusion detection models reliability," *Expert Syst. Appl.*, vol. 230, p. 120715, 2023, doi: 10.1016/j.eswa.2023.120715.

[22] A. Luque, A. Carrasco, A. Martín, and A. de las Heras, "The impact of class imbalance in classification performance metrics based on the binary confusion matrix," *Pattern Recognit.*, vol. 91, pp. 216–231, 2019, doi: 10.1016/j.patcog.2019.02.023.

[23] S. Sathyanarayanan and B. Roopashri Tantri, "Confusion matrix-based performance evaluation metrics," *African Journal of Biomedical Research*, vol. 27, no. 4s, pp. 4023–4031, 2024, doi: 10.53555/ajbr.v27i4s.4345.

[24] S. Dash and J. M. Acken, "Intrusion detection latency: the neglected metric," *Cybersecurity*, vol. 9, art. 144, 2026, doi: 10.1186/s42400-026-00574-7.

[25] B. Rozemberczki, "The Shapley value in machine learning," in *Proc. 31st Int. Joint Conf. Artificial Intelligence (IJCAI-22), Survey Track*, 2022, pp. 5572–5579. doi: 10.24963/ijcai.2022/778.

[26] M. Anagnostopoulos, S. Lagos, and G. Kambourakis, "Large-scale empirical evaluation of DNS and SSDP amplification attacks," *Journal of Information Security and Applications*, vol. 66, p. 103168, 2022, doi: 10.1016/j.jisa.2022.103168.

[27] S. A. Madoune *et al.*, "A novel approach for real-time DDoS detection in SDN using dimensionality reduction and ensemble learning," *Journal of Information Security and Applications*, vol. 94, 2025, doi: 10.1016/j.jisa.2025.104195.