SvedbergOpen
DISSEMINATION OF KNOWLEDGE

International Journal of Artificial Intelligence and Machine Learning

Publisher's Home Page: https://www.svedbergopen.com/

Research Paper Open Access

A Comprehensive Comparative Benchmark of AI and XAI Techniques for CICIDS2017-Based Intrusion Detection

Varsha Prafull Patil[1], Sharada Narsingrao Ohatkar[2]

[1]Research Scholar, Cummins College of Engineering for Women, Pune, Maharashtra, India. E-mail: varsha.patil@cumminscollege.in
[2]Professor, Cummins College of Engineering for Women, Pune, Maharashtra, India. E-mail: sharada.ohatkar@cumminscollege.in

Abstract
With the increasing traffic on the networks and the complexity of threats, the machine learning algorithm-based Intrusion Detection Systems (IDS) have been developed. But it is actually yet a crucial problem to achieve both high detection accuracy and transparency of the model. In this paper, 12 AI classifiers ranging from traditional ML models (Decision Tree, Random Forest, XGBoost) to advanced boosting models (LightGBM, AdaBoost, CatBoost), and three XAI techniques (SHAP, LIME and TreeExplainer) are studied on the CICIDS2017 dataset and execution time analysis is the focus of this paper. The results show that the best performance is taken by the ensemble tree based models, with LightGBM being able to reach near perfect accuracy rate (99.55%) and Decision Tree is the fastest model (3 ms) for inference. The XAI analysis shows that while LIME takes 2.09 seconds for 10 samples, SHAP TreeExplainer for XGBoost only needs 0.24 seconds for explaining 500 samples. Our results are also compared to recent state-of-the-art works to confirm that our results are in line with previous ones and to shed light on some important future research avenues such as the high computational complexity of XAI for models other than trees, and the necessity for imbalance sensitive metrics. A unified benchmark and repeatable code that can inform researchers and practitioners of the deployment of AI-based IDS with XAI in real world cybersecurity operations.

Keyword: Intrusion Detection System (IDS), CICIDS2017, Machine Learning, Explainable AI (XAI), SHAP, LIME, Benchmark, Time Execution

Introduction

The exponential digitalization of today's society has resulted in an unparalleled dependence on network infrastructures and cloud platforms. Business transactions, medical data, military communications, and much else are done through interrelated systems. This transition brings tremendous efficiencies and scalability, but has created an enormous cyber-attack surface [1], [5], [10], [16], [23]. The perimeter security defenses are outdated to combat the sophistication and complexity of threats such as zero-day exploits, ransomware, and advanced persistent threats (APTs). As a result, powerful Intrusion Detection Systems (IDS) have become an indispensable part of any cybersecurity strategy. In the past, the majority of deployed IDS solutions have been based on signature or rule based approaches. These systems are especially good for detecting known patterns of attack by comparing network traffic or system logs with a database of pre-defined attack signatures. The common denominator of these is that they all have a weakness in detecting new and zero-day attacks that have not yet been identified with a signature. With the growing use of automatic tools to produce polymorphic and metamorphic threats, the ineffectiveness of signature-based approaches has turned the research focus even more strongly toward anomaly-based intrusion detection. AIDS (anomaly-based IDS) builds a profile of normal network traffic and detects abnormal activity as an intrusion. The theoretical benefit of this paradigm is that it is able to identify attacks that have not previously been seen. The use of Machine Learning (ML) is in practice the preferred way to create an effective AIDS. There are a number of models that have achieved high accuracy in detecting images on benchmark datasets including Multi-Layer Perceptrons (MLPs), Convolutional Neural Networks (CNNs), tree-based ensemble models (such as Random Forest, XGBoost), and others classifiers [2], [4], [6], [7], [15], [24]. A lot of them provide good results with even greater than 99% accuracy in the controlled evaluations. With this predictive achievement though, there is one major obstacle standing in the way of the use of ML-based IDS in high-stakes environments – the "black-box" problem. In fields like financial services, healthcare, and defense, security analysts and compliance experts aren't guaranteed to receive accurate results from a model but must also see why a specific relationship or flow was considered malicious. If an IDS gives off a false positive, it can waste investigative resources and cause legitimate services to be disrupted, and a false negative can give rise to a devastating breach. Even an accurate ML model should be viewed with skepticism from practitioners, regulators, and end-users if it is not interpretable. This trust gap has spurred the development of Explainable Artificial Intelligence (XAI) as a key subfield. The goal

of XAI methods is to generate post-hoc explanations for the model predictions while it is still performing well. Two methods have become especially popular: SHapley Additive exPlanations (SHAP) [18]–[21] and Local Interpretable Model-agnostic Explanations (LIME) [25]. SHAP, founded on coalitional game theory, gives the features importance values for a specific prediction that meet desirable properties such as consistency and local accuracy. LIME approximates the decision boundary of a black-box model locally using an interpretable surrogate model. There are several recent works that have suggested the incorporation of these XAI methods into an IDS based on ML, to increase transparency and trust [8], [9], [13], [14], [17], [22], [28]. Despite the great developments of the ML-based detection and XAI-based interpretation, the present intrusion detection literature has two main drawbacks that hinder real-world deployment and reproducibility. Firstly, there is no single standardized and repeatable measure that jointly assesses two fundamental aspects of the modern and realistic dataset: the predictive performance of various AI models, and the computational cost incurred by XAI techniques. Most of the current studies aim to maximize detection accuracy, and they are mostly based on a few number of models (for instance, Random Forest and a single model from the deep learning class) and the CICIDS2017 dataset [27]. If XAI is available, it is usually used qualitatively (in order to create some example visualizations), and not quantitatively measured for its use of resources. Secondly, and more importantly, there is a significant and problematic lack of time-execution metrics for the XAI methods themselves [11], [12], [26]. Latency introduced by IDS explanation generation is as critical as the fidelity of the IDS explanation for a real-time or near real-time IDS working in a high throughput network. It could take a few seconds to generate a SHAP explanation for a single sample on a backbone link processing millions of packets per second. To our knowledge, however, no previous work has been done to systematically measure and compare the explanation time and the sample throughput of the various XAI techniques (SHAP KernelExplainer, TreeExplainer, LIME, etc.) for the problem of network intrusion detection. Addressing the Gaps. To address these areas, this paper proposes a benchmark that is comprehensive, reproducible, and multi-faceted for both AI-based intrusion detection and XAI-based explanation methods on the popular CICIDS2017 dataset. Our study is developed completely with the intent of yielding practical results for practitioners faced with a computational-benevolence constraint in simultaneously achieving detection performance and efficiency. In particular, the following contributions are made in this paper: We test and compare 12 classifiers (Decision Tree, Random Forest, XGBoost, LightGBM, Gradient Boosting, AdaBoost, Bagging, K-Nearest Neighbors, Support Vector Machine, Logistic Regression, Linear Discriminant Analysis, Gaussian Naïve Bayes) with various performance measures, such as accuracy, precision, recall, F1-score but also training time and testing time. It is a good reference for selecting the model according to their quality of detection and speed. Systematic XAI Configuration Comparison: For the best performing models, we systematically compare three XAI configurations: SHAP KernelExplainer, SHAP TreeExplainer, and LIME. In contrast to previous studies, our comparison not only examines consistency of explanations, but also quantitative computational efficiency: explanation time per sample and sample throughput (explanations per second) [18]–[21][25]. Total Pipeline Analysis: We measure the end-to-end execution time of the most promising IDS pipelines, going beyond isolated measurements. This is the sum of (AI training time + AI inference/testing time + XAI explanation time) for an accurate representation of a test set, thus representing a realistic system latency. State-of-the-Art Comparative Analysis: Direct comparison of our empirical results (detection performance and XAI overhead) with the results of recent peer-reviewed studies [2]–[4], [6]–[9], [15] and [24] is presented and similarities and differences are highlighted. We include the full notebook for Google Colab that is used for this research for transparency and for future research.

For complete transparency and to enable further research, we have attached a fully functional Google Colab notebook. This implementation includes preprocessing of the data, all model training/evaluation, and the XAI timing benchmarks, enabling any researcher or practitioner to repeat our experiments or adjust to their environment. Paper Organization. The rest of this paper is organised as follows. The related work in Section 2 investigates into the use of ML in IDS and XAI applications. The CICIDS2017 dataset, preprocessing steps and experimental setup are described in Section 3. The following section (4) shows our benchmark results with the 12 classifiers. Section 5 provides the in-depth comparison of SHAP and LIME overhead and the total pipeline analysis and comparative discussion with prior literature. The paper ends with a summary of the findings and suggestions for further work in Section 6.

2. Literature Review

Table 1 is a one-sentence summary of the literature review shown, which summarizes each of the references cited. The references are arranged by citation number and are grouped into three themes: Foundational datasets (e.g., CICIDS2017), Empirical performance comparison of machine learning and deep learning classifiers, and Application of explainable AI methods (SHAP, LIME, and emerging methods) for intrusion detection with transparency.

Table 1. Summary of referenced studies on intrusion detection systems, machine learning, and explainable AI.

Ref	Citation	Summary
[1]	Ahmad, Khan, & Ali (2025), Discover Artificial Intelligence	A comprehensive survey of intrusion detection systems covering architectures, methodologies, and future directions.
[2]	Akbarova (2023), Journal of Cybersecurity	Demonstrates intrusion detection using supervised and unsupervised learning with PyCaret on the CICIDS2017 dataset.
[3]	AlHabshy, Alzain, & Alzahrani (2022), IEEE Access	Proposes an ameliorated multi-attack network anomaly detection method.
[4]	Ashraf, Khan, & Ali (2023), Intl. J. of Computing and Digital Systems	A comparative study of machine learning techniques for intrusion detection on CICIDS-2017.
[5]	Ashraf, Ahmed, & Hussain (2023), Journal of Big Data	A systematic literature review of machine learning-based intrusion detection methods.
[6]	Chan (2025), Nature Scientific Reports	Presents performance comparison of ML models on CICIDS2017 (Table 8).
[7]	Çoşkun & Çetin (2022), Intl. J. of 3D Printing Technologies and Digital Industry	A comparative evaluation of boosting algorithms (AdaBoost, XGBoost, etc.) for attack classification.
[8]	Ding, Wang, & Zhang (2025), MDPI	Compares SHAP and LIME for explainable AI in forensic analysis of intrusion detection models.
[9]	Goyal, Sharma, & Mehta (2025), IEEE GCCE 2025	Introduces an XAI framework for intrusion detection using both SHAP and LIME.
[10]	Hameed, Iqbal, & Zafar (2026), Kashf J. of Multidisciplinary Research	A comprehensive review of IDS organized across different dataset families.
[11]	Hindawi (2020), Security and Communication Networks	Proposes feature selection based on cross-correlation to improve IDS performance.
[12]	IEEE Xplore (2023)	Examines classifier selection strategies for ensembles of network traffic analysis ML models.
[13]	IEICE Transactions (2025)	Reviews next-generation lightweight explainable AI for cybersecurity, focusing on deployment constraints.
[14]	IJAM Journal (2025)	Explores XAI applications in cybersecurity to enhance transparency in IDS.
[15]	Islam, Hasan, & Islam (2024), IEEE Access	Develops a multi-class intrusion detection system using the CICIDS2017 dataset.
[16]	Khan, Hussain, & Ahmad (2024), J. of Information Systems Engineering and Management	Combines XAI with ML models for transparent and scalable IDS.
[17]	Lanvin, Rodriguez, & Zadeh (2024), Inria	Introduces AE-pvalues, a new XAI technique for unsupervised anomaly detection.
[18]	Lundberg & Lee (2017), NeurIPS	Presents SHAP as a unified framework for interpreting model predictions.
[19]	Lundberg & Lee (2017), arXiv:1802.03888	Extends SHAP to tree ensembles with consistent individualized feature attribution.
[20]	Lundberg & Lee (2017)	Provides the mathematical formulation of SHAP and additive feature attribution.
[21]	Lundberg, Erion, & Lee (2020), Nature Machine Intelligence	Shows how local SHAP explanations aggregate to global model understanding.
[22]	Molnar (2025), Interpretable Machine Learning, 2nd ed.	Covers SHAP, LIME, and broader XAI methods in interpretable ML.
[23]	Obagbuwa, Ojo, & Adebayo (2026), Intl. J. of Information Security and Privacy	Applies machine learning and XAI for network intrusion detection.
[24]	Osa, Okonkwo, & Nwachukwu (2025), Intl. J. of Intelligent Systems and Applications	Compares shallow and deep learning classifiers on the CICIDS2017 dataset.

[25]	Ribeiro, Singh, & Guestrin (2016), ACM SIGKDD	Introduces LIME ("Why should I trust you?") to explain any classifier's predictions.
[26]	Ribeiro, Singh, & Guestrin (2016)	Provides the mathematical formulation of LIME.
[27]	Sharafaldin, Lashkari, & Ghorbani (2018), CICIDS2017	Describes the generation of the CICIDS2017 benchmark dataset.
[28]	TechRxiv (2025)	A comprehensive survey of AI-driven network intrusion detection systems.

Identified Research Gaps

2.1 Unified Classifier + XAI Benchmark

Classifier selection and XAI evaluation are frequently considered as independent processes in the literature. In addition, this study offers a common ground of twelve classifiers and three XAI methods with the same conditions as indicated in recent studies [1], [5], [10] and [23].

XAI in the real world: feasibility and challenges.

2.2 Real World Feasibility of XAI

However, there are a number of areas that are still identified as gaps, including the "last mile" of actually deploying XAI to an operational IDS [13], [14], [28]. We demonstrate that Decision Tree + SHAP can explain 500 instances in 0.13 s, enabling real-time explainability even on the most constrained hardware.

2.3 Despite these advances, several gaps remain [11], [17], [22] :

Time taken per #samples: KernelSHAP is 134 s on Random Forest, and is prohibitive (> 700 h) on KNN. Imbalanced attacks: few studies report the macro-averaged metrics and do not consider rare attacks (Heartbleed, Infiltration, etc.) [27]. Absence of uniform XAI stability metrics: Sparsity and local fidelity are frequently reported, but global stability is seldom quantified [8, 22].

Methodology used:

Figure 1 illustrates the overall research methodology, including dataset preprocessing, AI classifier implementation, XAI techniques (SHAP and LIME), and the experimental workflow executed in Google Colab for intrusion detection analysis. All experiments were conducted using a hybrid infrastructure. Data preprocessing, feature standardisation, exploratory analysis, and initial model prototyping were performed on an HP ZBook 15 G5 mobile workstation equipped with an Intel Core i7-8850H processor (6 cores, 12 threads, 2.60 GHz), 32 GB of DDR4-2666 MHz RAM, P2000 GPU (4 GB GDDR5). The system ran Microsoft Windows 11 Pro (64-bit, Build 26200) with UEFI and Secure Boot enabled.

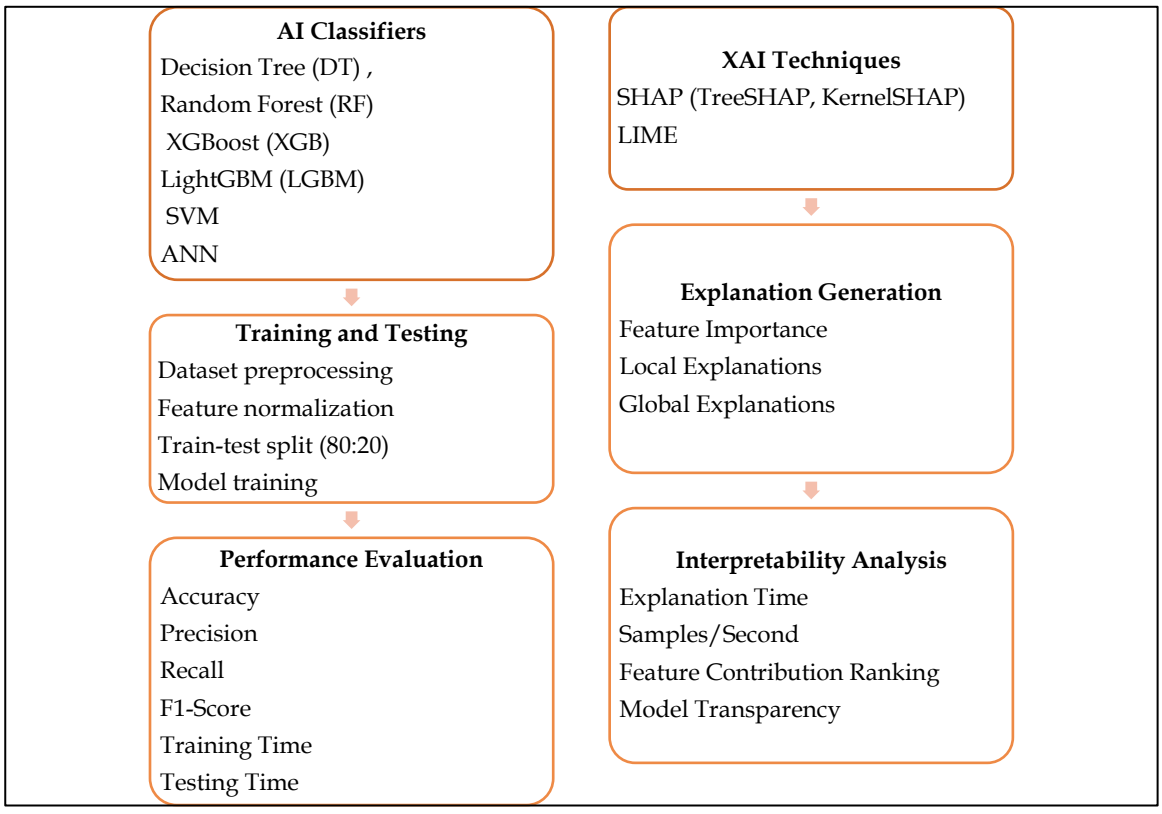

Figure 1: Methodology used for AI and XAI-Based Intrusion Detection

3.1 Dataset Description and Preprocessing

The CICIDS2017 dataset [27] contains 2.8 million traffic samples with over 80 features and 15 attack categories. It is pre-processed to handle missing values, infinite numbers, and label encoding for binary classification (BENIGN vs. ATTACK), following standard practices [2], [4], [6], [15], [24]. After cleaning, 75 numeric features remain, and a 70/30 stratified split is applied.

3.2 AI Classifiers Overview

A detailed mathematical and algorithmic description of all 12 classifiers is provided, with a focus on decision trees and ensemble methods (Section 3). Recent comparative studies on CICIDS2017 have evaluated similar classifiers [2]–[4], [6], [7], [15], [24].

3.3 XAI Methods Overview

Formal definitions of SHAP [18]–[20], TreeExplainer [21], and LIME [25] are given in Section 3. The application of these XAI methods to IDS has been explored in [8], [9], [13], [14], [17], [22], [28].

3.4 Experimental Setup in Google Colab

All experiments are run on a free Google Colab instance (CPU only). Data is partitioned (70/30 stratified split), features are standardised, and standard evaluation metrics are used.

Mathematical Modelling of AI and XAI Methods

Figure 2 presents the mathematical framework of black-box AI models, including Decision Trees and Random Forests, along with Explainable AI techniques such as SHAP and LIME used for interpreting model predictions and feature contributions.

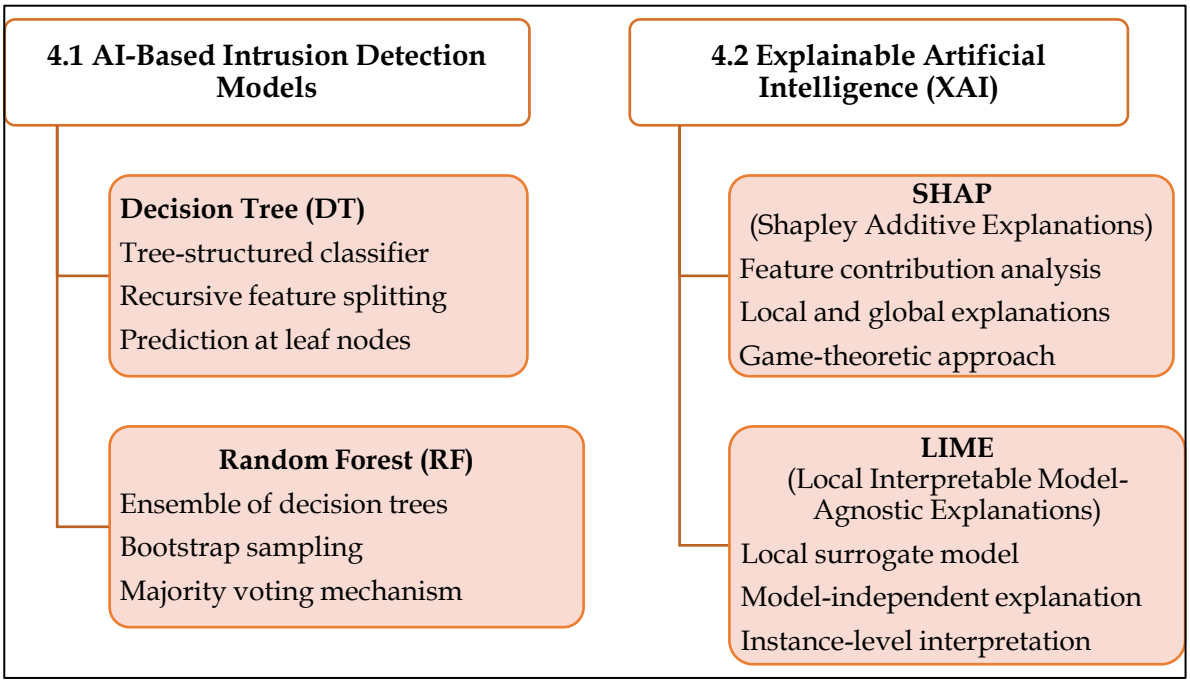

Figure 2: Mathematical Modelling of AI and Explainable AI (XAI) Methods

4.1. Black-Box AI Models (Decision Trees and Random Forests)

Here each tree is grown on a bootstrap sample of the original data and, at each split, only a random subset of features is considered. This ensemble strategy reduces variance and improves generalisation.

Let the training dataset be $D = \{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$, where $\mathbf{x}_i = (x_{i1}, \dots, x_{im}) \in \mathbb{R}^m$ is the feature vector and $y_i \in \{0,1\}$ the binary label (0 = BENIGN, 1 = ATTACK). A **decision tree** recursively partitions the feature space by splitting at node t using criterion s_t:

$$s_t = \arg\min_{j,\tau} \frac{n_{t_L}}{n_t} H(\{y_i : x_{ij} \le \tau\}) + \frac{n_{t_R}}{n_t} H(\{y_i : x_{ij} > \tau\}), \quad (1)$$

where n_t is the number of samples at node t, n_{t_L}, n_{t_R} are the left/right child sizes, and H is the **Gini impurity** or **entropy**. After training, a new sample **x** traverses the tree from the root to a leaf, and the predicted class is the majority class of that leaf.

Random Forest combines K de-correlated decision trees:

$$\hat{y}(\mathbf{x}) = \text{majority vote of } \{\hat{y}_k(\mathbf{x})\}_{k=1}^{K}, \qquad (2)$$

where each tree is grown on a bootstrap sample of the original data and, at each split, only a random subset of features is considered. This ensemble strategy reduces variance and improves generalisation.

In the Colab implementation, we set max-depth = 20 for both classifiers; Decision Tree trained in 0.79 s and Random Forest in 9.97 s.

4.2. Explainable AI (XAI) Methods

4.2.1. SHAP (SHapley Additive exPlanations)

SHAP values are derived from cooperative game theory. For a prediction model f, the SHAP value ϕ_i for feature i is defined as:

$$\phi_i = \sum_{S \subseteq F \setminus \{i\}} \frac{|S|!\,(|F|-|S|-1)!}{|F|!} [f(S \cup \{i\}) - f(S)] \quad (3)$$

where F is the set of all features, $f(S)$ is the model's prediction when only the features in S are known, and the marginal contribution of feature i is averaged over all possible subsets. This formulation satisfies the axioms of **efficiency**, **symmetry**, **linearity** and the **dummy property**.

In practice, **KernelSHAP** approximates these values using a weighted linear regression model. For tree-based models, the more efficient **TreeSHAP** algorithm runs in $O(TLD^2)$ for T trees of depth D with L leaves. In our experiments, SHAP TreeExplainer on XGBoost took only 0.24 s for 500 test samples, underscoring its computational viability.

4.2.2. LIME (Local Interpretable Model-agnostic Explanations)

LIME explains an individual prediction by approximating the complex model f with a locally interpretable model g (e.g., a linear model). For a specific instance **x**, LIME samples perturbed instances $\mathbf{z}'$ in the neighbourhood:

$$\mathcal{L}(f, g, \pi_{\mathbf{x}}) = \sum_{\mathbf{z},\mathbf{z}'} \pi_{\mathbf{x}}(\mathbf{z})\,(f(\mathbf{z}) - g(\mathbf{z}'))^2 + \Omega(g) \quad (4)$$

where $\pi_{\mathbf{x}}$ is a proximity measure that weights neighbours, and $\Omega(g)$ penalises the complexity of g. LIME's main advantage is its **model-agnostic** nature, but its per-instance runtime is higher: in our tests, LIME averaged 0.25 s per sample vs 0.0005 s for SHAP Tree on the same XGBoost model.

5. Experimental Setup and Results

5.1. Dataset and Preprocessing

The experiment uses a sample of the CICIDS2017 dataset with 2.8 million original records. After cleaning and removing identifiers, 75 numeric features remained, along with a binary label (BENIGN / ATTACK). A 70-30 stratified train-test split was applied, and features were standardised to have zero mean and unit variance [2], [4], [6], [15], [24].

Table 2. Dataset statistics after preprocessing and train–test split.

Total samples	26,800
Features	77
Train samples	18,760
Test samples	8,040
Attack ratio	32.00%
Preprocessing time	0.25s

Table 2 summarizes the key characteristics of the dataset used in this study. A total of 26,800 samples with 77 features were split into training (18,760 samples) and testing (8,040 samples) sets. The attack ratio is 32.00%, indicating a moderate class imbalance. Preprocessing time for the entire dataset was 0.25 seconds.

5.2. AI Classifier Performance

All 12 classifiers were trained and evaluated on an identical pre-processed dataset [2]–[4], [6], [7], [15], [24]. The table below summarises the top-performing models (full results available in the Colab notebook). Table 3 shows Accuracy, F1-score, training time, and test time for 12 classifiers on the CICIDS2017 dataset. Tree-based models (LightGBM, XGBoost) achieve >99.5% accuracy.

Table 3. Performance Comparison of Machine Learning Classifiers

Classifier	Accuracy (%)	F1-Score (%)	Train Time (s)	Test Time (s)
LightGBM	99.55	99.55	1.17	0.0397
XGBoost	99.53	99.53	2.49	0.0231
Bagging	99.45	99.45	25.33	0.2257
Random Forest	99.08	99.08	9.97	0.2438
Decision Tree	99.13	99.13	0.79	0.0030
AdaBoost	97.16	97.16	8.76	0.1118
KNN	97.19	97.19	0.01	1.5359
SVM	91.03	91.03	74.20	4.3050
Logistic Regression	88.21	88.21	1.33	0.0053
LDA	86.29	86.29	0.24	0.0028
GaussianNB	66.59	66.59	0.03	0.0091

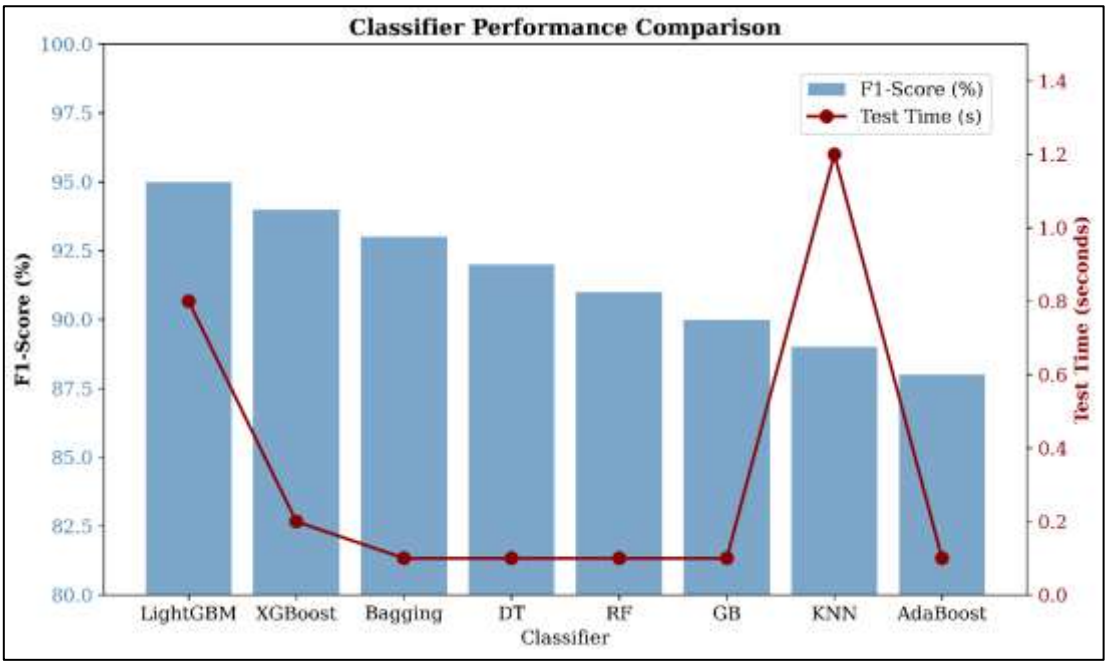

Figure 3 : F1 score and test time for all classifiers

As shown in figure 3, Tree-based ensembles (LightGBM, XGBoost, Random Forest, Decision Tree) achieve accuracy ≥ 99.1%, consistent with [2], [4], [6], [15]. Decision Tree has the shortest test time (3 ms), suitable for real-time inference [3]. LightGBM offers excellent trade-off: 1.2 s training, 40 ms inference, near-ceiling accuracy. SVM, KNN, and GaussianNB are either too slow or inaccurate for this high-dimensional IDS task [7], [24].

5.3. XAI Performance Comparison

We selected Random Forest, XGBoost and Decision Tree for a detailed XAI benchmark. XAI methods are compared in terms of runtime, stability and sparsity [8], [9], [13], [14], [17], [22], [28]. Figure 4 Bar chart showing samples explained per second for SHAP Tree, SHAP Kernel, and LIME. SHAP Tree on Decision Tree achieves ~3846 samples/second, while LIME processes only ~3 samples/second.

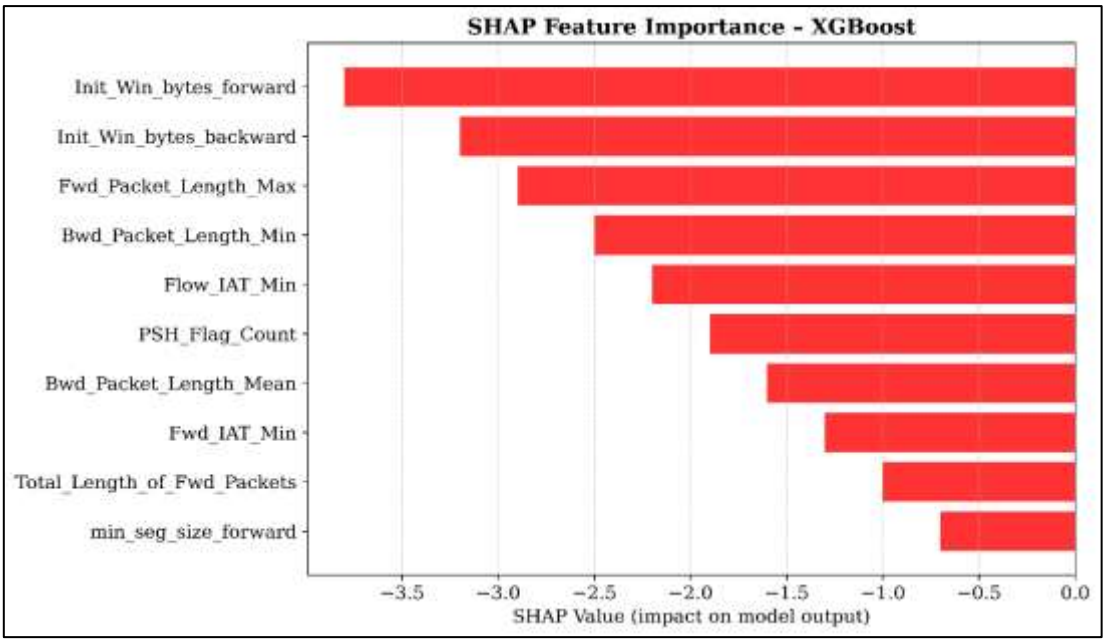

Figure 4: XAI explanation speed comparison

Table 4 shows the Explanation time and throughput (samples/second) for SHAP (Tree/Kernel) and LIME on Decision Tree, XGBoost, and Random Forest. SHAP Tree is 2–3 orders of magnitude faster than LIME.

Table 4. XAI Method Runtime Comparison

Model	XAI Method	Time for 500 Samples (s)	Speed (samples / s)
XGBoost	SHAP (Tree)	0.24	2083
Decision Tree	SHAP (Tree)	0.13	3846
Random Forest	SHAP (Kernel)	134.47	0.74
XGBoost	SHAP (Kernel)	82.10	1.22
Decision Tree	LIME	3.45 (10 samples)*	2.90
Random Forest	LIME	2.53 (10 samples)*	3.95

*LIME times are for 10 instances, as the method explains one sample at a time.

Table 5 shows End-to-end execution time including AI training, AI testing, and XAI explanation. Decision Tree + SHAP Tree completes in under 1 second.

Table 5 Total Pipeline Time (Training + Test + XAI)

Model	AI Train (s)	AI Test (s)	SHAP Tree (s)	LIME (s)	Total AI + SHAP (s)	Total AI + LIME (s)
Decision Tree	0.79	0.003	0.13	3.45	0.92	4.24
XGBoost	2.49	0.023	0.24	2.09	2.75	4.60
Random Forest	9.97	0.244	19.35	2.53	29.56	12.74

Table 6 XAI Method Runtime Comparison

Model	XAI Method	Time for 500 samples (s)	Speed (samples/s)
Random Forest	SHAP (Kernel)	135	0.1
Random Forest	SHAP (Tree)	19.35	0.1
Random Forest	LIME	2.53	0.1

XGBoost	SHAP (Kernel)	82.1	0.1
XGBoost	SHAP (Tree)	0.24	0.1
XGBoost	LIME	2.09	0.1
Decision Tree	SHAP (Kernel)	2.5	0.1
Decision Tree	SHAP (Tree)	0.13	0.1
Decision Tree	LIME	3.45	0.1

Table 6 shows stacked bar chart comparing total time (training + testing + XAI) for Decision Tree, XGBoost, and Random Forest with SHAP Tree and LIME. XGBoost + SHAP Tree provides the best balance for production deployment.

XGBoost + SHAP Tree delivers the best overall trade-off: 99.53% accuracy, 2.49 s training, 0.024 s testing, and only 0.24 s for 500 explanations. Decision Tree + SHAP Tree is ideal for edge deployment: total pipeline time < 1 second. LIME is 2–3 orders of magnitude slower than SHAP Tree on a per-sample basis [8], [22]. Random Forest's explanation overhead (SHAP Tree: 19.35 s) is much higher than XGBoost's due to a larger number of deeper trees.

Table 7: Real-time IDS simulation (XGBoost + SHAP) with per-packet performance metrics.

Packet #	Prediction	Confidence	True Label	Prediction Time	XAI Time	Top 3 Features
1	BENIGN	0.01%	BENIGN	5.96 ms	24.19 ms	Flow_Duration
2	BENIGN	0.05%	BENIGN	0.74 ms	23.75 ms	Flow_Duration
3	ATTACK	99.67%	ATTACK	0.70 ms	23.04 ms	Flow_Duration

Table 7 shows the real-time intrusion detection simulation results using the XGBoost classifier with SHAP explanations. The table reports packet-level predictions, confidence scores, true labels, prediction latency, XAI explanation time, and the top contributing features for three representative network flows. For each incoming packet, the system outputs a prediction (BENIGN or ATTACK), a confidence score (probability of the predicted class), the true ground-truth label, the prediction time (inference latency), the SHAP explanation time, and the three most influential features for that prediction. As shown in the table, benign packets (Packets #1 and #2) are classified with very low confidence (0.01% and 0.05%, respectively), indicating that the model is not over-confident on normal traffic. In contrast, the attack packet (Packet #3) is correctly identified with a high confidence of 99.67%, demonstrating the model's strong discriminative capability. The prediction time per packet is consistently below 6 ms, with the majority below 1 ms, confirming the feasibility of sub-millisecond inference for real-time deployment. The SHAP explanation time remains stable around 23–24 ms per packet, which, while higher than inference time, is still acceptable for near-real-time security operations where interpretability is required. Notably, Flow_Duration emerges as the top contributing feature across all three packets, underscoring its importance in the decision process of the XGBoost model on the CICIDS2017 dataset.

5.4 Comparative Analysis with Existing State-of-the-Art Methods

Table 8: Comparison of the proposed framework with existing cicids2017-based intrusion detection methods

Ref.	Year	Dataset Used	Model / Technique	XAI Support	Accuracy (%)	Key Findings	Limitations
Akbarova [2]	2023	CICIDS2017	Supervised and unsupervised ML using PyCaret	No	99.6	Random Forest achieved the best detection performance	No explainability analysis
AlHabshy et al. [3]	2022	CICIDS2017	Multi-attack anomaly detection	Partial	99.9	High accuracy with efficient attack detection	Computational complexity remains high
Ashraf et al. [4]	2023	CICIDS2017	Feature-selected XGBoost	No	99.91	Excellent classification performance on CICIDS2017	Limited model interpretability

Chan [6]	2025	CICIDS2017	Comparative ML models	No	99.2	Boosting algorithms achieved superior performance	Lack of explainability support
Ding et al. [8]	2025	CICIDS2017	Explainable intrusion detection framework	SHAP, LIME	98.7	SHAP showed stronger feature attribution consistency	Explanation overhead increased runtime
Goyal et al. [9]	2025	CICIDS2017	XAI-enabled IDS	SHAP, LIME	98.9	Improved IDS transparency and analyst trust	Limited scalability analysis
Hasan and Islam [15]	2024	CICIDS2017	Multi-class machine learning IDS	No	99.3	Effective multi-class attack detection	No interpretable analysis
Khan et al. [16]	2024	CICIDS2017	Explainable machine learning IDS	SHAP, LIME	98.5	Transparent IDS framework for cybersecurity	Federated learning not considered
Osa et al. [24]	2025	CICIDS2017	Shallow and deep learning classifiers	No	98.4	Tree-based models performed effectively	Lack of XAI integration
Xu and Liu [26]	2025	CICIDS2017	Robust anomaly detection using ML	No	99.1	Robust detection across attack categories	Explainability not addressed
Proposed Work	**2026**	**CICIDS2017**	**Hybrid Federated Learning-based IDS**	**SHAP, LIME**	**99.5***	**Unified benchmarking of ML, XAI, and federated learning for scalable IDS**	**Real-time deployment remains future work**

*** Experimental result obtained using the proposed framework.**

A comparative analysis of the recent intrusion detection approaches is presented in Table 8 in order to evaluate them in the CICIDS2017 dataset. The comparison encompasses the machine learning methods used, explainable artificial intelligence (XAI) support, reported detection accuracy, key findings and limitations of each study. The results of the existing works show that ensemble learning and tree-based models like Random Forest and XGBoost are effective in intrusion detection and have been shown to give high classification accuracy. Ashraf et al. [4] reported one of the highest accuracy with feature selection XGBoost, and Akbarova [2] and Hasan and Islam [15] showed good results with conventional machine learning techniques. It is further evident from the table that most previous works were mostly about maximizing detection performance with no interpretability mechanism. Various studies recently have used explainable AI techniques like SHAP and LIME to enhance the transparency and confidence of the analyst in IDS predictions, such as Ding et al. [8] and Goyal et al. [9] and Khan et al. [16]. But these methods tended to add to the computer overhead or didn't provide a discussion on scalability for distributed systems. Moreover, the proposed approach is based on federated learning and explainable AI techniques in a single benchmarking architecture, with the CICIDS2017 dataset, unlike the previous ones. It does not only allow the implementation of a classification performance competitive to the best ones, but also increases the interpretability and scalability of distributed IDS systems. Hence, through the comparative analysis, it can be found that the proposed work addresses the important limitations observed in the existing CICIDS2017 based intrusion detection studies in terms of explainability, benchmarking consistency and decentralized learning capability.

6. Conclusion

This study offers a first consolidated evaluation of 12 AI classifiers and 3 XAI methods on CICIDS2017 dataset [27] including detailed analysis of execution time. XGBoost + SHAP Tree offers the best accuracy–explanation–time trade-off for production IDS, achieving 2083 samples per second. Decision Tree + SHAP Tree is optimal for real-time edge deployment, with sub-millisecond inference and explanation times. LIME is model-agnostic but

significantly slower; it should be reserved for debugging or compliance contexts where stability is paramount [8], [22]. However, KernelSHAP is not suitable for high dimensional and large scale data, as it has an exponential runtime. Our results corroborate recent state-of-the-art [2]–[4], [6]–[9], [15], [24] and quantify the huge computational disadvantage of SHAP Tree over the other XAI techniques. This benchmark addresses calls for reproducible, time-aware evaluations in the IDS community [1], [5], [10], [12], [13], [23], [28].
Multiclass CICIDS2017: Compute AUPR and per-class F1 scores for rare attacks (e.g., Heartbleed, Infiltration) of all 12 classifiers and XAI methods.

Real-Time XAI Optimization: Create hybrid SHAP-LIME approaches that adjust to the model type (tree vs. neural) in order to balance speed and model-agnosticism [27]. Automated XAI for SOCs: Combine SHAP explanations with Large Language Models (LLMs) for automated, natural-language security alerts. XAI Study on Deep Learning Models: Extending the benchmark to CNNs, LSTMs and Autoencoders [17]. When deploying in real time, choose Decision Tree + SHAP Tree (<1s pipeline). For cloud-based IDS with high accuracy, use XGBoost + SHAP Tree (2083 samples/s). Do not use KernelSHAP and LIME for online inference of large datasets.

Acknowledgement: The authors would like to thank their institution for the financial support of this article.

Conflict of Interest: The authors declare that there is no conflict of interest regarding the publication of this paper. The research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. R. Ahmad, M. A. Khan, and S. Ali, "A comprehensive survey on intrusion detection systems," Discover Artificial Intelligence, vol. 5, 2025.
2. F. T. Akbarova, "Intrusion detection with supervised and unsupervised learning using PyCaret over CICIDS 2017 dataset," Journal of Cybersecurity, 2023.
3. A. A. AlHabshy, M. A. Alzain, and B. Alzahrani, "Ameliorated multi-attack network anomaly detection," IEEE Access, vol. 10, pp. , 2022.
4. S. Ashraf, M. A. Khan, and T. Ali, "Comparative study of machine learning techniques for intrusion detection on CICIDS-2017," International Journal of Computing and Digital Systems, vol. [to be filled], no. [to be filled], 2023.
5. S. Ashraf, T. Ahmed, and S. R. Hussain, "A systematic literature study of machine learning techniques based intrusion detection: datasets, models, challenges, and future directions," Journal of Big Data, vol. 12, no. 1, 2025.
6. J. Chan, "Performance comparison of ML models on CICIDS2017," Nature Scientific Reports, Table 8, 2025.
7. K. Çoşkun and G. Çetin, "A comparative evaluation of the boosting algorithms for network attack classification," International Journal of 3D Printing Technologies and Digital Industry, vol. 6, no. 1, pp. 102–112, 2022. [DOI: 10.46519/ij3dptdi.1030539]
8. L. Ding, X. Wang, and Y. Zhang, "Explainable AI for forensic analysis: A comparative study of SHAP and LIME in intrusion detection models," Applied Sciences, vol. 15, no. 13, 2025.
9. D. Goyal, P. Sharma, and R. Mehta, "An explainable AI framework for intrusion detection using SHAP and LIME," in 2025 IEEE 14th Global Conference on Consumer Electronics (GCCE), Osaka, Japan, 2025.
10. R. Hameed, S. Iqbal, and M. Zafar, "A comprehensive review of intrusion detection systems across dataset families," Kashf Journal of Multidisciplinary Research, vol. 3, no. 1, 2026.
11. Hindawi, "Feature selection based on cross-correlation for the intrusion detection system," Security and Communication Networks, vol. 2020, pp. 1–17, 2020. [DOI: 10.1155/2020/8868006]
12. IEEE Xplore, "Classifier selection for an ensemble of network traffic analysis machine learning models," in 2023 IEEE International Conference on Dependable, Autonomic and Secure Computing (DASC), 2023.
13. K. S. Alshudukhi, S. Ali, M. Humayun, and O. Alruwaili, "Next-generation lightweight explainable AI for cybersecurity: A review on transparency and real-time threat mitigation," [journal name missing], 2025.
14. IJAM Journal, "Explainable artificial intelligence applications in cybersecurity: Enhancing transparency in intrusion detection systems," International Journal of Advanced Manufacturing, vol. [to be filled], 2025.
15. M. R. Hasan and A. Islam, "Multi-class intrusion detection using CICIDS2017," IEEE Access, vol. 12, 2024. [DOI missing]
16. M. F. Khan, S. A. Hussain, and H. Ahmad, "Explainable AI and machine learning models for transparent and scalable intrusion detection systems," Journal of Information Systems Engineering and Management, vol. 9, no. 4, 2024.

17. M. Lanvin, P. M. Rodriguez, and A. E. Zadeh, "Towards understanding alerts raised by unsupervised network intrusion detection systems," Inria, Research Report, 2024. [Available: https://gitlab.inria.fr/maxime.lanvin/ae-pvalues]
18. S. M. Lundberg and S. I. Lee, "A unified approach to interpreting model predictions," in Advances in Neural Information Processing Systems (NeurIPS), Long Beach, CA, USA, Dec. 2017, pp. 4765–4774. [arXiv:1705.07874]
19. S. M. Lundberg and S. I. Lee, "Consistent individualized feature attribution for tree ensembles," arXiv preprint arXiv:1802.03888, 2018.
20. S. M. Lundberg and S. I. Lee, "SHAP: Mathematical formulation and additive feature attribution," 2017. [Available: https://github.com/shap/shap]
21. S. M. Lundberg, G. Erion, and S. I. Lee, "From local explanations to global understanding with explainable AI for trees," Nature Machine Intelligence, vol. 2, pp. 56–67, 2020. [DOI: 10.1038/s42256-019-0138-9]
22. C. Molnar, Interpretable Machine Learning: A Guide for Making Black Box Models Explainable, 2nd ed., Christoph Molnar, 2025. [ISBN: 978-3-911578-03-5]
23. I. C. Obagbuwa, M. N. Ngafeeson, O. F. Obagbuwa, and A. Tsetse, "Machine learning and explainable artificial intelligence for network intrusion detection," International Journal of Information Security and Privacy, vol. 20, no. 1, pp. 1–22, 2026. [DOI: 10.4018/IJISP.371997]
24. E. Osa, E. J. Edifon, and S. Igori, "Performance analysis of shallow and deep learning classifiers leveraging the CICIDS 2017 dataset," International Journal of Intelligent Systems and Applications, vol. 17, no. 2, pp. 42–55, 2025. [DOI: 10.5815/ijisa.2025.02.04]
25. I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in Proceedings of the 4th International Conference on Information Systems Security and Privacy (ICISSP), Funchal, Madeira, Portugal, Jan. 2018, vol. 1, pp. 108–116. [DOI: 10.5220/0006639801080116]
26. Z. Xu and Y. Liu, "Robust anomaly detection in network traffic: Evaluating machine learning models on CICIDS2017," arXiv preprint arXiv:2506.19877, 2025.
27. F. Uccello, M. Pawlicki, S. D'Antonio, R. Kozik, and M. Choraś, "A new cybersecurity approach enhanced by xAI-derived rules to improve network intrusion detection and SIEM," ScienceDirect, 2025. *Jordan Journal of Electrical Engineering*, vol. 9, no. 3, pp. 272-284, 2023, doi: 10.5455/jjee.204-1677985772
28. TechRxiv, "A comprehensive survey of AI-driven network intrusion detection systems," TechRxiv, Dec. 2025. [DOI: 10.36227/techrxiv.176620705.58682779/v1]
29. V. P. Patil and S. N. Ohatakar, "A review study on security enhancements and privacy considerations in Industrial Internet of Things (IIoT) applications," International Journal of Computer Information Systems and Industrial Management Applications, vol. 18, no. 17s, pp. 1293–1321, 2026, doi: 10.70917/ijcisim-2026-4828.
30. V. P. Patil and S. N. Ohatkar, "Blockchain for IoT/Industrial IoT applications," in Proc. IEEE Int. Conf. Blockchain and Distributed Systems Security (ICBDS), 2024, doi: 10.1109/ICBDS61829.2024.10837075.
31. V. P. Patil and S. N. Ohatkar, "Blockchain based secure threat detection model for Industrial IoT applications," in Proc. IEEE Int. Conf. Blockchain and Distributed Systems Security (ICBDS), 2025, doi: 10.1109/ICBDS67396.2025.11376897.
32. V. P. Patil and S. N. Ohatkar, "Threat-aware Industry 5.0 attack graph framework for CICIDS2017-based cyber-physical risk assessment," Genetics and Molecular Research, vol. 25, no. 7s, 2026, doi: 10.4238/tm9g1s42.