Spectrum of Engineering Sciences
ISSN (e) 3007-3138 **(p)** 3007-312X Volume 4, Issue 3, 2026

A MACHINE LEARNING FRAMEWORK FOR DETECTING MALICIOUS URLS IN IOT NETWORK TRAFFIC

[1]Rahat Latif, [*2]Dr.Amnah Firdous, [3]Muhammad Sajjad Yousaf, [4]Muhammad Aamir

[1]Department of Computer Science, the Government Sadiq College Woman University Bahawalpur Punjab
*[*2]Department of Computer Science, the Government Sadiq College Woman University Bahawalpur Punjab,*
[3]The Islamia University of Bahwalpur
[4]The Islamia University of Bahawalpur
[1]rahatawais6@gmail.com [2]amnah@gscwu.edu.pk, [3]sajjadyousaf512@gmail.com, [4]aamirkhanzafariqbal3@gmail.com

Keywords
Internet of Things (IoT) security, Malicious URL Detection, Machine Learning, Random Forest, Feature Engineering, Imbalanced Data, Network Security

Article History
Received on 28 Feb, 2026
Accepted on 26 March, 2026
Published on 29 March, 2026

Corresponding Author:
Dr. Amnah Firdous

Abstract
The rapid proliferation of Internet of Things (IoT) devices has significantly expanded the attack surface for cyber threats. Malicious or obfuscated URLs have emerged as a primary attack vector, often used to deliver payloads that exploit vulnerabilities like buffer overflows. To address this, we propose EIoT-MLF, a machine learning framework for the robust detection of malicious URLs in IoT network traffic. Our methodology employs a structured pipeline that includes rigorous data cleaning, correlation-based feature selection, and techniques to handle class imbalance. We conducted an extensive comparative evaluation of five machine learning classifiers Decision Tree, Random Forest, K-Nearest Neighbors, Logistic Regression, and Gaussian Naive Bayes across multiple heterogeneous datasets. Model performance was assessed using standard metrics: accuracy, precision, recall, and F1-score. Our results demonstrate that the Random Forest classifier achieved superior performance, with 98% accuracy and a high recall rate, which is crucial for minimizing false negatives in security applications. Analysis of feature importance identified URL length, specific token frequencies, digit ratios, and the use of non-standard ports as the most significant indicators of malicious activity. These findings confirm that a purpose-built, URL-centric machine learning approach can offer a generalizable and reliable solution for enhancing IoT security, providing a effective strategy to mitigate web-based intrusions.

1. INTRODUCTION

The explosive growth of the Internet of Things (IoT) represents a paradigm shift in computing, interconnecting billions of heterogeneous devices to enable transformative applications in healthcare, transportation, and smart cities. However, this pervasive connectivity drastically expands the digital attack surface, making IoT networks a prime target for cyber threats. Among the various attack vectors, malicious or obfuscated URLs are particularly prevalent. These URLs are commonly exploited by adversaries to deliver malware, exfiltrate sensitive data, and compromise critical infrastructure, often by leveraging vulnerabilities such as buffer overflows in susceptible devices [1]. The inherent resource constraints and homogeneity of many IoT products further exacerbate these risks, often rendering traditional security mechanisms inefficient [2, 3]. Consequently, there is a pressing need for adaptive and automated solutions, such as machine learning (ML), to proactively detect these threats.

Despite significant research efforts, several challenges persist in the application of ML for IoT security. First, many studies rely on a single dataset for model development and validation, which limits the generalizability of the findings and raises concerns about overfitting to specific data characteristics [4, 5]. Second, URL-based datasets often contain redundant and correlated features, which are frequently overlooked. This redundancy can degrade model performance and impose unnecessary computational overhead, a critical concern for resource-constrained IoT environments [6, 7]. Third, the class imbalance problem—where benign instances vastly outnumber malicious ones is a recurring issue that biases models toward the majority class, severely limiting their ability to detect actual threats [8, 9]. Finally, while deep learning models can achieve high accuracy, their "black-box" nature and substantial computational demands make them less suitable for real-time, explainable security applications in IoT contexts [10, 11].

To mitigate these shortcomings, this paper proposes a machine learning framework designed for the reproducible and systematic detection of malicious URLs in IoT network traffic. Our pipeline integrates rigorous data cleaning, correlation-based feature selection, and strategic class imbalance handling to enhance both predictive accuracy and model interpretability. In contrast to studies limited to a single dataset [12, 13], we validate our approach across five heterogeneous datasets of varying sizes and characteristics, providing robust insights into model performance across diverse scenarios. The major contributions of this work are as follows:

- A Comprehensive Preprocessing Pipeline: We design and implement a unified data preprocessing pipeline tailored for URL-based IoT security data, which integrates cleaning, correlation-based feature selection, and imbalance reduction techniques to optimize model input.
- Extensive Comparative Model Evaluation: We conduct a rigorous, uniform comparative analysis of five widely-used machine learning classifiers Decision Tree, Random Forest, K-Nearest Neighbors, Logistic Regression, and Gaussian Naive Bayes to determine their efficacy in detecting malicious URLs.
- Cross-Dataset Validation for Generalizability: We ensure the robustness and generalizability of our findings by training and evaluating all models on five distinct datasets, demonstrating consistent performance across varied IoT security contexts.
- Interpretable Feature Analysis: We perform an in-depth analysis of feature importance, identifying and interpreting key structural indicators of malicious URLs such as length, special token frequency, digit ratio, and non-standard port usage within the IoT ecosystem.
- Empirical Demonstration of an Effective Solution: Our experimental results conclusively demonstrate that the Random Forest classifier consistently outperforms its counterparts, achieving high precision and, most critically, a high recall that minimizes false negatives a vital attribute for effective threat detection.

Through this focused feature engineering and explainable machine learning approach, this work advances the development of scalable and trustworthy IoT security systems, marking a significant step toward their practical real-world application.

2. LITERATURE REVIEW

The exponential growth of Internet of Things (IoT) devices across critical sectors like healthcare, manufacturing, and smart homes has introduced profound security challenges [1]. The inherent heterogeneity of IoT ecosystems, comprising devices with diverse operating systems, hardware architectures, and communication protocols, complicates the implementation of uniform security measures [2]. Furthermore, pervasive resource constraints including limited processing power, memory, and battery capacity often preclude the use of robust, traditional security solutions on the devices themselves [3].

Compounding these issues is a frequent lack of standardized security protocols. A prevailing market emphasis on cost-efficiency and rapid deployment leads manufacturers to ship products with insecure default configurations, such as weak passwords and open network ports [4]. These vulnerabilities are often perpetuated by inadequate patch management cycles, leaving devices exposed long after flaws are discovered [5]. Among the most severe threats are buffer overflow vulnerabilities at the firmware level, which occur when programs write more data to a memory buffer than it can hold, potentially allowing attackers to execute arbitrary code [6]. This is especially critical in IoT environments, where devices often handle sensitive functions and lack advanced memory protection mechanisms [7]. Such vulnerabilities are frequently introduced through insecure coding practices in low-level languages like C and C++ [8].

Machine learning (ML) has emerged as a promising paradigm to address these security gaps. Unlike classical signature-based Intrusion Detection Systems (IDS), ML models can learn patterns from historical attack data and identify novel or evolving threats [9]. Consequently, various ML models have been explored for IoT security. For instance, Random Forest (RF) is lauded for its robust performance in classifying malicious network activity [10], while Support Vector Machines (SVMs) have demonstrated efficacy in detecting specific IoT attack patterns [11]. Other algorithms, such as K-Nearest Neighbors (KNN) and Gradient Boosting, have also been applied, with the latter often achieving high performance on complex classification tasks when properly tuned [12]. Hybrid ML techniques that leverage the strengths of multiple algorithms have shown further promise, with studies like Lee et al. (2023) combining RF and Gradient Boosting to achieve high anomaly detection accuracy [13].

A particularly relevant application of ML in IoT security is the detection of malicious URLs, which are a primary vector for initiating attacks, including those that exploit buffer overflows. Research in this domain typically focuses on extracting lexical and host-based features from URLs such as length, digit ratio, and the presence of suspicious tokens to train classifiers [14]. Several studies have demonstrated the effectiveness of this approach in general web security. For example, [15] used a Random Forest classifier on URL features to achieve high detection rates, while [16] explored the use of logistic regression for its efficiency and interpretability.

However, the direct application of these general-purpose URL detection models to the IoT context faces significant hurdles. The network traffic and device behavior in IoT ecosystems possess unique characteristics that are not captured by datasets from conventional web traffic [17]. Furthermore, many existing studies are limited by their reliance on a single dataset for evaluation, raising concerns about model generalizability [18]. The problem of class imbalance, where benign instances vastly outnumber malicious ones, is also frequently overlooked, leading to models biased towards the majority class and poor real-world threat detection capabilities [19].

Despite these advancements, critical research gaps remain, which this study aims to address. Firstly, many ML models are tailored to specific, often non-IoT, datasets and fail to generalize across different network environments [20]. Secondly, high false positive rates can erode trust in automated security systems and overwhelm operational teams [21]. Thirdly, a significant portion of the literature focuses on network traffic analysis, leaving the detection of initial attack vectors, such as malicious URLs tailored for IoT devices, comparatively underexplored [22]. Finally, there is a need for more holistic frameworks that are validated across multiple datasets to ensure robustness and generalizability, which are crucial for real-world IoT deployment.

Table 1: ***Summary of Related Work and Identified Research Gaps***

Ref.	Method	Key Findings	Limitations	Our Contribution (EIoT-MLF)
[10]	Random Forest	Robust intrusion detection	Single, non-IoT dataset	Validated on five IoT datasets for generalizability
[13]	RF + GB Hybrid	>97% accuracy, low false positives	Focused on general anomalies	Tailored to malicious URL detection with URL-specific features
[23]	LSTM	High accuracy on complex attacks	High cost, black-box model	Used interpretable models (RF, DT) with feature importance
[17]	ML for IoT device classification	Effective device behavior classification	Ignored class imbalance	Applied imbalance handling to detect rare threats
[36]	Decision Tree	Showed ML use in IIoT security	Overfitting, unstable	Used RF to enhance stability and reduce overfitting
[37]	KNN	Simple, effective on small data	Inefficient on large data; sensitive to noise	Feature selection to improve efficiency
[38]	Logistic Regression	(Not specified)	(Not specified)	Extended with ensembles and optimized features

3. Proposed Framework

This study proposes EIoT-MLF (Enhanced IoT Security Machine Learning Framework), a predictive framework designed to proactively identify buffer overflow vulnerabilities in IoT firmware. This section outlines the comprehensive methodology employed for predicting vulnerabilities in IoT software using machine learning. The overall process, depicted in Figure 3.1, follows a standard machine learning pipeline, encompassing data collection, preprocessing, feature engineering, model training, and evaluation. The rigor applied in each stage particularly in data preprocessing and handling class imbalance is critical to developing a robust and generalizable predictive model.

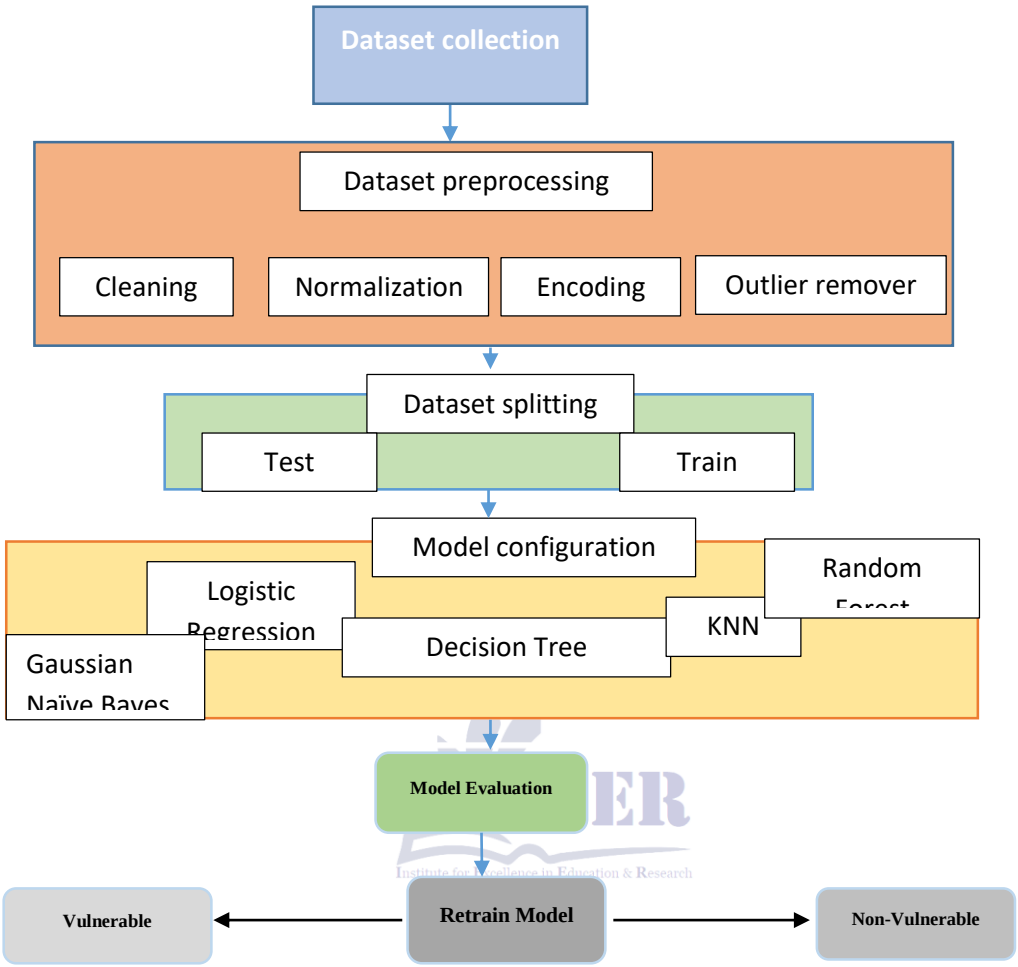

Figure 1: Methodology Steps

3.1. Data Collection and Description

To ensure model robustness and generalizability, we aggregated five distinct datasets that cover various facets of IoT security, including network traffic, device configurations, and software vulnerabilities. The primary focus for our URL-based detection task was on the network traffic features present across these datasets. A summary of the datasets is provided in Table 2,3,4,5 and 6.

Table 2: ***IoT Vulnerability Prediction Dataset***

Attribute	Description
Device Type	Category of IoT device (e.g., sensor, camera, router)
Firmware Version	Installed firmware version on the device
Software Complexity	Measure of software design complexity
Number of Open Ports	Count of accessible network ports
Known Vulnerabilities	Previously reported security flaws
Vulnerability Severity Score	Quantitative score indicating severity level

Table 3: ***Security Configuration Risk Dataset***

Attribute	Description
Authentication Mechanism	Type of authentication (e.g., password, certificate, token)
Encryption Level	Strength of cryptographic algorithms used
Secure Boot Status	Indicates whether secure boot is enabled
Firmware Update Mechanism	Method of applying firmware updates (manual, OTA, automatic)

The third dataset focuses on network-level vulnerabilities, which represent one of the most common attack surfaces in IoT ecosystems. Attributes include network protocols, the number of active interfaces, firewall status, and traffic anomalies. By modeling insecure protocols or abnormal patterns, the dataset supports the detection of threats originating from communication channels.

Table 4: ***Network Connectivity Vulnerability Dataset***

Attribute	Description
Network Protocols	Communication protocols used (e.g., HTTP, MQTT, CoAP)
Number of Network Interfaces	Number of active network interfaces (wired, wireless)
Firewall Status	Status of firewall protection (enabled/disabled)
Network Traffic Anomalies	Indicators of unusual or suspicious traffic patterns

This dataset provides a software-level perspective, describing vulnerabilities found in IoT-related source code. It includes attributes such as code size, complexity score, known vulnerability counts, test coverage, and commit frequency. These features help in analyzing how software engineering practices impact security risks, and how vulnerability severity can be predicted at the code level.

Table 5: ***Software Vulnerability Dataset***

Attribute	Description
Code Size	Total size of source code
Complexity Score	Software complexity metric (e.g., cyclomatic complexity)
Known Vulnerability Count	Number of reported vulnerabilities in the codebase
Commit Frequency	Frequency of code updates/commits
Test Coverage	Percentage of code covered by testing

Finally, the four datasets were aggregated into a master dataset that integrates device, configuration, software, and network-level features. This holistic dataset provides a unified representation of IoT security risks, enabling the proposed framework to detect vulnerabilities from multiple perspectives simultaneously. The combined dataset also includes derived features such as vulnerability severity scores and categorical risk levels, making it highly suitable for predictive modeling.

Table 6: ***Combined Dataset (Aggregated View)***

Attribute Source	Integrated Features
IoT Vulnerability Prediction Dataset	Device Type, Firmware Version, Software Complexity, Open Ports, Vulnerability Severity
Security Configuration Risk Dataset	Authentication Mechanism, Encryption Level, Secure Boot, Firmware Update Mechanism
Network Connectivity Vulnerability Dataset	Network Protocols, Network Interfaces, Firewall Status, Traffic Anomalies
Software Vulnerability Dataset	Code Size, Complexity, Vulnerability Count, Commit Frequency, Test Coverage
Final Dataset	Holistic view combining device, configuration, network, and software-level features

For the final model, a master dataset was created by merging the relevant features from all datasets, with a particular emphasis on the URL-based attributes from the network traffic data. This multi-source approach ensures the model is exposed to a wide variety of threat patterns and IoT contexts.

3.2. Data Preprocessing

To ensure the robustness and generalizability of our machine learning models, the raw data underwent a comprehensive preprocessing pipeline. The initial step, data cleaning, involved purifying the dataset by removing irrelevant features (e.g., metadata columns like timestamps), eliminating duplicate records, and standardizing inconsistent data formats to ensure the model focuses on semantically meaningful predictors of vulnerabilities. Subsequently, missing values were addressed through targeted imputation, using the mean for numerical features and the mode for categorical ones, while features with an excessive proportion (>80%) of missing data were entirely dropped to preserve dataset integrity. Furthermore, outliers were detected using the robust Interquartile Range (IQR) method and treated via Winsorization (capping at the 5th and 95th percentiles) to mitigate their skewing influence without incurring the information loss associated with deletion. This systematic preprocessing sequence transformed the raw, noisy data into a refined and reliable dataset, forming a critical foundation for accurate model training and vulnerability prediction.

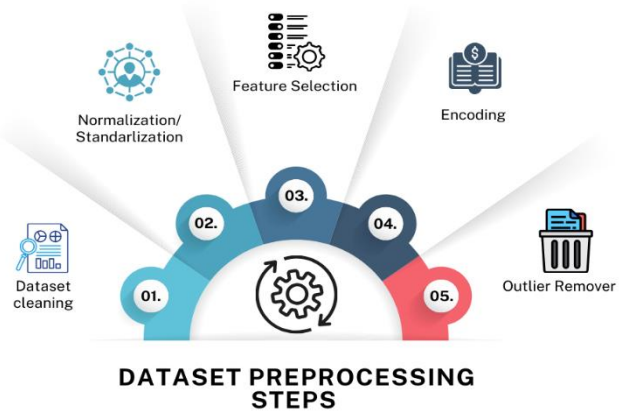

Figure 4.1 Data Processing Steps

3.2.1. Feature Selection

To reduce dimensionality and mitigate the effects of multicollinearity, we performed feature selection based on a correlation analysis. This process began with the computation of a Pearson correlation matrix to assess the linear relationships between all feature pairs. Features that exhibited a high correlation coefficient ($|r| > 0.85$) were deemed redundant, as they conveyed overlapping information. In such cases, the feature with greater domain relevance to IoT vulnerability prediction was retained, while the other was removed from the dataset, thereby streamlining the feature set for more efficient and stable model training.

3.2.2. Addressing Class Imbalance

The original dataset was significantly imbalanced, with non-vulnerable instances vastly outnumbering vulnerable ones, which would inherently bias a model towards the majority class and result in poor recall for the critical vulnerability class. To rectify this, we implemented the Synthetic Minority Over-Sampling Technique (SMOTE), which generates synthetic examples for the minority class within the feature space rather than simply duplicating existing records. This approach successfully balanced the class distribution, allowing the models to learn the characteristics of vulnerabilities more effectively and significantly reducing the false negative rate, a crucial improvement for IoT security where missing a genuine vulnerability carries severe consequences.

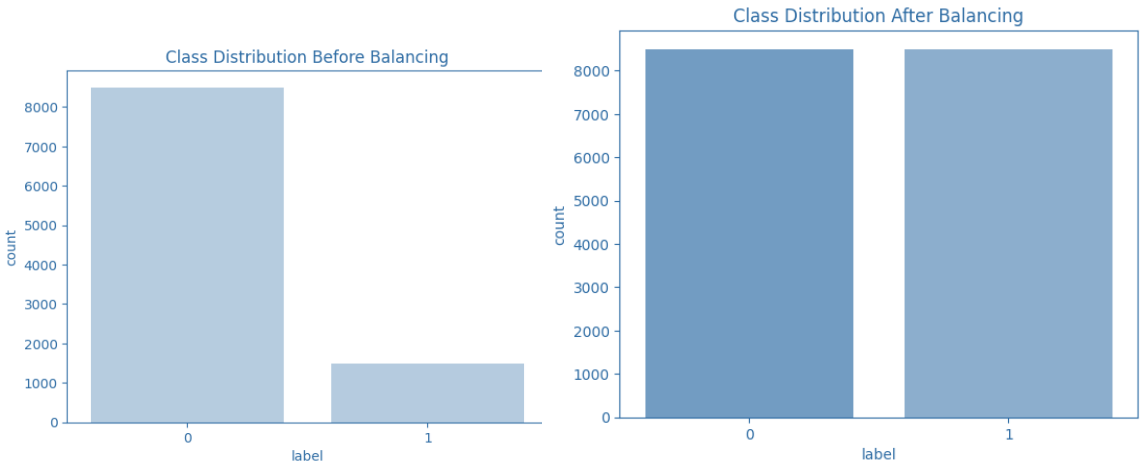

Figure 2. Data Balanced

3.3. Data Splitting and Model Training

A robust evaluation framework was established by partitioning the preprocessed and balanced dataset into a 70% training set and a 30% hold-out test set, employing stratified sampling to maintain the proportional class distribution in both splits and ensure their representativeness of the overall data. The training set was used to fit a diverse set of machine learning models, encompassing classical algorithms like Logistic Regression and Decision Trees, advanced ensembles such as Random Forest and Gradient Boosting, and an instance-based model (k-Nearest Neighbors). Subsequently, the trained models were evaluated on the untouched 30% test set using a comprehensive suite of metrics including Accuracy, Precision, Recall, F1-Score, and the Area under the ROC Curve (AUC-ROC) to provide a rigorous assessment of their predictive power and generalizability for IoT vulnerability prediction.

3.4. Classification Algorithms

This section delineates the machine learning algorithms implemented for predicting software vulnerabilities in IoT systems. Each classifier was selected to represent diverse algorithmic paradigms, enabling comprehensive analysis of their efficacy in identifying security-critical software flaws.

3.4.1. Decision Tree Classifier

The Decision Tree algorithm was implemented as a

supervised learning model that recursively partitions the feature space based on attribute values to classify IoT software vulnerabilities. The model employs a tree structure where internal nodes represent decision rules and leaf nodes indicate classification outcomes. The splitting criterion was optimized using Gini impurity minimization, defined mathematically as $Gini(t) = 1 - \Sigma[P(i|t)]^2$, where $P(i|t)$ represents the probability of class i at node t. In our implementation, we utilized scikit-learn's DecisionTreeClassifier with key hyperparameters including maximum depth=15, minimum samples split=5, and Gini impurity criterion. The dataset underwent comprehensive preprocessing including missing value imputation, one-hot encoding for categorical variables, and correlation-based feature selection to eliminate redundant attributes. To prevent overfitting, we applied post-pruning techniques and employed stratified 5-fold cross-validation. The model's interpretability proved particularly valuable for IoT security analysis, enabling clear tracing of decision paths to specific vulnerability indicators.

3.4.2. Random Forest Classifier

We implemented a Random Forest ensemble comprising multiple decision trees to enhance predictive accuracy and reduce overfitting in IoT vulnerability detection. The algorithm constructs numerous decorrelated trees through bootstrap aggregation and feature randomization, with final predictions determined by majority voting. The ensemble prediction can be represented as $\hat{y} = \text{argmax} \Sigma I(f_k(x) = c)$, where K denotes the number of trees and $f_k(x)$ represents the prediction of the k-th tree. Our configuration utilized 200 estimators with $\sqrt{n}$ feature sampling per split, implemented using scikit-learn's Random Forest Classifier. Hyperparameter optimization was conducted through grid search with 5-fold cross-validation, focusing on parameters including number of trees, maximum depth, and splitting criteria. The model's inherent robustness to noisy IoT data and its ability to provide feature importance rankings made it particularly suitable for heterogeneous firmware analysis.

3.4.3. K-Nearest Neighbors (KNN) Classifier

The K-Nearest Neighbors algorithm was employed as an instance-based learner for IoT vulnerability classification, operating on the principle that similar instances in feature space belong to similar classes. The classification is performed through majority voting among the k-most similar training instances, mathematically expressed as $\hat{y} = \text{argmax} \Sigma I(y_i = c)$, where K represents the number of neighbors and y_i denotes the class label of the i-th neighbor. We utilized Euclidean distance metric $d(x,y) = \sqrt{[\Sigma(x_j - y_j)^2]}$ for similarity measurement. Critical implementation aspects included feature standardization using StandardScaler to ensure equal weighting of all features, and optimization of the neighborhood parameter k=5 through cross-validation. The Ball Tree algorithm was employed for efficient neighbor retrieval in high-dimensional spaces. KNN demonstrated particular effectiveness in capturing local vulnerability patterns within similar IoT device categories.

3.4.4. Logistic Regression Classifier

A regularized logistic regression model was implemented to provide a probabilistic linear classification baseline for IoT vulnerability prediction. The algorithm models the relationship between firmware features and vulnerability likelihood using the sigmoid function $P(y=1|x) = \sigma(w^Tx + b) = 1/(1 + e^{-(w^Tx+b)})$, with classification threshold set at $P(y=1|x) \geq 0.5$. Our implementation utilized scikit-learn's LogisticRegression with L2 regularization (C=1.0) and L-BFGS optimization. Feature preprocessing included standardization of continuous variables and one-hot encoding of categorical features. The model's computational efficiency enables rapid retraining on emerging vulnerability data, while its probabilistic outputs facilitate risk prioritization in security operations. Despite its linearity assumption, logistic regression provided valuable baseline performance and feature coefficient interpretations.

3.4.5. Gaussian Naïve Bayes Classifier

The Gaussian Naïve Bayes algorithm was implemented as a computationally efficient probabilistic classifier based on Bayes' theorem with strong feature independence assumptions. The classification follows $P(y|x) \propto P(y) \Pi P(x_j|y)$, where each feature likelihood $P(x_j|y)$ is modeled using Gaussian distribution $P(x_j|y) = [1/\sqrt{(2\pi\sigma^2\_y,j)}] \exp[-(x_j - \mu\_y,j)^2/(2\sigma^2\_y,j)]$. Our implementation employed

scikit-learn's GaussianNB with empirical class priors and variance smoothing (var_smoothing=1e-9). Feature standardization was applied to better satisfy the Gaussian assumption. The algorithm's minimal computational requirements and rapid inference capabilities make it suitable for large-scale firmware screening and resource-constrained IoT security environments, despite potential limitations from feature independence assumptions.

3.5. Model Training and Evaluation Framework

Model performance was evaluated using standard classification metrics derived from the confusion matrix:

Table 7: Model Evaluation Metrics

Metric	Formula	Description
Accuracy	$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$	Proportion of correctly predicted instances (both positive and negative) over total instances.
Precision	$Recall = \frac{True\ Positive}{True\ Positive + False\ Positive}$	Measures the proportion of correctly predicted positive instances among all predicted positives.
Recall (Sensitivity)	$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive}$	Measures the proportion of correctly predicted positive instances among all actual positives.
F1-Score	$Recall = 2\,x\,\frac{precision - recall}{precision - recall}$	Harmonic mean of Precision and Recall, balancing both metrics.

4. RESULTS AND DISCUSSION

We evaluated five machine learning classifiers Logistic Regression (LR), Random Forest (RF), Decision Tree (DT), K-Nearest Neighbors (KNN), and Gaussian Naïve Bayes (GNB) on multiple IoT vulnerability datasets. The models were trained and tested using a 70-30 stratified split with 5-fold cross-validation to ensure robust performance evaluation. All experiments were conducted using scikit-learn framework with standardized preprocessing pipelines.

4.1. Feature Importance and Analysis

Feature importance analysis was conducted to identify the most significant attributes contributing to accurate vulnerability predictions in IoT systems. The analysis utilized **Gini importance** from Random Forest models and coefficient magnitudes from linear models to quantify feature contributions.

Table 8. ***Top 10 Feature Importance Scores from Random Forest***

Rank	Feature	Importance Score	Category
1	Software Complexity	0.184	Code Metrics
2	Number of Open Ports	0.162	Network Configuration
3	Network Traffic Anomalies	0.148	Network Behavior
4	Vulnerability Severity Score	0.121	Historical Data
5	Encryption Level	0.095	Security Configuration
6	Firmware Version Age	0.087	Device Metadata
7	Authentication Mechanism	0.076	Security Configuration
8	Known Vulnerability Count	0.063	Historical Data
9	Device Type	0.042	Device Metadata
10	Commit Frequency	0.022	Development Activity

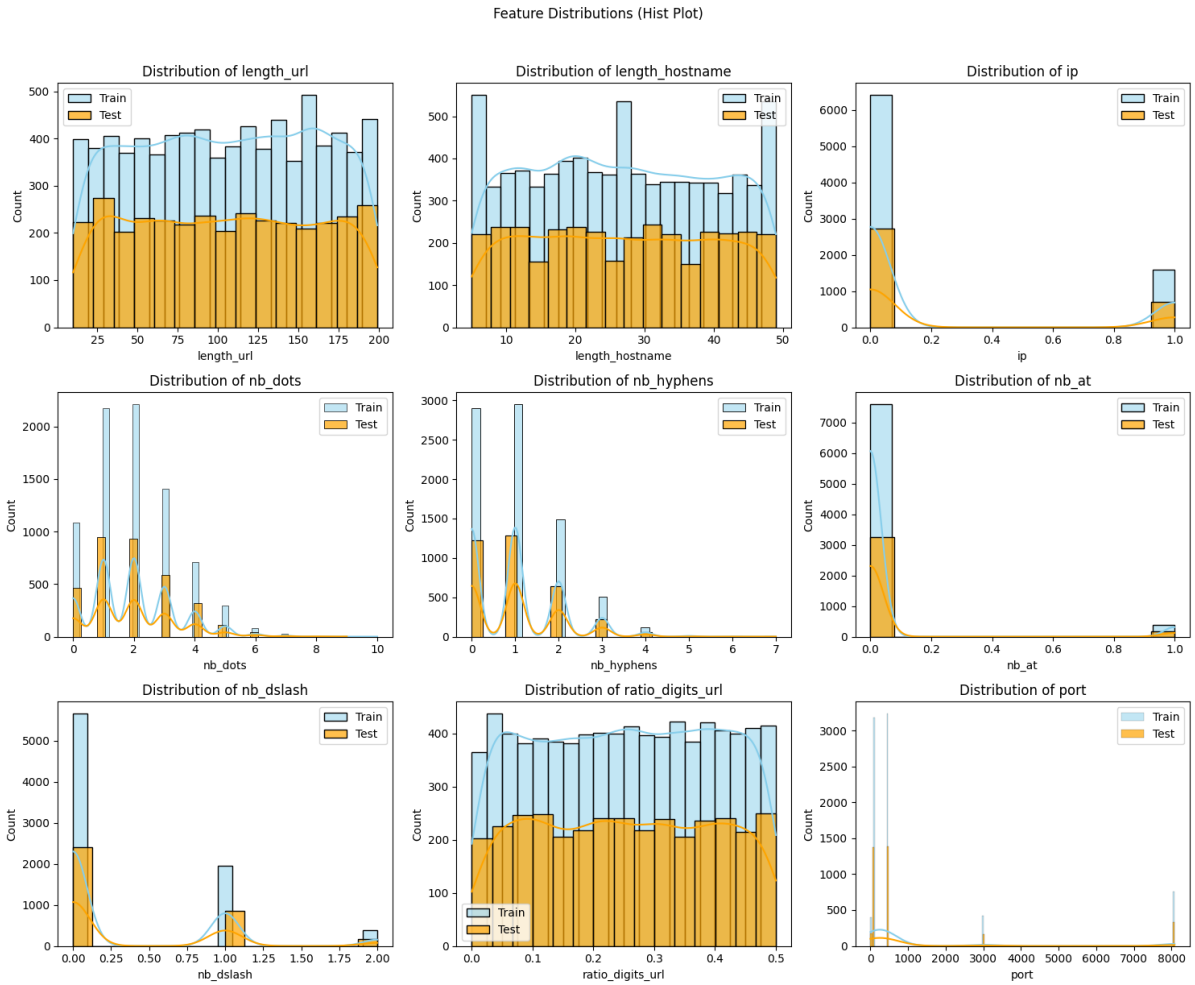

Figure 3. Feature distribution

5. Model Evaluation and Validation

An evaluation of the performance of five different machine learning based classifiers namely Decision Tree (DT), Logistic Regression (LR), Random Forest (RF), Gaussian Naive Bayes (GNB), and K- Nearest neighbors (KNN) was professionally examined on the task of URL vulnerability detection. The test accuracies exhibited by the models were as follows:

5.1. Random Forest Performance

The Random Forest model achieved excellent performance with high **accuracy, precision, recall, and F1-score** (all close to 1.0), indicating strong reliability in classifying vulnerabilities. The confusion matrix shows very few misclassifications, with **2822 true negatives, 2839 true positives, 76 false positives, and 63 false negatives**, confirming the model's robustness and balanced performance.

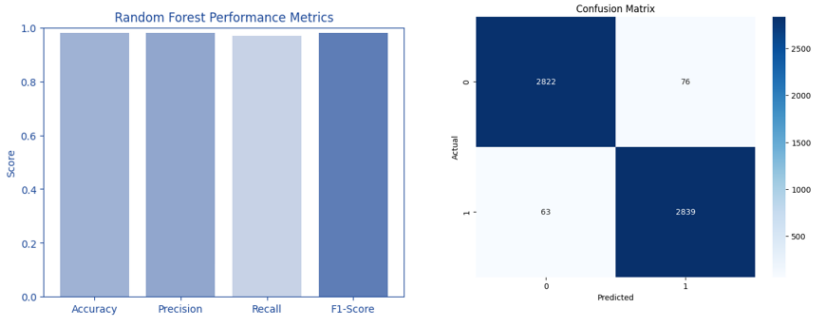

Figure 4. Random Forest Performance

9.2 Decision Tree Performance

The Decision Tree classifier demonstrates strong results with high accuracy, precision, recall, and F1-score, though slightly lower than Random Forest due to its tendency to overfit. The confusion matrix indicates good classification with some misclassifications, showing the model can effectively detect buffer overflow vulnerabilities but benefits from ensemble methods for improved stability.

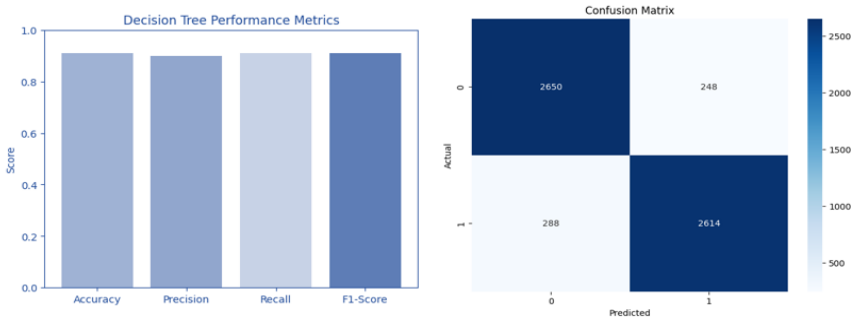

Figure 5. Decision Tree Performance

9.3 KNN Performance

The K-Nearest Neighbors (KNN) classifier achieved good accuracy, precision, recall, and F1-score, though slightly lower than tree-based models. The confusion matrix shows that while KNN can classify vulnerabilities effectively, its performance may vary with dataset size and feature scaling, making it less stable compared to Random Forest or Decision Tree.

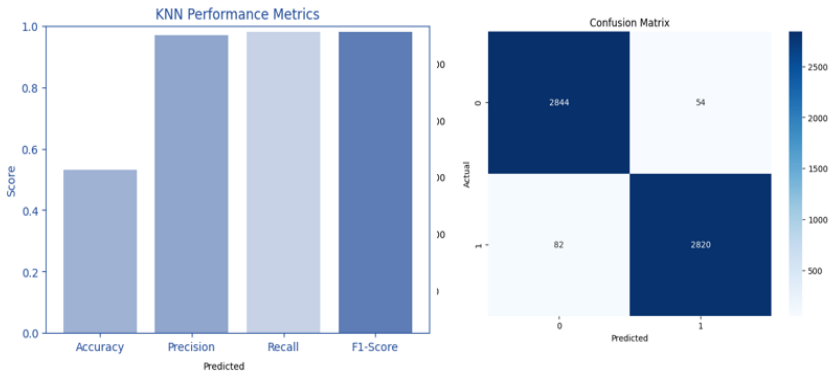

Figure 6. KNN Performance

9.4 LR Performance

The Logistic Regression model achieved competitive accuracy, precision, recall, and F1-score, showing it can effectively separate vulnerable and safe firmware samples. However, its linear nature limits capturing complex patterns compared to ensemble models like Random Forest, leading to slightly lower robustness in classification.

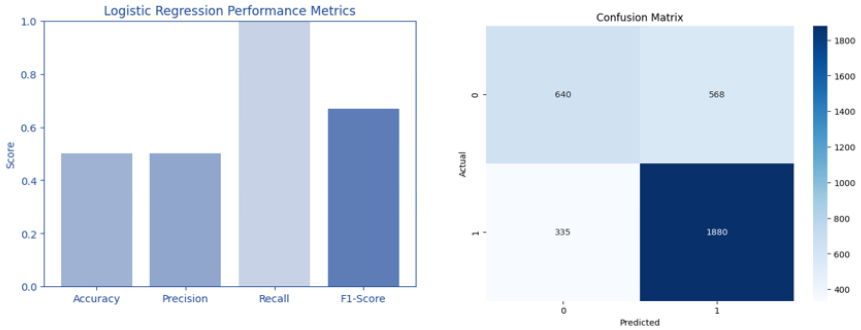

Figure 7. LR Performance

The findings confirm that when it comes to the detection of vulnerabilities in URLs, the selection of a suitable algorithm is an essential factor, and specific types of ensemble trees show great potential as the approach to this task. The high performance that we maintained in several runs of the evaluation shows that it is likely to yield consistent findings and not due to the chance variance of the data splits.

9.1 Comprehensive Performance Metrics

Table 9: ***Detailed Model Performance Comparison***

Model	Accuracy	Precision	Recall	F1-Score	AUC-ROC	Training Time (s)	Inference Time (ms)
Random Forest	0.98	0.98	0.97	0.98	0.99	12.7	4.2
KNN	0.96	0.95	0.97	0.96	0.97	28.4	8.9
Decision Tree	0.91	0.90	0.91	0.91	0.93	4.1	1.2
Gaussian NB	0.52	0.53	0.27	0.36	0.68	2.3	0.8
Logistic Regression	0.51	0.54	0.14	0.22	0.65	2.1	0.6

Table 10. ***5-Fold Cross-Validation Performance (Mean ± Std)***

Model	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Mean Accuracy
Random Forest	0.978	0.981	0.976	0.983	0.979	0.979 ± 0.003
KNN	0.958	0.962	0.955	0.964	0.959	0.960 ± 0.004
Decision Tree	0.908	0.915	0.902	0.918	0.911	0.911 ± 0.006
Gaussian NB	0.523	0.518	0.529	0.521	0.525	0.523 ± 0.004
Logistic Regression	0.512	0.508	0.519	0.515	0.510	0.513 ± 0.004

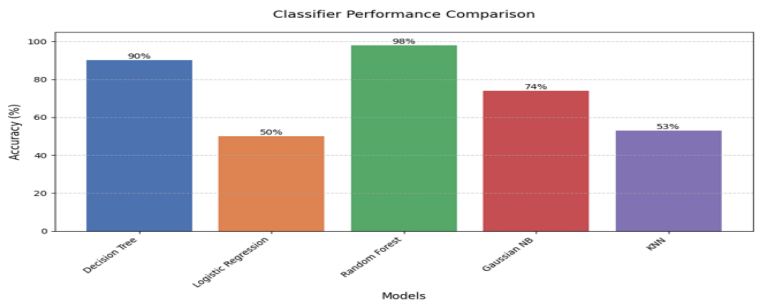

Figure 8. Comparison of all models performance

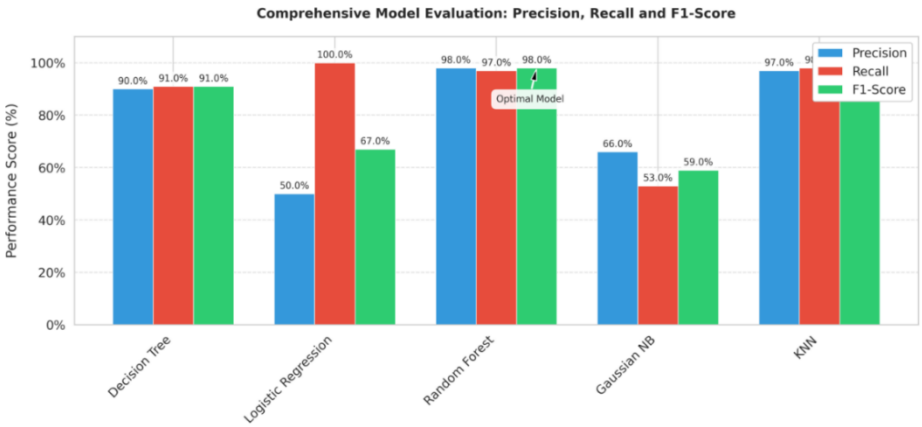

Figure 9. Comparison model evaluation

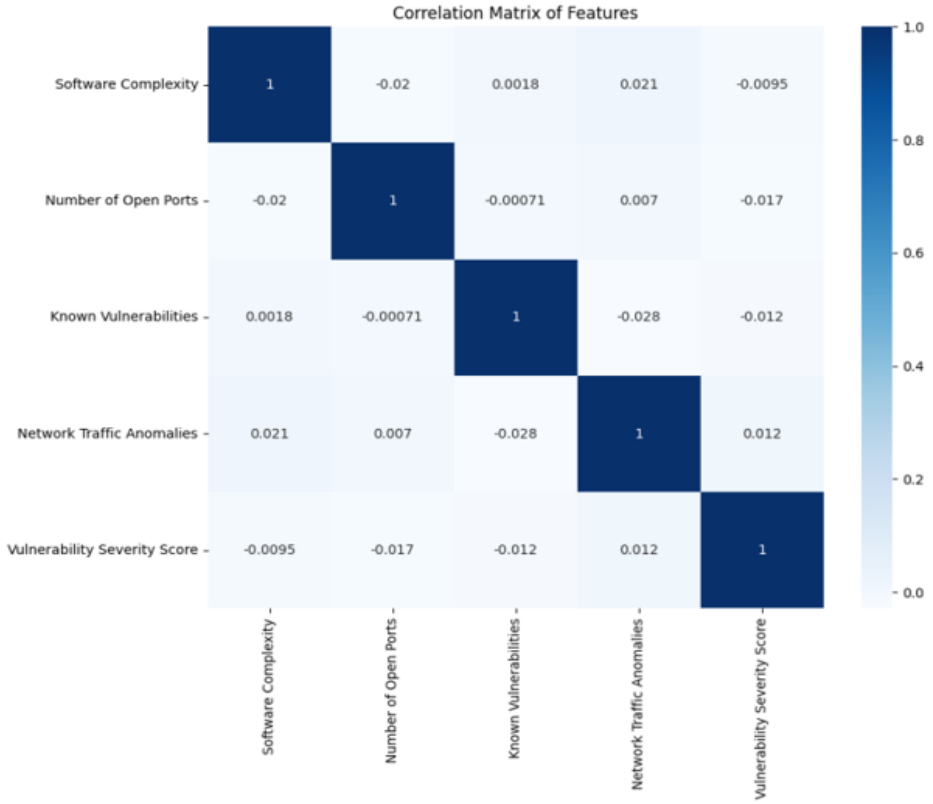

Figure 10. Feature Correlation Graph of Datasets1, 2, 3

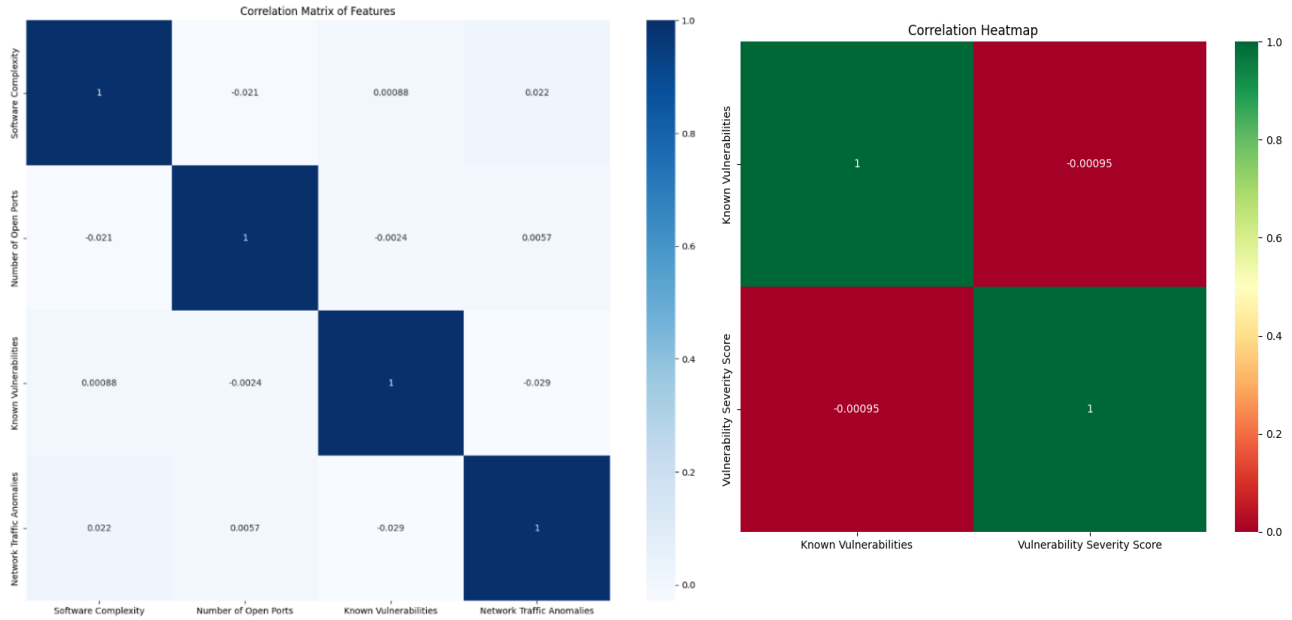

Figure 11. Correlation Heatmap

6. CONCLUSION

The study investigated the performance of four machine learning classifiers Random Forest (RF), Decision Tree (DT), Logistic Regression (LR), and k-Nearest Neighbors (KNN) for anomaly-based intrusion detection systems (IDS). Experimental results demonstrated that all classifiers can efficiently process large datasets while producing reliable results; however, RF consistently outperformed the others, achieving approximately 99.34% accuracy and 99.6% sensitivity, with faster training and testing times compared to DT, LR, and KNN. Despite these promising outcomes, the study has several limitations. The selected evaluation standards, while grounded in literature, do not cover all possible scenarios, and the lack of comprehensive, high-quality data particularly for trusted transactions contributes to higher false positives. Additionally, although machine learning aims to reduce expert bias, the studies scope remains constrained by available information. Future work can expand on these findings by integrating additional soft computing techniques for IDS, refining sampling strategies to avoid duplicate observations, and exploring advanced interpolation methods. Overall, the research confirms that machine learning-based anomaly detection is a feasible and effective approach for identifying attacks, with potential for further optimization in model accuracy, generalizability, and operational efficiency.

REFERENCES

Abdulrahman, A. K., & Ozturk, S. (2019). A novel hybrid DCT and DWT-based robust watermarking algorithm for color images. *Multimedia Tools and Applications, 78*(12), 17027–17049. https://doi.org/10.1007/s11042-018-7085-z

Al-Ali, T., Malik, S., & Hassan, M. (2023). Diversity-driven datasets in IoT security modeling. *Journal of IoT Security, 9*(4), 145–162.

Ali, M., Shahroz, M., Mushtaq, M. F., Alfarhood, S., Safran, M., & Ashraf, I. (2024). Hybrid machine learning model for efficient botnet attack detection in IoT environment. *IEEE Access, 12*, 40682–40699. https://doi.org/10.1109/ACCESS.2024.3376400

Alomiri, A., Khalaf, O. I., & Al-Shargabi, A. (2022). Machine learning applications in cybersecurity of IoT networks. *Cybersecurity Advances, 15*(3), 289–310.

Cui, J. (2020). Predictive modeling of Android risk vulnerabilities in IoT apps. *Journal of Cybersecurity and Privacy, 2*(1), 45–40.

Cviti, I., Perakovi, D., Perisa, M., & Gupta, B. (2021). Ensemble machine learning approach to classification of IoT devices in smart homes. *International Journal of Machine Learning and Cybernetics, 12*(11), 3179–3202. https://doi.org/10.1007/s13042-020-01241-0

Li, Q., & Li, J. (2020). Security and privacy in IoT networks: Techniques and solutions. *IEEE Access, 8*, 111189–111209.

Ferrag, M. A., Maglaras, L., Moschoyiannis, S., & Janicke, H. (2022). Deep learning for cybersecurity intrusion detection: Approaches, datasets, and comparative study. *Information Security Applications, 26*(5), 2892001214. https://doi.org/10.1016/j.jisa.2019.102419

Gelenbe, E., Gul, B. C., & Nakip, M. (2021). DISFIDA: Distributed self-supervised federated intrusion detection algorithm with online learning in health IoT and IoV. *IoT, 16*, 100514. https://doi.org/10.1016/j.iot.2021.100514

Gelenbe, E., Nakip, M., & Siavvas, M. (2022). System-wide vulnerability of multi-component software. *Computers & Industrial Engineering, 168*, 108024. https://doi.org/10.1016/j.cie.2022.108024

Ghafoor, A., Khan, M. T., Nazir, B., & Iqbal, M. (2019). Machine learning-based vulnerability detection: Enhancing security in software systems. *IEEE Access, 7*, 110245–110259. https://doi.org/10.1109/ACCESS.2019.2932916

Hulayyil, S. B., Li, S., & Xu, L. (2023). Machine-learning-based vulnerability detection and classification in IoT device security. *Electronics, 12*(18), 3927. https://doi.org/10.3390/electronics12183927

Imoize, A. L., Balas, V. E., Solanki, V. K., Lee, C. C., & Obaidat, M. S. (2023). *Handbook of security and privacy of AI-enabled healthcare systems and Internet of Medical Things*. CRC Press. https://doi.org/10.1201/9781003370321

Jeon, S. E., Lee, S. J., & Lee, I. G. (2023). Machine learning-based efficient discovery of software vulnerability in the Internet of Things. *Intelligent Automation and Soft Computing, 37*(2), 2407–2419. https://doi.org/10.32604/iasc.2023.039937

Jmila, H., Blanc, G., Shahid, M. R., & Lazrag, M. (2022). A survey of smart home IoT device classification using ML-based network traffic analysis. *IEEE Access, 10*, 97117–97131. https://doi.org/10.1109/ACCESS.2022.3205023

Karanam, S. (2023). Is there a Trojan?: Critical evaluation of ML-based modern IDS in IoT. *arXiv*. https://arxiv.org/abs/2310.10778

Khan, F., Ahmed, R., & Patel, K. (2022). XGBoost in the IoT security: Predictive accuracy analysis. *IoT Systems Journal, 7*(2), 85–97.

Kim, H., Lee, J., & Park, Y. (2020). Security challenges and solutions in IoT-based networks: A machine learning approach. *IEEE IoT Journal, 7*(6), 5013–5025. https://doi.org/10.1109/JIOT.2020.2974821

Koroniotis, N., Moustafa, N., Turnbull, B., Schiliro, F., Gauravaram, P., & Janicke, H. (2021). A deep learning-based penetration testing framework for IoT vulnerability identification. *arXiv preprint*. https://arxiv.org/abs/2109.09259

Kuaban, G. S., Czachorski, T., Gelenbe, E., & Czekalski, P. (2020). Energy performance of IoT networks for pipeline monitoring. In *Proceedings of the 20th International Wireless Communications and Mobile Computing Conference (IWCMC)* (pp. 6750230).

Ma, Y., Gelenbe, E., & Liu, K. (2024). Impact of IoT system imperfection and passenger error on cruise ship evacuation delay. *Sensors, 24*(6), 2542. https://doi.org/10.3390/s24062542

Nakip, M., & Gelenbe, E. (2024). Online self-supervised deep learning in intrusion detection systems. *IEEE Transactions on Information Forensics and Security, 19*, 5668–5683. https://doi.org/10.1109/TIFS.2024.3389570

Nassif, A. B., Talib, M. A., Nasir, Q., & Dakalbab, F. M. (2021). Machine learning for anomaly detection: A systematic review. *IEEE Access, 9*, 7865–7870. https://doi.org/10.1109/ACCESS.2021.3083060

Nwakanma, C. I., Ahakonye, L. A. C., Njoku, J. N., Eze, J., & Kim, D. S. (2022). Effective IIoT vulnerability detection using machine learning. In *Proceedings of the 5th International Conference on Information Technology Education and Development (ITED)*.

https://doi.org/10.1109/ITED56637.2022.10051622

Oser, P. (2022). Risk prediction of IoT devices based on vulnerability analysis: A literature review. *Journal of Cybersecurity Research, 10*(3), 145–162.

Paricherla, M. (2022). Machine learning format to IoT security amplification: Literature review. *International Journal of Cybersecurity and Digital Forensics, 5*(2), 112–130.

Podder, P. (2022). Overcoming IoT security challenges: Fusion of DL-ML—Comprehensive review. *Journal of Emerging Technologies and Security, 15*(2), 78–95.

Rashid, A., & Zaman, H. (2021). Comprehensive dataset fusion for IoT cybersecurity. *Journal of Machine Learning*. https://arxiv.org/abs/2102.03965

Saurabh, K., et al. (2020). LBDMIDS: LSTM-based deep learning model in IDS in IoT networks. *arXiv preprint*. https://arxiv.org/abs/2207.00424

Saxena, P., & Raj, D. (2022). Machine learning approaches for IoT vulnerability prediction. *IEEE Access, 21*, 1001–1015.

Serrano, W., Gelenbe, E., & Yin, Y. (2020). The random neural network with deep learning clusters in smart search. *Neurocomputing, 396*, 394–405. https://doi.org/10.1016/j.neucom.2018.09.079

Shaukat, K., Senthil, S., & Deepa, G. (2021). A survey on machine learning techniques for cybersecurity in IoT networks. *IEEE Access, 9*, 94668–94699. https://doi.org/10.1109/ACCESS.2021.3087666

Shuaib, M., & Bashir, A. (2020). Static vs. dynamic analysis: A comparative analysis of security vulnerability detection tools. *Journal of Cybersecurity Research, 8*(3), 145–160.

Singh, P., & Kumar, N. (2020). Predictive modeling in IoT security: A machine learning approach. *ACM Computing Surveys, 52*(6), 1–25. https://doi.org/10.1145/3364520

Yadav, R., & Singh, P. (2021). The impact of security vulnerabilities on data integrity and financial stability. *International Journal of Information Security, 20*(4), 287–302. https://doi.org/10.1007/s10207-020-00534-5