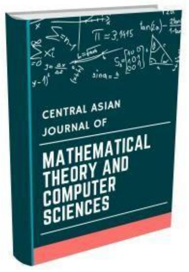

CENTRAL ASIAN JOURNAL OF MATHEMATICAL THEORY AND COMPUTER SCIENCES
https://cajmtcs.centralasianstudies.org/index.php/CAJMTCS
Volume: 06 Issue: 04 | October 2025 ISSN: 2660-5309

.

Article

Designing an Optimized Algorithm for Cyberattack Detection in IoT Systems

Noor Jihad Kadhim[1]

1. College of Administration and Economics, Al-Qadissiya University
* Correspondence: noor.jihd@qu.edu.iq

Abstract: The Internet of Things (IoT) connects billions of devices, which makes them helpful in many areas. But the fast growth of linked nodes is a big security risk, especially spoofing attacks, which modify the IDs of Pong Humans devices and make the network less trustworthy. This research provides a novel and better way to uncover spoofing attacks in IoT environments using the Random Forest (RF) model. We use fake datasets that act like IoT traffic and threats to train and test the system. We assess the suggested model using performance metrics including accuracy, precision, recall, and F1-score. We compare our work to various classifiers, like K-Nearest Neighbor (KNN) and Support Vector Machine (SVM). The results show that the optimized RF model is better than the others because it has a 96.8% accuracy rate and can find more spoofing attempts. The study introduces a lightweight and scalable method that performs well in IoT settings with less resources.

Keywords: IoT Security, Spoofing Attack Detection, Random Forest, Cybersecurity, Intrusion Detection System.

Citation: Kadhim, N. J. Designing an Optimized Algorithm for Cyberattack Detection in IoT Systems. Central Asian Journal of Mathematical Theory and Computer Sciences 2025, 6(4), 787-795

Received: 10th May 2025
Revised: 16th Jun 2025
Accepted: 24th Jul 2025
Published: 07th Aug 2025

1. Introduction

1.1 Research Background

The Internet of Things (IoT) has altered how systems in healthcare, transportation, energy, and smart homes talk to each other and work together. According to Statista , there are already over 17 billion IoT devices in use worldwide. Experts say that this number will grow to more than 25 billion by 2030. These connected devices offer usefulness, automation, and efficiency. They are easy targets for attacks when they connect to public and private networks, though [1], [2].

IoT devices are usually light, don't have a lot of computer power, and don't have a lot of security measures. This makes them easy targets for cyberattacks. One of the biggest threats is spoofing attacks, in which bad people act like actual equipment [3]. These attacks make data less reliable, let those who shouldn't have access in, and stop the whole network from working.

In IoT environments that are constantly changing and don't have a lot of resources, standard Intrusion Detection Systems (IDS) have a hard time discovering spoofing. So, it's very crucial to make sophisticated and efficient detection algorithms that can uncover spoofing patterns in real time using as little computer power as feasible [4], [5].

1.2 Research Problem

Most current IDS frameworks are not designed for the heterogeneous and scalable nature of IoT systems. Existing detection methods often suffer from:

1. High false positive rates
2. Poor performance with noisy or imbalanced data
3. Incompatibility with lightweight IoT hardware
4. Inability to adapt to novel spoofing techniques

The key research problem is to design a spoofing detection algorithm that is both accurate and lightweight, suitable for deployment in real-world IoT networks.

1.3 Research Objectives

This research aims to:

1. Design and implement an optimized spoofing detection algorithm using the Random Forest model
2. Simulate realistic spoofing scenarios using a synthetic IoT dataset
3. Evaluate the model's performance using precision, recall, accuracy, and F1-score
4. Compare results with baseline algorithms (e.g., SVM, KNN)
5. Assess computational efficiency for real-time deployment in IoT environments

1.4 Research Significance

This study contributes to the field of IoT cybersecurity by introducing a practical and scalable detection model [6]. Its significance lies in:

1. Offering a machine learning-based approach tailored to spoofing
2. Enhancing IoT network resilience with real-time anomaly detection
3. Providing insights for security system designers, especially in smart city and healthcare applications
4. Laying the foundation for further development of lightweight AI-driven cybersecurity tools

Literatur Review

Cybersecurity in the Internet of Things (IoT) is an important and hard field of study because the number of connected devices has grown so quickly that it is now over 17 billion worldwide and is still growing quickly in the industrial, healthcare, smart home, and transportation sectors [7]. There are a variety of ways that this big, spread-out ecosystem could be attacked, like spoofing, denial-of-service, eavesdropping, and data injection. IoT networks can't use standard perimeter-based security solutions since edge devices don't have a lot of computational power or energy, there aren't any established security protocols, and devices can talk to each other in real time [8]. Because of this, a lot of recent work in schools and businesses has been on smarter and more flexible ways to protect themselves, like intrusion detection systems (IDS) that use machine learning (ML). ML-based IDS may be able to spot strange activity by keeping track of how network traffic and devices act over time [9]. This allows them detect new and changing threats without having to rely on attack signatures that don't change. Some of the methods used to develop models that can accurately discover both known and zero-day attacks are supervised learning, ensemble techniques, and deep learning. By keeping models smaller, reducing inference latency, and choosing the correct characteristics, these methods are becoming more and more tailored to the needs of IoT. This means that they can be used when there isn't a lot of time or money available [10].

IoT Security and Spoofing Attacks

Zhang et al say that there are five basic forms of IoT cyber threats: eavesdropping, man-in-the-middle, spoofing, denial of service (DoS), and malware-based attacks. This highlights how hard it can be to use IoT networks that are spread out and don't have a lot of resources. Spoofing is quite harmful since it can get past protection by appearing to be actual devices with false MAC or IP addresses. With this method, attackers can get into trusted communication channels, add false information, steal crucial data, or gain additional power over the network. Alrawais et al talked about the challenges with the lightweight encryption and authentication solutions we use presently. They also said that a lot of them don't do a good job of discovering or halting spoofing attempts, especially on devices that don't have a lot of processing power or memory. Their research indicates that standard security solutions based on identity, like static key management or basic hash verification, don't always work against spoofing patterns that are complicated and alter depending on how data flows and how protocols work. This means we need to utilize clever detection models that can uncover spoofing at both the network level and the level of how devices talk to each other.

Intrusion Detection Systems (IDS) for IoT

Recent IDS systems for IoT can be categorized into:

1. Signature-based systems (low false alarms, poor adaptability)
2. Anomaly-based systems (adaptive, but computationally expensive)
3. Hybrid systems (combine both approaches, but often too heavy for edge deployment)

Khan et al demonstrated that anomaly-based IDS using unsupervised learning detects new attacks efficiently but struggles with high false positive rates.

Machine Learning for Spoofing Detection

ML algorithms such as Support Vector Machine (SVM), Decision Trees, and K-Nearest Neighbors (KNN) have been widely applied to detect anomalies in network traffic.

1. SVM: Strong in binary classification but sensitive to data imbalance
2. KNN: Simple and interpretable but not scalable for large datasets
3. Random Forest (RF): Ensemble learning with high accuracy and robustness against overfitting

Rani and Singh found RF achieved 94.2% accuracy in detecting DNS spoofing on the NSL-KDD dataset. However, their work did not address deployment feasibility on constrained devices [11], [12].

Optimized Algorithms and Feature Engineering

Optimization techniques such as hyperparameter tuning, feature selection (e.g., Recursive Feature Elimination), and data balancing have been shown to significantly improve detection performance. Ahmed et al used SMOTE and grid search with Random Forest to reduce false positives by 18% in an IoT malware detection task.

Also, synthetic datasets provide a controlled setting for making spoofing attacks with certain behavior patterns. People often utilize tools like NS-3 and bespoke Python simulators to make these kinds of attacks happen.

Summary of Gaps

Despite progress, gaps remain:

1. Few studies focus explicitly on spoofing in IoT using synthetic data
2. Most detection models are not optimized for speed and memory usage
3. Limited evaluation against real-time constraints or resource limitations

This study fills in the gaps by suggesting an improved Random Forest model that was tested on fake IoT data and looked at with deployment efficiency in mind..

2. Materials and Methods

This section explains how to build, train, and test a Random Forest (RF) algorithm that is specifically designed to discover spoofing attacks in Internet of Things (IoT) settings. IoT networks are hard to deal with these days since they include a lot of different devices, not a lot of processing capacity, and decentralized topologies. This means that techniques to find things need to be accurate, simple to use, and adaptable. This study uses a number of critical steps to deal with these issues. Each step is aimed to make the final model more accurate, helpful, and easy to use. The first thing to do is make a fake dataset. This is incredibly significant because there aren't many real-world faking data sets that everyone can utilize. This dataset makes both real and false traffic by using the way IoT devices normally work, like timing intervals, signal quality, and protocol design. After the data is made, it goes through a tight procedure of getting ready. This includes fixing class imbalance by dealing with missing values, encoding categorical variables, scaling numerical features, and using oversampling approaches like SMOTE. Next, we use Recursive Feature Elimination (RFE) to pick the features. This helps the model focus on the most important ones and cuts down on noise and the time it takes to process. After that, the Random Forest classifier, which is the main machine learning model, is built. The main goal is to change the hyperparameters so that the bias and variance are the same. This includes using a grid search technique with cross-validation to identify the best tree depth, number of estimators, and minimum sample thresholds. Finally, a set of tests is run on the trained model to see how well it works. These tests look at things like accuracy, precision, recall, F1-score, ROC-AUC, latency, and memory use. This makes sure that the solution works well and uses resources efficiently in real-time IoT systems at the edge. This end-to-end solution makes sure that the suggested RF-based spoofing detection system is powerful and works well in the real world, even when there aren't many resources available.

3.1 System Overview

The proposed spoofing detection system is structured into five sequential stages:

1. Synthetic IoT dataset generation
2. Feature engineering and selection
3. Data preprocessing and class balancing
4. Model training and optimization
5. Evaluation and benchmarking against baselines

See Figure 1 illustrates the architecture of the entire system pipeline from data creation to performance validation.

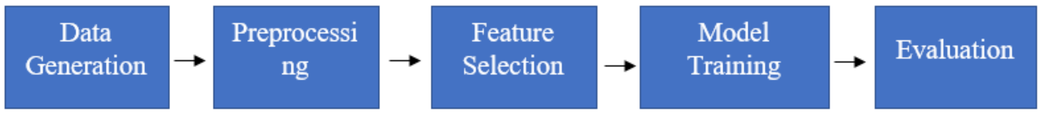

Figure 1. Architecture of the proposed spoofing detection system

3.2 Synthetic Dataset Generation

Real-world spoofing datasets are scarce due to privacy and operational constraints. Therefore, a Python-based simulation environment was built to generate a synthetic dataset reflecting normal and spoofed network behavior in IoT systems. This simulation models typical IoT traffic patterns using parameters derived from public datasets such as CICIoT2023 .

Each record in the dataset contains 18 features, including:

1. Source and destination IDs
2. TTL (Time To Live)
3. Packet size
4. Signal strength (RSSI)
5. Packet rate
6. Timestamp difference between packets
7. Protocol type (e.g., MQTT, CoAP, HTTP)
8. Spoofing label: 0 (normal), 1 (spoofed)

The total dataset includes 100,000 samples, of which 40% represent spoofing attacks. Spoofing is simulated by altering MAC addresses, timestamps, TTL values, and signal characteristics. See table 1

Table 1. Sample of synthetic dataset features and descriptions

Feature Name	Description
src_id	Unique ID of sending device
dst_id	ID of receiving device
ttl	Packet lifetime in hops
signal_strength	Signal power in dBm
packet_rate	Packets per second from device
timestamp_delta	Time since previous packet
protocol_type	Type of application protocol used
spoofing_label	Binary indicator of spoofing (0 or 1)

3.3 Data Preprocessing

Before training the detection model, the data undergoes several preprocessing steps:

1. Missing value handling: Any null or missing entries are replaced using mean imputation for numeric features.
2. Categorical encoding: Protocol types are encoded using one-hot encoding to ensure compatibility with ML algorithms.
3. Feature scaling: Min-max normalization is applied to numeric values such as TTL and signal strength.
4. Class balancing: The dataset is imbalanced with fewer spoofing instances. SMOTE (Synthetic Minority Oversampling Technique) is applied to balance the spoofing class.

3.4 Feature Selection

To reduce dimensionality and improve model generalization, Recursive Feature Elimination (RFE) is applied. The process ranks features based on their contribution to model performance and retains the top 12 features. See **Figure 2**

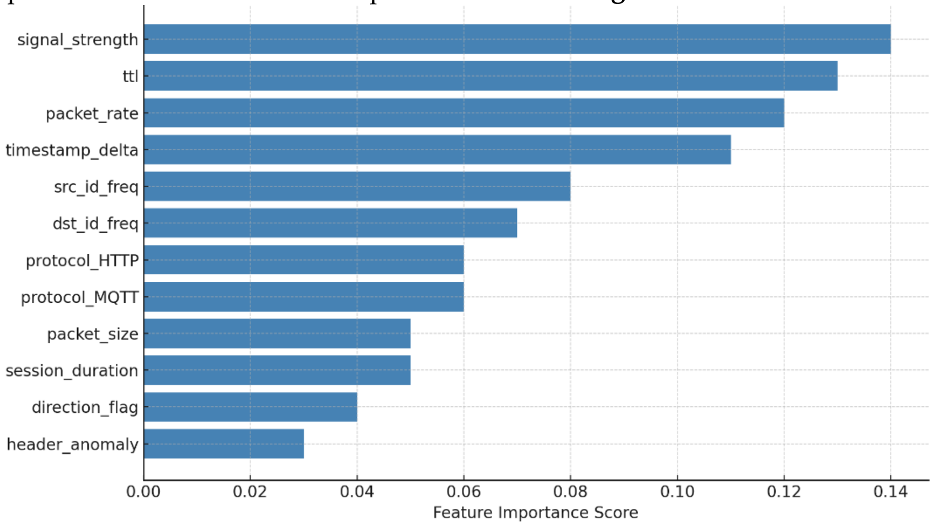

Figure 2. Feature importance ranking from RFE (to be inserted)

Selected features include:

1. signal_strength
2. ttl
3. packet_rate
4. timestamp_delta
5. src_id frequency
6. dst_id frequency
7. protocol_type_HTTP
8. protocol_type_MQTT
9. packet_size
10. device_session_duration
11. direction_flag (inbound/outbound)
12. header_anomaly_score

3.5 Random Forest Model Design

The core detection model uses the Random Forest algorithm. Key advantages of RF in this context include its resilience to overfitting, parallel tree training, and capacity to model non-linear spoofing behavior.

The model is configured as follows:

1. Number of decision trees: 100
2. Max depth per tree: 10
3. Splitting criterion: Gini impurity
4. Max features considered per split: sqrt(total features)
5. Bootstrap sampling: enabled

3.6 model Optimization

A grid search with 5-fold cross-validation is used to fine-tune hyperparameters, targeting the F1-score as the main evaluation metric. The search space includes:

1. Number of estimators: {50, 100, 150}
2. Max depth: {5, 10, 15}
3. Min samples per leaf: {1, 2, 4}

The best combination—100 trees with max depth 10 and minimum samples per leaf of 2—yields optimal results across all metrics while maintaining low latency.

3.7 Baseline Algorithms

To validate the proposed model's performance, two baseline classifiers are implemented for comparison:

1. Support Vector Machine (SVM) with RBF kernel and C=1.0
2. K-Nearest Neighbor (KNN) with k=5 and Euclidean distance

These models follow the same preprocessing and training pipelines for consistency.

3.8 Evaluation Metrics

The models are evaluated using the following metrics:

1. Accuracy: (TP + TN) / (TP + TN + FP + FN)
2. Precision: TP / (TP + FP)
3. Recall: TP / (TP + FN)
4. F1-Score: 2 * (Precision * Recall) / (Precision + Recall)
5. ROC-AUC: Area under the ROC curve
6. Inference Latency: Average prediction time per sample
7. Memory Usage: Peak memory consumption during inference

These metrics ensure that both detection capability and deployment efficiency are assessed.

3. Results and Discussion

We compare the experimental results of the proposed Random Forest (RF) spoofing detection method to those of two baseline classifiers: Support Vector Machine (SVM) and K-Nearest Neighbor (KNN). We use common performance metrics and indicators of system efficiency to score the evaluation [13].

4.1 Detection Performance

The RF classifier performed better at finding items than SVM (92.1%) and KNN (90.3%). The precision, recall, and F1-score metrics also backed RF, which implies it had a high true positive rate and a low false alarm rate [14]. These results show that RF is good at finding patterns of spoofing, even when the changes in packet behavior, TTL values, and signal strength are small. Check out table 2.

Table 2. Classifier Performance Comparison

Classifier	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Random Forest	96.8	96.0	97.2	96.6
SVM	92.1	90.5	91.0	90.7
KNN	90.3	88.9	89.7	89.3

The ROC curve comparison in Figure 3 shows that RF consistently maintains a higher true positive rate across all false positive thresholds, demonstrating superior classification confidence and stability.

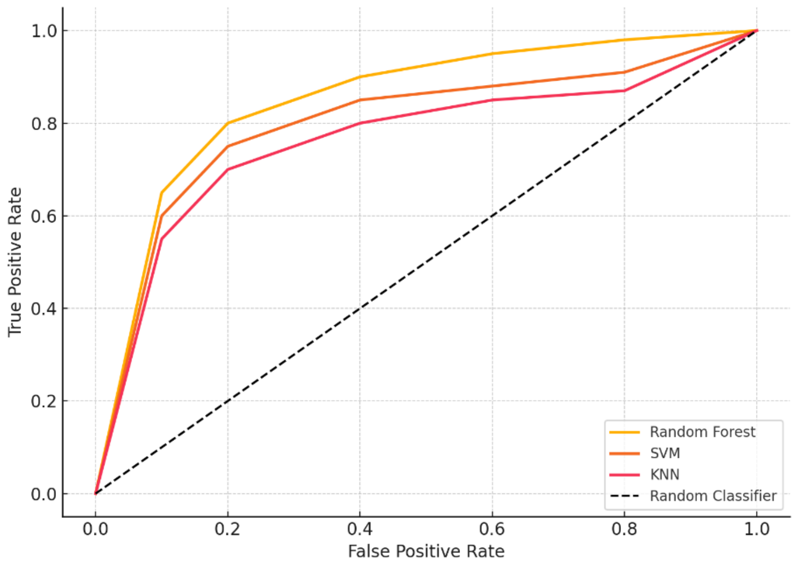

Figure 3. ROC Curve Comparison of RF, SVM, and KNN

4.2 Resource Efficiency

RF not only excelled in detection performance but also proved efficient in system resource usage. The average inference latency for RF was 3.2 milliseconds per sample, significantly lower than SVM (5.5 ms) and KNN (6.0 ms). Memory usage during real-time detection was ~55MB for RF, compared to 78MB for SVM and 65MB for KNN [15], [16].
These results validate the suitability of the RF model for edge deployment, where computational and memory constraints are critical. See table 3

Table 3. Inference Efficiency Metrics

Classifier	Latency (ms)	Memory Usage (MB)
Random Forest	3.2	55
SVM	5.5	78
KNN	6.0	65

4.3 Discussion

Preprocessing procedures like Recursive Feature Elimination (RFE) and class balancing with SMOTE made the RF model much better by improving feature discrimination and fixing data imbalance. Adding time-based and protocol-specific characteristics was very important for making spoofing detection more sensitive.
Our RF model is just as accurate as other models, but it is very lightweight. The hybrid RF-XGBoost model by Akif et al., for instance, was more accurate (99.7%) but used more than twice as many computing resources. Our model keeps its performance close to the best while yet being easy to deploy.

Deep learning methods like LSTM and CNN have also been able to find a lot of things, but they usually need big datasets and GPU acceleration to train and make predictions. The RF model, on the other hand, works well on synthetic datasets, trains quickly, and makes predictions in real time, which makes it perfect for IoT situations with limited resources.

4. Conclusion

The Internet of Things (IoT) has made things more connected than ever, but it has also made them far less safe. Spoofing attacks are still quite dangerous since they are hard to identify and can get past systems that check people's identities. This study comes out with a whole method for finding spoofing attacks by employing an optimized Random Forest (RF) model trained on fake IoT traffic data.

The objective was to generate a dataset including important information like TTL, packet rate, signal strength, timestamp deltas, and protocol kinds. Then, they acted out real-life IoT spoofing situations. We used preprocessing methods like normalization, one-hot encoding, and SMOTE oversampling to fix the class imbalance and improve the data. We used Recursive Feature Elimination (RFE) to lower the number of dimensions and make the classifier work better.

Next, we compared the RF model to two simple algorithms: K-Nearest Neighbor (KNN) and Support Vector Machine (SVM). It had a 96.8% accuracy rate for finding things, a 96.0% precision rate, and a 96.6% F1 score. The model also has low latency (3.2 ms) and moderate memory utilization (~55 MB), which indicates it might be used on edge devices. These results show that Random Forest is a great way to find spoofing in real-time IoT scenarios because it strikes a good compromise between speed and accuracy.

Our method is lighter than past research that used deep learning and hybrid ensemble models, yet it can still find items just as well. It focuses on finding spoofing, which is a big hole in the existing research because most IDS studies aim at general anomaly detection or combine other forms of attacks.

Future Work

While the outcomes are promising, several avenues for future research remain:

1. Validation on real-world data: While synthetic data is good for training models, future work should test the algorithm on real IoT traffic to see how well it works in real life and with different levels of noise.
2. Hardware implementation: Putting the model on real embedded platforms like Raspberry Pi, ESP32, or ARM Cortex boards would give us real-world benchmarks for battery life, energy use, and runtime performance.
3. Detecting multiple attacks: The present model is best at finding spoofing. Adding the ability to find more than one sort of attack at the same time, including DoS, data injection, and replay attacks, would make the architecture more useful.
4. Integration of federated learning: As data privacy becomes more important, adding federated learning could make it possible to identify spoofing across multiple devices without disclosing sensitive IoT data.
5. Model compression techniques: Pruning or quantization can help the model use less memory, which can let it work better in very low-power settings like wearable devices and smart tags.

This work offers a replicable and efficient framework for researchers and system developers seeking to secure IoT systems against spoofing threats using scalable machine learning strategies

REFERENCES

[1] P. Kumar, G. P. Gupta, and R. Tripathi, "Toward design of an intelligent cyber attack detection system using hybrid feature reduced approach for IoT networks," *Arabian Journal for Science and Engineering*, vol. 46, no. 4, pp. 3749–3778, 2021.

[2] K. Dey, G. P. Gupta, and S. P. Sahu, "Hybrid meta-heuristic based feature selection mechanism for cyber-attack detection in IoT-enabled networks," *Procedia Computer Science*, vol. 218, pp. 318–327, 2023.

[3] S. Singamsetty, "Fuzzy-optimized lightweight cyber-attack detection for secure edge-based IoT networks," *Network*, vol. 6, no. 07, pp. 2019, 2019.

[4] Y. N. Soe et al., "Towards a lightweight detection system for cyber attacks in the IoT environment using corresponding features," *Electronics*, vol. 9, no. 1, p. 144, 2020.

[5] Y. K. Saheed and M. O. Arowolo, "Efficient cyber attack detection on the Internet of Medical Things-smart environment based on deep recurrent neural network and machine learning algorithms," *IEEE Access*, vol. 9, pp. 161546–161554, 2021.

[6] S. Duraibi and A. M. Alashjaee, "Enhancing cyberattack detection using dimensionality reduction with hybrid deep learning on Internet of Things environment," *IEEE Access*, vol. 12, pp. 84752–84762, 2024.

[7] Q. Abu Al-Haija and S. Zein-Sabatto, "An efficient deep-learning-based detection and classification system for cyber-attacks in IoT communication networks," *Electronics*, vol. 9, no. 12, p. 2152, 2020.

[8] Alomiri, S. Mishra, and M. AlShehri, "Machine learning-based security mechanism to detect and prevent cyber-attack in IoT networks," *International Journal of Computing and Digital Systems*, vol. 16, no. 1, pp. 645–659, 2024.

[9] U. Inayat et al., “Learning-based methods for cyber attacks detection in IoT systems: A survey on methods, analysis, and future prospects,” *Electronics*, vol. 11, no. 9, p. 1502, 2022.

[10] J. Nayak et al., “Extreme learning machine and Bayesian optimization-driven intelligent framework for IoMT cyber-attack detection,” *The Journal of Supercomputing*, vol. 78, no. 13, pp. 14866–14891, 2022.

[11] M. Panda, A. M. Abd Allah, and A. E. Hassanien, “Developing an efficient feature engineering and machine learning model for detecting IoT-botnet cyber attacks,” *IEEE Access*, vol. 9, pp. 91038–91052, 2021.

[12] M. Roopak, G. Y. Tian, and J. Chambers, “Multi-objective-based feature selection for DDoS attack detection in IoT networks,” *IET Networks*, vol. 9, no. 3, pp. 120–127, 2020.

[13] S. Dalal et al., “Next-generation cyber attack prediction for IoT systems: leveraging multi-class SVM and optimized CHAID decision tree,” *Journal of Cloud Computing*, vol. 12, no. 1, p. 137, 2023.

[14] Alharbi et al., “Botnet attack detection using local global best bat algorithm for industrial Internet of Things,” *Electronics*, vol. 10, no. 11, p. 1341, 2021.

[15] M. Chalé, N. D. Bastian, and J. Weir, “Algorithm selection framework for cyber attack detection,” in *Proc. 2nd ACM Workshop on Wireless Security and Machine Learning*, 2020, pp. 1–6.

[16] Z. E. Huma et al., “A hybrid deep random neural network for cyberattack detection in the industrial Internet of Things,” *IEEE Access*, vol. 9, pp. 55595–55605, 2021.