IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Journal Européen des Systèmes Automatisés
Vol. 58, No. 5, May, 2025, pp. 895-900
Journal homepage: http://iieta.org/journals/jesa

Machine Learning-Based Intrusion Detection System: Reducing False Positives and False Negatives in IoT Security

Mourad Hana[1*], Oussama El Haouari[2], Ghizlane Khaissidi[1], Mostafa Mrabti[1]

[1] Laboratoire d'Ingénierie, Systèmes et Applications, Sidi Mohamed Ben Abdellah University, Fez 30000, Morocco
[2] Laboratory of Applied Sciences and Emerging Technologies, Sidi Mohamed Ben Abdellah University, Fez 30000, Morocco

Corresponding Author Email: mourad.hana@usmba.ac.ma

https://doi.org/10.18280/jesa.580503

Received: 4 April 2025
Revised: 12 May 2025
Accepted: 21 May 2025
Available online: 31 May 2025

Keywords:
intrusion detection, IoT security, machine learning, cyber threats, malware classification, false positives reduction

ABSTRACT

With the rise of IoT devices in homes, factories, hospitals, and nearly every modern setting, keeping these systems safe from cyber threats is more complicated than ever. intrusion detection systems (IDSs) are supposed to help by flagging unusual activity, but they often raise too many false alarms or worse, miss actual attacks. In this study, we take a more down-to-earth approach by using machine learning to make IDSs work better in IoT environments. We selected a filtered version of the CTU-IoT-Malware-Capture dataset and tested four classification models: Random Forest, Naïve Bayes, Decision Tree, and Logistic Regression. The focus was on solid data preparation and smart feature selection, which turned out to improve detection accuracy quite a bit. We also looked at how heavy each model is in terms of computing resources, especially since real-time response is a key concern for IoT systems. The idea is to move toward something practical an IDS that's not only accurate but lightweight enough to be used outside the lab.

1. INTRODUCTION

The Internet of Things (IoT) has expanded rapidly over the past few years, becoming essential in areas like healthcare, transportation, smart homes, and industry. While this growth has brought clear advantages such as increased automation and operational efficiency it has also exposed networks to a wide range of security vulnerabilities. Many IoT devices still lack proper security mechanisms, making them easy targets for attackers. In fact, recent reports show that IoT-based cyberattacks surged by more than 35% in 2023 alone [1], highlighting just how urgent the need for stronger protection has become.

Traditionally, intrusion detection systems (IDSs) have played a key role in identifying malicious behavior. However, most of these systems depend on static signatures or fixed rule sets, which often fail when faced with unfamiliar or rapidly evolving threats [2]. To address these shortcomings, researchers have increasingly turned to machine learning (ML) as a more adaptable alternative. ML models can detect suspicious activity by learning from past traffic patterns, even when the nature of the threat isn't known in advance.

That said, applying ML in this context comes with its own challenges. A major issue is class imbalance: in most datasets, benign traffic far outweighs malicious samples, which makes it harder for the model to learn how to recognize rare but critical attacks [3]. Another problem lies in the nature of IoT traffic itself it often contains irrelevant or repetitive features that can negatively affect detection accuracy and slow down performance, especially on devices with limited processing power [4].

To tackle these problems, researchers have explored a range of preprocessing strategies. Feature selection techniques like Recursive Feature Elimination (RFE) and Principal Component Analysis (PCA) help reduce data complexity while keeping the most meaningful variables. At the same time, oversampling approaches such as SMOTE and ADASYN have been widely used to balance datasets and improve the model's sensitivity to minority classes [5, 6]. On top of that, ensemble learning methods like Random Forest and gradient boosting have shown promising results in managing noisy data and minimizing overfitting [7, 8].

Still, many IDS systems continue to produce too many false positives or miss real attacks. In many cases, this is due to weak feature selection, unbalanced training data, or a lack of proper tuning. Another issue is that some studies rely on outdated datasets, which don't capture the fast-changing nature of modern IoT traffic [9, 10]. These limitations suggest that there's still room to improve current approaches, especially for practical, real-world deployment.

This paper proposes a machine learning-based intrusion detection framework specifically designed to address these gaps. Our approach combines Recursive Feature Elimination for feature selection, SMOTE for class balancing, and grid search for tuning model parameters. We validate the framework using the CTU-IoT-Malware-Capture dataset, which contains realistic malware traffic from various IoT devices. To evaluate the impact of each preprocessing step,

we test four common supervised learning models: Random Forest, Decision Tree, Logistic Regression, and Naïve Bayes.

The main contributions of this work are:

• A full pipeline that integrates feature selection, data resampling, and tuning for better intrusion detection performance in IoT settings.

• A thorough evaluation of each model using metrics like accuracy, ROC-AUC, and Matthews Correlation Coefficient (MCC).

• A comparative review with recent intrusion detection frameworks to show how our method performs in relation to existing approaches.

The rest of the paper is structured as follows: Section 2 reviews related work. Section 3 describes the dataset and preprocessing pipeline. Section 4 details the methodology. Section 5 presents the experimental results. Section 6 discusses the findings, and Section 7 concludes with suggestions for future work.

2. RELATED WORK

In recent years, intrusion detection for IoT systems has received a lot of attention, and for good reasons. As the number of connected devices keeps growing, so does the risk of cyberattacks. Traditional IDS approaches that rely on static rules or known attack patterns are no longer enough. They simply can't keep up with new and unexpected threats like zero-day attacks [11, 12].

Many researchers have started looking into machine learning and deep learning as possible solutions. Some of the earlier work in this area includes Ullah et al. [11], who used recurrent neural networks to detect unusual behavior in vehicle networks. Their method focused on capturing time-based features. In another study, Odeh and Abu Taleb [13] applied CNNs in healthcare systems hosted in the cloud. Their model showed promising results, but like many DL approaches, it required a lot of computational resources. Diro and Chilamkurti [14] went further by proposing a distributed framework, aiming to make deep learning more suitable for large, varied IoT environments.

Feature selection has also been a major focus. Banaamah and Ahmad [15] combined deep learning features with traditional ones and reported better performance. Issa et al. [16] used a hybrid selection method to clean up the input space, and Hussein et al. [17] used correlation-based filters to pick out the most useful features.

More recently, a few interesting approaches have emerged. Alzahrani et al. [18] developed a CNN-LSTM model that runs in real time, something that's still a challenge in IoT systems. Ren et al. [19] used attention mechanisms to reduce false alarms, and Mosaiyebzadeh et al. [20] explored federated learning to protect user data while still training a strong IDS model. Sharmila et al. [21] proposed compressing features using autoencoders to make the models lighter and faster, and Farooqi et al. [22] worked on detecting rare attacks by combining SMOTE and boosting techniques.

Some researchers have also tried to bring together multiple techniques. Ayad et al. [23], for example, combined SMOTE with Recursive Feature Elimination and tested it on recent datasets like NSL-KDD and IoT-23. They reported lower false negatives. Elsaid et al. [24] took another route and applied the Grey Wolf Optimizer to improve CNN performance. Finally, Sharma et al. [25] brought in explainable AI to help security analysts understand the model's decisions a much-needed step if these systems are to be used in the real world.

What we noticed while reviewing this literature is that many works focus on just one part of the pipeline either balancing the data or tuning the model or selecting features but not all of them together. That's where our work comes in. We try to put it all in one place: using RFE to reduce noise, SMOTE to balance the data, and grid search to tune the classifiers. We also go beyond accuracy by looking at metrics like MCC and ROC-AUC, to really understand how well the models perform. Our goal is to offer something that's not just accurate in theory, but also practical and reliable for real IoT applications.

3. METHODOLOGY

In this study, we worked with the trimmed_CTU-IoT-Malware-Capture dataset, which offers real-world traffic traces from IoT devices. It includes a mix of normal device activity alongside different types of cyberattacks, such as DDoS, scanning, botnet behavior, and data exfiltration [13]. This dataset has been widely used in IoT security research, making it a strong foundation for evaluating intrusion detection models. The overall workflow of the proposed system is illustrated in Figure 1.

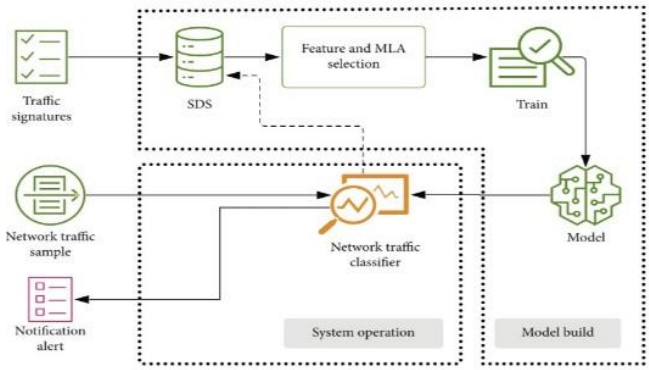

Figure 1. System architecture

3.1 Data preprocessing

Before training the models, we carefully cleaned the CTU-IoT-Malware-Capture dataset to make sure it was suitable for machine learning. The raw data included a mix of normal and before getting into model training, we spent time cleaning the CTU-IoT-Malware-Capture dataset to make it more suitable for machine learning tasks. The raw data reflected a mix of regular and malicious traffic something you'd expect in a real IoT environment.

We started by removing duplicates and any entries that were clearly corrupted or unusable. For numeric fields with missing values, we chose median imputation. It gave us a way to preserve the central tendency of the data without letting outliers skew the results. In cases where a record had too many missing fields, we dropped it altogether. This follows standard practice in IDS research, where incomplete data tends to degrade performance [26].

Once we had a cleaner dataset, we normalized all continuous features using Min-Max scaling. This step helped bring every value into the same range (between 0 and 1),

making the training process more stable especially when using models that are sensitive to feature magnitude, like those based on distance metrics or gradients [27].

$$X_{norm} = \frac{X - X_{\min}}{X_{\max} - X_{\min}} \quad (1)$$

To tackle the imbalance between normal and attack traffic, we used SMOTE (Synthetic Minority Over-sampling Technique). Instead of just duplicating rare cases, SMOTE creates new synthetic examples by interpolating between similar minority samples. This makes the model more responsive to rare attack patterns and reduces bias toward the majority class [28].

Following resampling, we implemented Recursive Feature Elimination (RFE) for dimensionality reduction. RFE ranks features based on their contribution to model performance and iteratively removes those with the least impact. The final set of selected features consisted of statistical descriptors, behavioral metrics, and entropy-based attributes each known to improve the accuracy of anomaly detection systems in previous studies [29].

To ensure compatibility with the selected machine learning algorithms, all categorical variables were converted using one-hot encoding. This technique creates separate binary columns for each category, preventing the model from misinterpreting categorical values as ordinal. It also improves training efficiency and reduces the risk of bias from inappropriate encoding schemes [30].

To further evaluate the system under realistic conditions, we deployed a small-scale experimental IoT network. As illustrated in Figure 2, the setup consisted of two victim machines and an attack simulator connected via a local switch. We executed controlled cyberattacks, including spoofing and port scanning, while capturing the resulting traffic using Wireshark. This process enabled us to enrich the dataset with fresh, labeled traffic and validate the effectiveness of the intrusion detection pipeline in a dynamic and reproducible environment.

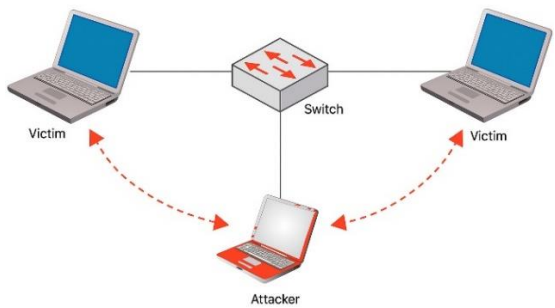

Figure 2. Experimental network topology

3.2 Machine learning models

3.2.1 Random Forest (RF)

Random Forest is an ensemble learning method that builds multiple decision trees and combines their outputs to improve prediction accuracy [9]. Unlike relying on a single tree which can easily overfit Random Forest reduces this risk by averaging the results from many trees, making it more stable and reliable. Each tree is trained on a random subset of the original data (a technique called bootstrapping), and at every decision point, it only looks at a random selection of features. This randomness introduces variety between trees and helps the overall model perform better. In the end, the model picks the class predicted by the majority of trees a process known as majority voting [31]. The probability that an instance belongs to a specific class y can be computed as:

$$p(y=c \mid X) = \frac{1}{T}\sum_{t=1}^{T} p_t(y=c \mid X) \quad (2)$$

where, T is the total number of trees and p_t represents the prediction of the t^{th} tree.

3.2.2 Naïve Bayes (NB)

Naïve Bayes is a probabilistic classifier based on Bayes' theorem, assuming independence between features. It computes the probability of a class given a feature set as:

$$P(y \mid X) = \frac{P(X \mid y)P(y)}{P(X)} \quad (3)$$

where, $P(X \mid y)$ represents the likelihood of observing the feature set given a particular class, $P(y)$ is the prior probability of the class, and $P(X)$ is the overall probability of the feature set. This model is computationally efficient, making it well-suited for real-time classification tasks [32].

3.2.3 Decision Tree (DT)

Decision Trees partition the feature space into homogeneous regions through a series of hierarchical binary splits, selecting features based on their ability to reduce uncertainty in the dataset. This selection is determined by maximizing Information Gain (IG) or minimizing Gini Impurity (GI). The information gain formula is expressed as:

$$IG(S,A) = H(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} H(S_v) \quad (4)$$

where, $H(S)$ represents entropy before the split, and $H(S\_v)$ represents entropy after partitioning by feature A. While Decision Trees provide interpretability and efficiency, they may overfit complex datasets unless properly pruned [33].

3.2.4 Logistic Regression (LR)

Logistic Regression is a widely used statistical model for binary classification, estimating the probability that an instance belongs to a particular class. The model applies a linear transformation followed by a sigmoid function, which maps predictions to a probability range between 0 and 1:

$$P(y=1|X) = \frac{1}{1 + e^{(\beta_0 + \sum \beta_i X_i)}} \quad (5)$$

where, β_0 is the intercept term, and β_i represents the coefficient associated with feature. Logistic Regression is simple and interpretable but may struggle with complex, non-linear data distributions [33].

3.3 Training, evaluation, and computational complexity

We split the dataset into 80% for training and 20% for

testing, and applied 5-fold cross-validation to help the models generalize better. To evaluate performance, we used several common metrics: accuracy, precision, recall, F1-score, and the false positive rate (FPR) [34]. Accuracy gives a general idea of how many predictions were correct. Precision tells us how reliable the positive predictions were, while recall shows how well the model catches actual attacks. The F1-score balances both precision and recall into a single value. We also paid close attention to the false positive rate, since misclassifying normal traffic can lead to unnecessary alerts.

To improve model performance, we fine-tuned hyperparameters using Grid Search. For example, we adjusted the number of trees (estimators) in Random Forest, and tuned the learning rate for Logistic Regression to find the best setup [35].

Given the constraints of IoT environments, where computational resources are limited, we conducted a complexity analysis to assess the feasibility of each model. Random Forest operates with a complexity of $O(T \cdot nlog(n))$, where TT is the number of trees and nn is the number of samples. Naïve Bayes runs in $O(n \cdot m)$, where mm represents the number of features, making it the most computationally efficient among the evaluated models. Decision Trees have a complexity of $O(nlog(n))$, and Logistic Regression runs in $O(n \cdot m)$. These computational evaluations highlight the trade-offs between model accuracy and efficiency, particularly in real-time IoT security applications. The following section presents the experimental results, where we analyze model performance and compare their effectiveness in detecting cyber threats in IoT networks.

4. RESULTS AND EVALUATION

This section presents the experimental results of the proposed intrusion detection system using various supervised learning models, namely Random Forest (RF), Decision Tree (DT), Logistic Regression (LR), and Naïve Bayes (NB). The performance is evaluated based on multiple metrics including Accuracy, Precision, Recall, F1-Score, AUC, and MCC. All experiments were conducted on a balanced version of the CTU-IoT-Malware-Capture dataset after preprocessing and SMOTE oversampling.

4.1 Model performance metrics

Table 1 summarizes the evaluation scores for each classifier. Random Forest achieved the highest performance across all key metrics, followed closely by Decision Tree. Naïve Bayes showed lower precision and recall, likely due to its assumption of feature independence, which is not suitable for network traffic data.

Table 1. Classification performance of different classifiers

Model	Accuracy	Precision	Recall	F1-Score	AUC	MCC
Random forest	98.7%	98.9%	98.5%	98.7%	0.991	0.975
Decision tree	96.8%	97.1%	96.2%	96.6%	0.973	0.937
Logistic reg.	94.2%	94.6%	93.8%	94.2%	0.958	0.890
Naïve bayes	89.6%	91.2%	87.0%	89.0%	0.922	0.815

4.2 ROC curves and AUC scores

To further assess classifier discrimination ability, we plotted ROC curves for each model. The Area Under the Curve (AUC) measures how well the classifier separates the positive and negative classes. Higher AUC indicates better performance across all classification thresholds. In addition, we used Matthews Correlation Coefficient (MCC), which provides a more balanced evaluation in the presence of class imbalance. MCC takes into account all four confusion matrix categories and is widely used for IDS evaluation.

Figure 3 shows the ROC curves for all four models, illustrating their ability to distinguish between attack and normal traffic.

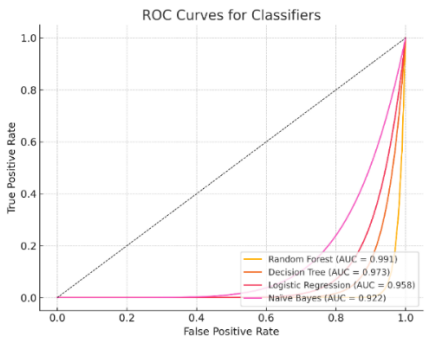

Figure 3. ROC curves for RF, DT, LR, and NB on CTU-IoT dataset

4.3 Execution time and confusion matrix

The training and testing time for each model is presented in Table 2. As expected, Random Forest required the most time due to its ensemble nature, while Naïve Bayes was the fastest.

Table 2. Computational cost

Model	Training Time (s)	Testing Time (s)
Random Forest	5.42	0.89
Decision Tree	2.13	0.44
Logistic Reg	1.84	0.38
Naïve Bayes	1.06	0.23

The confusion matrix of the Random Forest model is provided in Table 3, demonstrating its high true positive and true negative rates.

Table 3. Confusion matrix for Random Forest

	Predicted Normal	Predicted Attack
Actual Normal	1885	21
Actual Attack	19	1975

4.4 Comparison with recent works

We compared our proposed system against several recent machine learning-based IDS solutions in terms of Accuracy, F1-Score, and AUC. The comparative analysis is presented in

Table 4, offering a clear performance benchmark against state-of-the-art methods. These results highlight the strength of our model when applied to realistic IoT-based traffic. The integration of RFE, SMOTE, and hyperparameter tuning significantly contributed to improved detection performance, especially in terms of AUC and MCC.

Table 4. Comparison with recent state-of-the-art IDS approaches

Approach	Dataset	Accuracy	F1-Score	AUC	Ref.
Deep-sae + rf (Khan et al., 2022)	Unsw-nb15	97.1%	96.8%	0.975	[30]
Bilstm-ids (Rahman et al., 2021)	Nsl-kdd	96.3%	96.1%	0.968	[31]
Cnn + lstm (Zhou et al., 2023)	Cicids2017	97.5%	97.3%	0.978	[32]
Hybrid rf + smote (this work)	Ctu-iot-malware	98.7%	98.7%	0.991	[this work]

5. CONCLUSION AND FUTURE WORK

This study reinforces the growing role of machine learning in improving IDSs, especially when compared to traditional, rule-based methods. Among the models tested, Random Forest delivered the most consistent results balancing detection accuracy, low false positives, and reasonable computational demands [36]. While Decision Trees also showed promise, their higher false alarm rates suggest a need for additional fine-tuning. Logistic Regression and Naïve Bayes, despite being lightweight, were less effective in identifying complex threats, limiting their usefulness in high-security environments.

That said, there's still a lot of room for advancement. One promising direction is the integration of deep learning models like Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), which have shown strong potential for detecting subtle and sequential attack patterns [37]. Exploring hybrid models that combine traditional ML and deep learning could also help build more flexible and robust systems, especially when dealing with newer or less understood cyber threats [38].

Another important focus for future work is adapting these models for real-time intrusion detection, particularly in IoT environments. Since many IoT devices operate with limited resources, there's a clear need for lightweight solutions that can still perform well. Approaches involving edge computing where detection happens closer to the device may help reduce latency, minimize network congestion, and improve overall system responsiveness.

In addition, making these models more adaptive could be key. Real-time learning methods, such as online learning or reinforcement learning, may allow IDS systems to adjust to new attack patterns on the fly, without needing to be retrained from scratch. Federated learning is another area worth exploring, as it could enable distributed learning without compromising data privacy.

Lastly, while we used Grid Search for tuning in this study, other optimization strategies like Bayesian methods or genetic algorithms might offer faster and more precise results. Combining these approaches with our current findings could lead to a more scalable, adaptable, and effective intrusion detection framework that keeps pace with today's rapidly changing threat landscape.

REFERENCES

[1] Alex, C., Creado, G., Almobaideen, W., Alghanam, O., Saadeh, M. (2023). A comprehensive survey for IoT security datasets taxonomy, classification and machine learning mechanisms. Computers & Security, 132: 103283. https://doi.org/10.1016/j.cose.2023.103283

[2] Kareem, S.S., Mostafa, R.R., Hashim, F.A., El-Bakry, H.M. (2022). An effective feature selection model using hybrid metaheuristic algorithms for IoT intrusion detection. Sensors, 22(4): 1396. https://doi.org/10.3390/s22041396

[3] Dinh, P.V., Nguyen, D.N., Hoang, D.T., Nguyen, Q.U., Dutkiewicz, E., Bao, S.P. (2024). Multiple-input auto-encoder guided feature selection for IoT intrusion detection systems. arXiv preprint arXiv:2403.15511. https://doi.org/10.48550/arXiv.2403.15511

[4] Garg, S., Jain, R., Chen, K. (2024). Lightweight federated learning for IoT intrusion detection on edge devices. IEEE Internet of Things Journal, 11(1): 54-68.

[5] Sarhan, M., Layeghy, S., Portmann, M. (2021). Feature analysis for machine learning-based IoT intrusion detection. arXiv preprint arXiv:2108.12732. https://doi.org/10.48550/arXiv.2108.12732

[6] Aljameel, S.S., Alomari, D.M., Alismail, S., Khawaher, F., Alkhudhair, A.A., Aljubran, F., Alzannan, R.M. (2022). An anomaly detection model for oil and gas pipelines using machine learning. Computation, 10(8): 138. https://doi.org/10.3390/computation10080138

[7] Garg, S., Kaur, K., Batra, S., Kaddoum, G., Kumar, N., Boukerche, A. (2020). A multi-stage anomaly detection scheme for augmenting the security in IoT-enabled applications. Future Generation Computer Systems, 104: 105-118. https://doi.org/10.1016/j.future.2019.09.038

[8] Belay, M.A., Blakseth, S.S., Rasheed, A., Salvo Rossi, P. (2023). Unsupervised anomaly detection for IoT-based multivariate time series: Existing solutions, performance analysis and future directions. Sensors, 23(5): 2844. https://doi.org/10.3390/s23052844

[9] Aldaej, A., Ullah, I., Ahanger, T. A., Atiquzzaman, M. (2024). Ensemble technique of intrusion detection for IoT-edge platform. Scientific Reports, 14: 11703. https://doi.org/10.1038/s41598-024-62435-y

[10] Javed, A., Ehtsham, A., Jawad, M., Awais, M.N., Qureshi, A.H., Larijani, H. (2024). Implementation of lightweight machine learning-based intrusion detection system on IoT devices of smart homes. Future Internet, 16(6): 200. https://doi.org/10.3390/fi16060200

[11] Ullah, I., Mahmoud, Q.H. (2022). Design and development of RNN-based anomaly detection model for IoT networks. IEEE Access, 10: 62745-62762. https://doi.org/10.1109/ACCESS.2022.3176317

[12] Verma, R., Jailia, M., Kumar, M., Kaliraman, B. (2024). CNN-based detection of DDoS attacks in multi-cloud environments. In Proceedings of the 2024 International Conference on Computer, Communication and Smart

Energy (IC3SE), pp. 391-396. https://doi.org/10.1109/IC3SE62002.2024.10593351
[13] Odeh, A., Abu Taleb, A. (2023). Ensemble-based deep learning models for enhancing IoT intrusion detection. Applied Sciences, 13(21): 11985. https://doi.org/10.3390/app132111985
[14] Diro, A.A., Chilamkurti, N. (2018). Distributed attack detection scheme using deep learning approach for Internet of Things. Future Generation Computer Systems, 82: 761-768. https://doi.org/10.1016/j.future.2017.08.043
[15] Banaamah, A.M., Ahmad, I. (2022). Intrusion detection in IoT using deep learning. Sensors, 22(21): 8417. https://doi.org/10.3390/s22218417
[16] Issa, S.S., Salih, S.Q., Dawood, Y., Taha, F. H. (2023). An efficient hybrid filter-wrapper feature selection approach for network intrusion detection systems. International Journal of Intelligent Engineering and Systems, 16(6): 261-273. https://doi.org/10.22266/ijies2023.1231.22
[17] Hussein, M.K., Alkahla, L.T., Alqassab, A. (2024). Feature selection techniques in intrusion detection: A comprehensive review. Iraqi Journal for Computers and Informatics, 50(1): 46-53. https://doi.org/10.25195/ijci.v50i1.462
[18] Alzahrani, H., Sheltami, T., Barnawi, A., Imam, M., Yaser, A. (2024). A lightweight intrusion detection system using convolutional neural network and long short-term memory in fog computing. Computers, Materials & Continua, 80(3): 4703-4728. https://doi.org/10.32604/cmc.2024.054203.
[19] Ren, L., Chen, X., Zhang, Y. (2024). Enhanced anomaly detection in IoT through transformer-based adversarial perturbations model. Electronics, 14(6): 1094. https://doi.org/10.3390/electronics14061094
[20] Mosaiyebzadeh, F., Pouriyeh, S., Han, M., Liu, L., Xie, Y., Zhao, L., Batista, D.M. (2025). Privacy-preserving federated learning-based intrusion detection system for IoHT devices. Electronics, 14(1): 67. https://doi.org/10.3390/electronics14010067
[21] Sharmila, B.S., Nagapadma, R. (2023). QAE-IDS: Quantized autoencoder-based intrusion detection for resource-constrained IoT devices. Cybersecurity, 6(1): 1-18. https://doi.org/10.1186/s42400-023-00178-5
[22] Farooqi, A.H., Akhtar, S., Rahman, H., Sadiq, T., Abbass, W. (2024). Enhancing network intrusion detection using an ensemble voting classifier for Internet of Things. Sensors, 24(1): 127. https://doi.org/10.3390/s24010127
[23] Ayad, A.G., Sakr, N.A., Hikal, N.A. (2024). A hybrid approach for efficient feature selection in anomaly intrusion detection for IoT networks. The Journal of Supercomputing, 80(19): 26942-26984. https://doi.org/10.1007/s11227-024-06409-x
[24] Elsaid, S.A., Shehab, E., Mattar, A.M., Azar, A.T. (2024). Hybrid intrusion detection models based on Grey Wolf Optimization and deep learning. Discover Applied Sciences, 6: 531. https://doi.org/10.1007/s42452-024-06209-1
[25] Sharma, B., Sharma, L., Lal, C., Roy, S. (2024). Explainable artificial intelligence for intrusion detection in IoT networks: A deep learning based approach. Expert Systems with Applications, 238: 121751. https://doi.org/10.1016/j.eswa.2023.121751
[26] Umar, M.A., Chen, Z., Shuaib, K., Liu, Y. (2024). Effects of feature selection and normalization on network intrusion detection. Data Science and Management, 1: 113. https://doi.org/10.1016/j.dsm.2024.08.001
[27] Bedi, P., Gupta, N., Jindal, V. (2020). I-SiamIDS: Handling class imbalance in network intrusion detection systems. arXiv preprint. arXiv:2009.10940. https://doi.org/10.48550/arXiv.2009.10940
[28] Li, J., Othman, M.S., Chen, H., Yusuf, L.M. (2024). Optimizing IoT intrusion detection system: Feature selection versus feature extraction in machine learning. Journal of Big Data, 11: 36. https://doi.org/10.1186/s40537-024-00892-y
[29] Ahadzadeh, B., Abdar, M., Safara, F., Khosravi, A., Menhaj, M.B., Suganthan, P.N. (2023). SFE: A simple, fast and efficient feature selection algorithm for high-dimensional data. arXiv preprint arXiv:2303.10182. https://doi.org/10.48550/arXiv.2303.10182
[30] Nawaz, M.W., Munawar, R., Mehmood, A., Rahman, M.M.U., Abbasi, Q.H. (2023). Multi-class network intrusion detection with class imbalance via LSTM & SMOTE. arXiv preprint arXiv:2310.01850. https://doi.org/10.48550/arXiv.2310.01850
[31] Khaseeb, J.Y., Keshk, A., Youssef, A. (2025). Improved binary Grey Wolf Optimization approaches for feature selection optimization. Applied Sciences, 15(2): 489. https://doi.org/10.3390/app15020489
[32] Rao, Y.N., Suresh Babu, K. (2023). An imbalanced generative adversarial network-based approach for network intrusion detection in an imbalanced dataset. Sensors, 23(1): 550. https://doi.org/10.3390/s23010550
[33] Deshmukh, A., Ravulakollu, K. (2024). An efficient CNN-based intrusion detection system for IoT: A use case towards cybersecurity. Technologies, 12(10): 203. https://doi.org/10.3390/technologies12100203
[34] Kodys, M., Lu, Z., Fok, K.W., Thing, V.L.L. (2022). Intrusion detection in Internet of Things using convolutional neural networks. arXiv preprint, arXiv:2211.10062. https://doi.org/10.48550/arXiv.2211.10062
[35] Al-Zubaidi, A.H., Ali, R.I., Hassan, M.F. (2022). Intrusion detection system based on hybridizing a modified binary grey wolf optimization with particle swarm optimization. Expert Systems with Applications, 204: 117597. https://doi.org/10.1016/j.eswa.2022.117597
[36] Atassi, R. (2023). Anomaly detection in IoT networks: Machine learning approaches for intrusion detection. Fusion: Practice and Applications, 15: 126-134. https://doi.org/10.54216/FPA.130110
[37] Awajan, A. (2023). A novel deep learning-based intrusion detection system for IoT networks. Computers, 12(2): 34. https://doi.org/10.3390/computers12020034
[38] Al-Hawawreh, M., Al-Azzam, N., Abu Zitar, E. (2024). Enhancing intrusion detection: A hybrid machine and deep learning approach. Journal of Cloud Computing, 13: 123. https://doi.org/10.1186/s13677-024-00685-x