




Synthetic Minority Over-sampling Technique for detecting Malicious Traffic targeting Internet of Things' devices

Jaqueline Damacena Duarte  [University of Brasilia | jaqueline.duarte@aluno.unb.br]


Guilherme Dantas Bispo  [University of Brasilia | guilherme.bispo@redes.unb.br]

Gabriel Arquelau Pimenta Rodrigues   [University of Brasilia | gabriel.arquelau@redes.unb.br]

André Luiz Marques Serrano  [University of Brasilia | andrelms@unb.br]

Gabriela Mayumi Saiki  [University of Brasilia | gabriela.saiki@redes.unb.br]

Vinícius Pereira Gonçalves  [University of Brasilia | vpgvinicius@unb.br]

 Department of Electrical Engineering, Faculty of Technology, University of Brasília (UnB), Brasília, DF, 70910-900.

Received: 30 November 2024 • **Accepted:** 12 May 2025 • **Published:** 19 June 2025

Abstract This study proposes a multiclass machine learning approach for detecting 34 distinct types of cyberattacks in Internet of Things (IoT) traffic using the CICIoT2023 dataset. We evaluate the performance of lightweight classifiers—Bernoulli Naive Bayes, Decision Tree, Random Forest, and XGBoost—under highly imbalanced conditions. To address class imbalance and improve minority-class detection, we apply the Synthetic Minority Over-sampling Technique (SMOTE). In addition, we conduct hyperparameter tuning using RandomizedSearchCV and assess model performance using macro-average metrics, including recall, precision, and F1-score. Experimental results demonstrate that XGBoost and Random Forest, when optimized and combined with SMOTE, consistently achieves high and balanced detection rates across all classes. These findings suggest its applicability to real-world IoT intrusion detection scenarios, particularly in resource-constrained environments.

Keywords: IoT, machine learning, malicious traffic, over-sampling

1 Introduction

The advent of Industry 4.0, characterized by intelligent automation and pervasive connectivity, has catalyzed a profound digital transformation across various sectors. Central to this transformation is the Internet of Things (IoT), which interconnects a vast array of sensors, actuators, and devices. These technologies have revolutionized industries by enabling real-time data collection, analysis, and decision-making processes, leading to unprecedented operational efficiencies. However, this increased connectivity has also introduced new and complex cybersecurity challenges, that may lead, for example, to data leaks [Pimenta Rodrigues *et al.*, 2024]. Data protection frameworks may be poorly structured to regulate the secure collection and processing of user data from IoT devices [Pimenta Rodrigues *et al.*, 2024]. As these devices are more broadly adopted, the volume of network traffic they generate expands correspondingly, creating a multitude of potential attack vectors for malicious actors. The heterogeneity of IoT devices, along with their often limited computational resources and the use of legacy systems, further exacerbates the security vulnerabilities in these networks [Pedreira *et al.*, 2021; Hou *et al.*, 2019].

In the context of IoT security, credential-related attacks remain relevant. A recent analysis of the RockYou2024 leak [Rodrigues *et al.*, 2025] revealed ongoing vulnerabilities in password usage patterns, especially within industrial systems, emphasizing the need for enhanced anomaly detection mechanisms. The critical nature of these security challenges is emphasized by several high-profile cyberattacks on IoT devices. These incidents may result in substantial finan-

cial losses and significant operational disruptions across industries, highlighting the consequences of inadequate IoT security. For instance, attacks such as the Mirai botnet, which exploited poorly secured IoT devices, demonstrated the potential for massive Distributed Denial-of-Service (DDoS) attacks that can cripple internet infrastructure globally [Sharma *et al.*, 2023]. As the adoption of IoT continues to accelerate, so does the urgency to address the inherent vulnerabilities of these devices [Noman and Abu-Sharkh, 2023]. The growing reliance on interconnected systems makes it imperative to develop and implement robust security measures that can effectively mitigate these risks. This includes securing the devices and protecting the network infrastructure and ensuring the integrity of the data transmitted across these networks.

In response to these escalating threats, Machine Learning (ML) algorithms have emerged as powerful tools for detecting anomalies within the vast and complex datasets generated by IoT devices. Traditional security measures, such as signature-based detection, are increasingly inadequate in the face of zero-days exploitations and sophisticated attack methods. Conversely, ML models can autonomously learn the patterns from data, enabling them to detect previously unseen threats. By analyzing vast amounts of data in real-time, these models can identify subtle deviations from normal behavior that may indicate malicious activity, significantly reducing the response time to potential threats. Moreover, ML algorithms can adapt to evolving threats, providing a dynamic defense mechanism that improves over time [Bispo *et al.*, 2024]. They can also enhance the security of sensitive data and information [Vergara *et al.*, 2024].

Building on the strengths of ML, this research focuses on

the application of these models to identify malicious traffic specifically targeting IoT devices. We examine a comprehensive range of attack types, including Denial of Service (DoS), DDoS, Botnet, Reconnaissance (Recon), Spoofing, and Web application attacks. The success of this approach hinges on the availability of high-quality, labeled datasets that accurately reflect real-world attack scenarios. Thus, we utilize a comprehensive dataset of real-time network traffic meticulously curated by the Canadian Institute for Cybersecurity (CIC). This dataset is specifically designed to analyze large-scale attacks in IoT environments, making it an invaluable resource for both researchers and practitioners. It allows for the testing and validation of ML models in a realistic and challenging environment, ensuring that the developed solutions are robust and effective in practical applications [Neto et al., 2023].

The contributions of this paper are threefold. First, we provide a detailed analysis of the current state-of-the-art in IoT security, identifying the strengths and limitations of existing approaches. Second, we develop a multi-class classifier capable of distinguishing between different types of attacks with high accuracy. Third, we offer a comprehensive evaluation of various machine learning models, including Bernoulli Naive Bayes (BNB), Decision Tree Classifier (DTC), and Random Forest Classifier (RFC), using accuracy and recall metrics to determine the most effective model. The results of our study have significant implications for the design of future IoT security systems, potentially leading to more resilient and adaptive defense mechanisms.

Despite the increasing number of studies applying machine learning to IoT traffic analysis, many existing approaches are limited in scope. Several works focus on binary classification tasks (e.g., attack vs. benign), which do not reflect the complexity of real-world IoT environments. Others restrict their analysis to a small subset of attack types or fail to address the severe class imbalance present in these datasets, leading to biased or incomplete detection. Moreover, the evaluation of such models often relies on metrics like overall accuracy, which may obscure poor performance on minority classes. In contrast, this study proposes a multiclass classification approach capable of identifying 34 distinct attack types using lightweight and interpretable models. We also incorporate strategies to handle class imbalance (SMOTE and undersampling) and adopt macro-average performance metrics to ensure balanced evaluation across all classes. These contributions aim to improve the practical applicability of intrusion detection systems in heterogeneous and resource-constrained IoT environments.

The structure of this paper is organized as follows: Section 2 provides a review of related work, discussing existing approaches and identifying ongoing challenges in the field of IoT security. Section 3 presents the database and category classifier, detailing the machine learning models employed for training and classification. Section 4 describes the materials and methods used to develop a multi-class classifier, with a particular focus on accuracy and recall metrics to select the most effective models, namely Bernoulli Naive Bayes, Decision Tree Classifier, and Random Forest Classifier. In Section 5, we present and analyze the results, highlighting the performance of each model across different stages. Ul-

timately, Section 6 concludes the paper by summarizing the key findings and discussing potential directions for future research.

2 Related works

This section reviews relevant cybersecurity works, with a specific focus on the challenges and advancements in securing the Internet of Things. We explore the application of machine learning for malicious traffic detection. We will discuss relevant research exploring data feature reduction and data augmentation methods used in the machine learning context from 2021-2025.

2.1 Feature reduction

Many researches on feature reduction techniques are employed on the aim to identify and retain only the most informative features, and they have been largely recommended as they work on simplifying the dataset and have significantly enhanced model accuracy, efficiency, and interpretability, mitigating the risk of overfitting, where a model learns irrelevant patterns in the data without compromising its interpretability. Feature selection and feature extraction are two common techniques often used in combination or interchangeably in Network Intrusion Detection Systems (NIDS). Feature selection identifies and retains a subset of the original features, while feature extraction transforms these features into a lower-dimensional representation. Both techniques aim to enhance the efficiency and accuracy of NIDS models by reducing noise, redundancy, and computational complexity [Ji et al., 2024; Li et al., 2024].

Many approaches to feature selection, like the Filtering method, Exhaustive Search, Random Search, and Manual selection. Filtering methods evaluate the relevance between features regardless of the type of machine learning algorithm or deep learning algorithm and select useful features based on importance. Hou et al. [2019] and Amaouche et al. [2023] use mutual information (MI), which are representative filtering methods, to evaluate features and select features with high relevance for learning and anomaly detection.

Exhaustive Search is a method that systematically examines every possible combination of features to find the best subset for a given machine learning task. Starting with the entire set of features, it gradually explores smaller subsets, including combinations with a single feature up to the full set. Another study investigated the challenging task of selecting the optimal subset of features from a large pool, which is a computationally complex problem. Jeon et al. [2024] proposed a suboptimal feature selection method that combines permutation importance and exhaustive search. While this novel approach still requires further research, the result demonstrated that it can offer a less complex alternative to traditional methods like greedy and random search. Despite its suboptimal nature, the results produced by this method are comparable to those obtained using the more computationally intensive traditional approaches.

Manual selection is an approach to feature selection that relies on an expert's intuition, understanding of the domain,

and the specific requirements of the problem. Numerous studies have demonstrated the effectiveness of manually removing irrelevant features from encrypted traffic to enhance anomaly detection. This feature selection method has also been employed in data standardization processes to prepare data for use in detection models or algorithms [Ji *et al.*, 2024].

Recent work relates a lack of research on feature selection and feature extraction approaches applied to malicious traffic detection. To address this, a comprehensive comparison was conducted between feature selection, based on the Pearson Correlation Coefficient, and feature extraction, based on Principal Component Analysis. The evaluation was performed on both binary and multi-class classification tasks using various machine learning algorithms, including Decision Tree, Random Forest, k-Nearest Neighbors, Naive Bayes, and Multi-Layer Perceptron. The analysis utilized a well-known heterogeneous dataset collected from telemetry data of IoT and IIoT sensors as well as operating system datasets from Windows and Unix-based environments. The study related the performance of models trained with Feature Extraction surpasses that of ones trained on Feature Selection when considering a small number of features, highlighting a possible approach in contribution for lightweight models [Li *et al.*, 2024].

A Feature Entropy Estimation (FEE) method is also proposed to identify the most relevant features. FEE combines rank-based chi-square, Pearson correlation, and f-score correlation to extract significant features from the dataset. Subsequently, feature entropy estimation is applied to validate the relationships among these extracted features and select the most informative ones using the information gain criterion. An Extreme Gradient Boosting (XGBoost) classification model was tested on three different datasets, demonstrating superior accuracy compared to previous studies on the same databases [Diwan *et al.*, 2021]. Awad *et al.* [2022] used three-based feature importance to assess the significance of individual features in representing attack class patterns and evaluate model performance on binary and multi-class classifiers in different importance thresholds for select the best feature set on detecting malicious traffic using decision three [Awad *et al.*, 2022].

Some studies applied Least Absolute Shrinkage and Selection Operator (LASSO) for feature selection, which is a machine learning algorithm used for feature selection and regularization. It's a regression analysis technique that selects variables and applies regularization to improve the statistical model's interpretation and prediction accuracy. LASSO was related to identifying the most important features in the dataset that contribute to the prediction of the target variable, reducing overfitting, multicollinearity, interpretability, and handling high-dimensional data [Dasari and Kaluri, 2024; Arshad *et al.*, 2023].

The studies included in Table 1 were selected based on their relevance and impact in the field, considering factors such as publication venue, citation count, and recency. This approach follows established literature review practices, as recommended by [Bispo *et al.*, 2024], and ensures that our comparison is made against representative and influential works.

2.2 Data sampling

Network intrusion detection may encounter imbalanced datasets where the prevalence of normal instances far exceeds that of rare intrusions. This imbalance can introduce biases and limit the effectiveness of machine learning models. To mitigate these challenges, some approaches, such as class weighting or under-sampling the majority class, can be applied to the training set for balancing the dataset and improving model performance. It is worth noting that, as under-sampling reduces the majority class to match the size of the minority class, it can lead to data loss when insufficient data is remaining for effective model training. Another alternative is oversampling techniques, that can be achieved, for example, with data augmentation, which is a common method in computer vision. It generates additional training data by modifying existing examples, expanding the dataset, and improving model generalization. Oversampling increases the number of minority class samples, reducing model variance and addressing class imbalance. However, excessive oversampling can introduce noise, leading to synthetic samples that may not accurately represent the true distribution. For this reason, oversampling is recommended to be used when under-sampling is not suitable for achieving good performance Chalé and Bastian [2022]. On the proposition of a methodology to evaluate generated data for Network Intrusion Detection System, Chalé and Bastian [2022] developed a study using Conditional Tabular Generative Adversarial Network (CTGAN) and Tabular Variational Autoencoder (TVAE) comparing model performance to real data training and proposes a minimal percentage of data generated and real data for optimal result. Some recent studies also evaluate the Synthetic Minority Over-sampling Technique. Musleh *et al.* [2023] applies SMOTE to balance a dataset while experimenting with several feature extraction methods. Zhang and Liu [2022] proposes an improved Conditional Variational Autoencoder (CVAE) combined with the Borderline Synthetic Minority Oversampling Technique (BSM) to address class imbalances on an IoT intrusion detection based on machine learning. Attou *et al.* [2023] employed a hybrid resampling approach by oversampling the minority class, using SMOTE combining to under-sampling the majority class and additionally applying coast-sensitive to minority classes.

As summarized in Table 2, several methods have been proposed to address class imbalance in network intrusion detection, especially within IoT environments. Techniques like SMOTE and Conditional Variational Autoencoder have been employed to enhance model performance and bolster security.

2.3 Hyperparameter optimizing

Machine learning models are characterized by their internal structure and learning capacity, which are largely determined by a set of parameters known as hyperparameters. These hyperparameters are predefined by the model designer prior to the training process and significantly influence the behavior of the learning algorithm. Each ML model has a unique set of hyperparameters, and the optimal configuration is heavily dependent on the specific characteristics of the training data,

Table 1. Summary of recent studies on feature selection and reduction techniques in IoT security.

| Study | Methodology/Technique | Contribution to IoT Security |
|----------------------------------|---|---|
| [Ji <i>et al.</i> , 2024] | Systematic review of AI-based anomaly detection over encrypted traffic, focusing on data feature reduction, pre-processing, and performance indicators. | Analyzed the most relevant feature reduction and selection methods. |
| [Li <i>et al.</i> , 2024] | Comparison of feature reduction techniques for ML-based intrusion detection in IoT networks. | Proposed feature reduction for lightweight models. |
| [Amaouche <i>et al.</i> , 2023] | Method to detect malicious traffic in Vehicular Ad Hoc Networks (VANETs). | Applied mutual information filtering and class balancing with SMOTE. |
| [Jeon <i>et al.</i> , 2024] | Feature selection approach using permutation feature importance and exhaustive search. | Proposed a suboptimal feature selection model (SFSM) for detecting malicious behavior in resource-limited IoT environments. |
| [Diwan <i>et al.</i> , 2021] | Feature reduction approach based on FEE. | Used FEE for feature extraction combined with information gain for feature selection. |
| [Awad <i>et al.</i> , 2022] | Analysis of NetFlow features for detecting IoT network intrusions. | Utilized a tree-based feature importance model for feature selection and analyzed performance at different thresholds of importance measures. |
| [Arshad <i>et al.</i> , 2023] | Novel ensemble method for securing IoT devices against botnet attacks. | Applied LASSO for feature selection. |
| [Dasari and Kaluri, 2024] | Classification of DDoS attacks in a distributed network using hierarchical ML. | Applied LASSO for feature selection. |
| [Chaudhary <i>et al.</i> , 2025] | Entropy-based anomaly detection combined with attribute selection via Symmetrical Uncertainty and K-means clustering. | Proposed a multi-phase DDoS detection and mitigation framework for SDN-enabled fog-based IoT using hybrid statistical and ML-based models. Attribute selection improved multiclass classification accuracy and reduced false positives. |
| [Mittal <i>et al.</i> , 2025] | Deep learning ensemble using CNN, LSTM, ANN, with class balancing via SMOTE and NearMiss. Explainable AI applied using LIME. | Proposed the DLEX-IMD framework for IoT malware detection, achieving near-perfect accuracy. Feature relevance interpreted through LIME improves trust in automated decisions and provides transparency for deployment in real IoT contexts. |
| Current Work | Synthetic Minority Oversampling Technique (SMOTE) for multi-class classifier on malicious traffic detection in IoT traffic. | Utilized a tree-based feature importance model and compared model performance with and without balancing via class weights and oversampling. |

Table 2. Summary of data sampling techniques in IoT intrusion detection.

| Study | Methodology/Technique | Contribution to IoT Security |
|---------------------------------|--|--|
| [Chalé and Bastian, 2022] | Methodology to evaluate generated data for Network Intrusion Detection Systems using Machine Learning. | Evaluated the performance of CTGAN and TVAE compared to real data training. Discussed a minimum percentage of generated and real data for optimal results. |
| [Musleh <i>et al.</i> , 2023] | Machine learning approach using image format for intrusion detection. | Applied SMOTE technique for class balancing. |
| [Zhang and Liu, 2022] | Data augmentation techniques for IoT intrusion detection. | Proposed addressing dataset imbalance using CVAE combined with BSM. |
| [Attou <i>et al.</i> , 2023] | Combined data balancing techniques for IoT intrusion detection. | Proposed oversampling and under-sampling techniques combined with a cost-sensitive model. |
| [Algabroun and Håkansson, 2025] | Parametric machine learning optimization for adaptive sampling (DSRA-PMLO), based on signal dynamics and error threshold tuning. | Proposed an energy-efficient sampling strategy using machine learning to dynamically control the sampling rate based on real-time signal variations, reducing data volume without violating accuracy constraints. Suitable for IoT environments with limited energy and bandwidth. |
| Current Work | SMOTE for malicious traffic detection in IoT, focused on optimal macro-average recall. | Compared the performance of ML models on imbalanced and balanced datasets using SMOTE and class weights to achieve the best macro-average recall. |

including its distribution, size, format, and features. Dasari and Kaluri [2024].

Optimizing hyperparameters to achieve optimal model performance is an important task in machine learning, contributing to improving model performance, avoiding overfitting, and maximizing resource utilization. This task involves systematically exploring and evaluating different combinations of hyperparameter values to identify the configuration that maximizes model performance according to specific metrics. Generally, the optimal set can be achieved by an exhaustive search of all possible combinations of parameter sets, but often, it's not possible due to limited resources [Abdelmoumin *et al.*, 2022; Dasari and Kaluri, 2024]. Liu *et al.* [2021] proceed a literature review involving deep learning and machine learning researches on the identification and detection of Internet of Things devices, this study listed several techniques that have been proposed to address the hyperparameter optimizing, such as grid search, random search, prediction-based approaches, and evolutionary algorithms.

Grid search is an approach to hyperparameter optimization, involving a comprehensive exploration of the parameter space at regular intervals, basically, it uses a pre-defined grid to try different configurations or settings and identify the best model. However, this method can be computationally inefficient due to its exhaustive search nature, which can evaluate many suboptimal parameter combinations. As an alternative, random search conserves more system resources by randomly sampling points within the parameter space. This approach introduces greater diversity and can outperform grid search when only a small subset of hyperparameters significantly impacts model performance [Liu *et al.*, 2021]. Both grid search and random search have been widely used. Whilst grid search is more suited to use when the best parameter settings are unknown, random search is more suitable to use when increasing performance based on user-defined metrics is the goal. Abdelmoumin *et al.* [2022]

In Abdelmoumin *et al.* [2022], an anomaly-based intrusion detection system for the Internet of Things develops Deep learning models and explores random search methods for searching best hyperparameters to achieve the best performance on AUC, F1-Score, and accuracy.

Some studies have also been developed utilizing automated machine learning tools, like Auto AI and Auto ML, that has their own mechanism of hyperparameter tuning. In Behnke *et al.* [2021] developed malicious activity detection in the DNSOver-HTTPs Protocol using machine learning utilizing Auto AI, which performs hyperparameter optimization by executing a variety of hyperparameter tuning options and then ranks based on metrics such as accuracy and precision. Dasari and Kaluri [2024] developed a classification of DDoS Attacks in a distributed network using Hierarchical machine learning and utilizes AutoML CatBoost for hyperparameters optimization.

Prediction-based methods, such as Bayesian optimization, employ a search strategy that initially conducts random trials to build a model of the relationship between hyperparameters and model performance. Subsequently, they focus on exploring regions of the parameter space that are more likely to yield improved results. An example of a prediction-based method is the Bayesian optimization Liu *et al.* [2021].

Another approach is the evolutionary algorithms, which are inspired by biological evolution and apply heuristic search techniques to optimize hyperparameters. These algorithms simulate processes like selection, mutation, and crossover to identify promising parameter configurations [Liu *et al.*, 2021]. Thaseen *et al.* [2021] utilizes an evolutionary algorithm, Artificial Immune Network for hyperparameter tuning on machine learning anomaly detection for IoT.

A recent study, [Arshad *et al.*, 2023], utilizes manual fine-tuning to research a suitable set of hyperparameters to identify botnet attacks on IoT using an ensemble technique involving K-neighbors, Decision trees, and Random forests.

Despite numerous studies addressing feature selection and class imbalance in IoT security, few works provide a comparative evaluation of lightweight ML classifiers on large-scale multi-class IoT traffic using oversampling techniques like SMOTE concerning macro-average metrics. This gap underscores the need for scalable, accurate, and interpretable models, especially those evaluated with suitable metrics—such as macro-average recall, precision, and F1-score—which provide a fair assessment of model performance across all classes, including underrepresented ones that are often overlooked by traditional accuracy measures.

In summary, although prior works have applied machine learning to detect IoT-related threats, most of them focus on binary classification or a reduced set of attack categories. Additionally, many studies overlook the effects of strong class imbalance or rely on complex models with limited applicability in resource-constrained IoT environments. These gaps highlight the need for lightweight, interpretable, and multiclass-capable models that can operate effectively under realistic conditions. This study addresses these challenges by proposing a practical solution tailored to the detection of 34 different attack types in highly imbalanced IoT traffic.

3 Materials and methods

In this section, information about the datasets used in the study is provided. Afterwards, the data preparation and feature selection, modeling and training steps are explained. Then, the evaluation metrics and hyperparameter selection steps.

Figure 1 illustrates the steps involved in developing the most suitable model for this task.

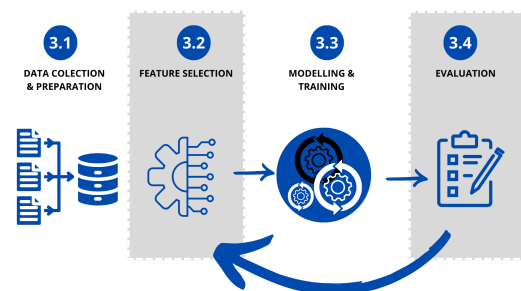


Figure 1. Diagram for the research methodology

Stage 1, the dataset preparation, includes data pre-

Table 3. Summary of hyperparameter optimization techniques in IoT security.

| Study | Methodology/Technique | Contribution to IoT Security |
|------------------------------------|--|--|
| [Abdelmoumin <i>et al.</i> , 2022] | Intrusion detection for the Internet of Things. | Used a random search algorithm for hyperparameter tuning. |
| [Liu <i>et al.</i> , 2021] | Literature review of machine learning for detecting and identifying IoT devices. | Discussed ML applications for IoT intrusion detection, including hyperparameter search strategies. |
| [Behnke <i>et al.</i> , 2021] | Malicious activity detection in the DNS-over-HTTPS protocol using ML. | Used AutoAI for automated hyperparameter optimization. |
| [Thaseen <i>et al.</i> , 2021] | ML-based anomaly detection in IoT using a Hadoop framework. | Utilized an artificial immune network for hyperparameter tuning. |
| [Arshad <i>et al.</i> , 2023] | Novel ensemble method for enhancing IoT device security against botnet attacks. | Performed manual fine-tuning of hyperparameters. |
| [Dasari and Kaluri, 2024] | Classification of DDoS attacks in a distributed network using hierarchical ML. | Applied AutoML with CatBoost for hyperparameter optimization. |
| Current Work | SMOTE-based malicious traffic detection in IoT, focusing on maximizing macro-average recall. | Used Randomized Search Cross-Validation for hyperparameter optimization. |

processing and cleaning. In Stage 2, a feature importance score was applied to enhance feature selection before randomly splitting into a data training set (80%) and an evaluation set (20%). In Stage 3, the modeling and training stage, the training dataset is balanced using the SMOTE technique and subsequently used to train Decision Tree, Random Forest, and Naive Bayes models. Finally, the evaluation dataset is used to test and evaluate the models based on accuracy and recall metrics. Each of these stages is explained in the next sections.

3.1 Dataset preparation

This research utilizes the CICIOT2023 dataset, a comprehensive collection of real-time IoT attack data curated by the Canadian Institute for Cybersecurity (CIC). This dataset is accessible after completing a formal request at the CIC official website. Collected using Wireshark within the CIC's IoT Lab and exported in PCAP format, the dataset has proven valuable for malware and malicious traffic detection research due to its realistic portrayal of IoT attacks. It employs a complex topology involving various real IoT devices, with some acting as attackers and others as victims.

The raw dataset consists of 46,686,579 traffic lines exchanged between 105 devices, categorized into benign activity and seven attack types: DDoS, Brute Force, Spoofing, DoS, Recon, Web-based, and Mirai [Neto *et al.*, 2023]. Originally composed of 46 numerical, 21 of them categorical variables, distributed in 34 distinct labels, the dataset underwent pre-processing. All data is converted into parquet format for efficient reading, with categorical data transformed into small integers and continuous data into int32 format to optimize memory usage without compromising data quality.

3.2 Feature selection

This study uses an embedded three-based feature selection. In this method, the feature selection occurs whilst the classifier is being trained, leading to both a chosen set of features and a trained classifier. These methods are often faster and

less prone to overfitting, as they do not require separate feature selection steps [Büyükeçeci and Okur, 2022].

In this method all 46 features were ranked by the given feature importance and a threshold is defined to reduce the feature set leading to the most explainable and efficient model classification.

3.3 Modeling and training

At this stage, the training dataset may exhibit class imbalance, where certain classes have significantly fewer samples than others, leading to biased models. To address this issue in the training dataset, several techniques can be employed. Class weighting, under-sampling, and oversampling are commonly used methods. Class weighting assigns higher weights to minority class samples during training, allowing the model to focus on underrepresented classes. Under-sampling reduces the majority class to match the minority class size, although this can potentially result in the loss of valuable information. Oversampling techniques, such as data augmentation, increase the number of minority class samples and are useful to reduce model variance, helping to balance the dataset and improve model performance. However, it is important to note that data augmentation can also introduce noise into the data if used excessively. Overly aggressive oversampling can lead to the creation of synthetic samples that may not be representative of the true underlying distribution. Therefore, it is essential to carefully consider the parameters and hyperparameters of SMOTE when applying it to a dataset, and the sample size should be used in cases where under-sampling is not suitable for good performance [Chalé and Bastian, 2022; Abdul Samad *et al.*, 2023].

Our research experimented with class weights, undersampling and a combination of oversampling techniques based on the Synthetic Minority Over-sampling Technique with undersampling. SMOTE, as represented in 2, operates on an imbalanced dataset by generating synthetic samples for the minority class based on existing instances and their nearest neighbors. This process involves randomly selecting a minority class instance, identifying its k-nearest neighbors within the same class, and creating new synthetic samples

along the line segments connecting the selected instance to its neighbors [Zhang and Liu, 2022].

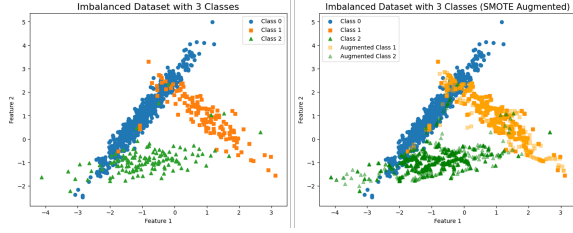


Figure 2. Smote augmentation technique illustration

After balancing the dataset, Bernoulli Naive Bayes, Decision Tree Random Forest, and XGBoost classifiers were trained and compared. These models were evaluated on both the original imbalanced dataset and the balanced dataset created using weighting and sampling techniques. The goal was to determine the most suitable classifier for this task.

Bernoulli Naive Bayes is a probabilistic-based algorithms based on Bayes Yalçın *et al.* [2024]. Naive Bayes Theorem is given by Equation 1.

$$[P(c/x) = \frac{P(c) * P(x/c)}{P(x)}] \quad (1)$$

Where

- $P(c)$: Is the class probability
- $P(x/c)$: Is the probability of the features given a class
- $P(x)$: Is the predictor probability
- $P(c/x)$: Is the posterior probability

Naive Bayes based models simplify complex models by the assumption that a given features are independent of each other. BNB variation assumes a Bernoulli data distribution, it's means it is suited for problems where features represent binary data.

Decision Tree are machine learning models that simplify complex problems by creating a hierarchical structure resembling a tree. Each node in the tree represents a decision based on a specific feature of the data, leading to different branches depending on the outcome. This structure makes DTs highly interpretable, revealing the decision-making process behind each prediction. DTs are versatile and can handle both binary classification and multi-class classification [Coscia *et al.*, 2024].

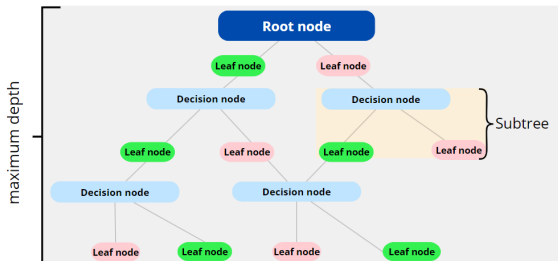


Figure 3. Decision tree illustration

Figure 3 illustrates a decision tree model. The tree starts with a root node representing the entire dataset. Internal nodes, called decision nodes, split the data into subsets based

on conditions related to specific features. Terminal nodes, or leaf nodes, represent the predicted classes. The maximum depth of the tree determines its complexity, and carefully tuning parameters like maximum depth, minimum samples for split, and minimum samples for leaf is essential to prevent overfitting and ensure a generalizable model.

The Random Forest algorithm has garnered significant interest in the academic community due to its ability to deliver both speed and accuracy in categorization tasks. Within the realm of predictive modeling and machine learning, RF is a model well-suited to supervised learning procedures, capable of handling both regression and classification problems. As illustrated in Figure 4, its approach leverages an ensemble of multiple decision trees. By aggregating the predictions from these individual trees, RF applies a majority vote in classification to achieve a more accurate and reliable prediction [Rimal *et al.*, 2024].

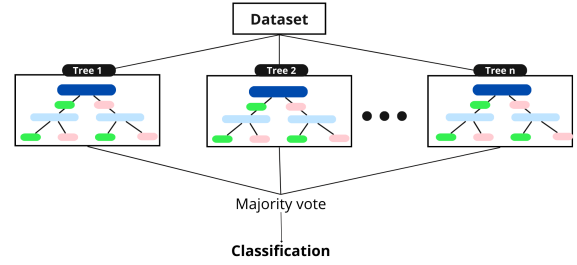


Figure 4. Random Forest model illustration

The Random Forest algorithm is characterized by some key parameters that influence its performance. These parameters, inherited from Decision Trees, include the maximum depth, minimum samples per leaf, and minimum samples per split. Additionally, a specific parameter, the number of estimators, determines the number of decision trees within the ensemble.

Extreme Gradient Boosting (XGBoost) is a machine learning algorithm based on the gradient boosting approach. It builds a sequence of models—typically decision trees—where each new tree corrects the errors made by the previous ones using gradient descent. Initially, all features are treated equally and passed to the trees. When some instances are misclassified, their weights are increased so that the next tree gives them more attention. This process helps the model improve over time Dasari and Kaluri [2024]. XGBoost optimizes a cost function that includes a differentiable loss term $b(\cdot)$ and a regularization term $\beta(\cdot)$, as shown in:

$$\Omega(\theta) = \sum_{m=1}^n b(x_i, \hat{x}_i) + \sum_{j=1}^J \beta(p_j) \quad (2)$$

- \hat{x}_m : are the predicted values
- n : is the number of training instances,
- J : is the total number of trees in the ensemble, and
- f_k : denotes the k -th tree.

For regularization, XGBoost typically uses a combined regularization function that includes both L1 regularization

(Lasso), which penalizes the absolute values of the leaf weights and L2 regularization (Ridge), which penalizes the squared values of the leaf weights Dasari and Kaluri [2024].

$$\beta(f_u) = \gamma L + \frac{1}{2} \left[\alpha \sum_{i=1}^L |c_i| + \lambda \sum_{i=1}^L c_i^2 \right] \quad (3)$$

Where, γ is the minimum loss reduction required to split a leaf, λ controls the penalty on leaf weights, and α penalizes the absolute values of the weights c_i .

3.4 Evaluation and hyperparameter tuning

At this stage, Randomized Search Cross-Validation (RandomizedSearchCV) was applied to get the best model's hyperparameter based on recall. This technique improves the model performance and avoids overfitting the data. Overfitting happens when the model is too complex and captures noise or random fluctuations in the training data rather than the underlying pattern. As a result, it performs well on training data but poorly on new, unseen data. RandomizedSearchCV uses cross-validation to find good hyperparameter settings by sampling a fixed number from defined distributions instead of evaluating all possible parameter combinations, reaching good results with efficient usage of memory and processing [Rimal et al., 2024].

To improve model performance and avoid overfitting, we employed RandomizedSearchCV to perform hyperparameter optimization for each classifier. For the Decision Tree and Random Forest models, the main hyperparameters tuned included `max_depth`, `min_samples_split`, and `min_samples_leaf`, which control the complexity of the trees and help prevent overfitting. Additionally, we explored the `criterion` parameter (Gini vs. entropy) and `max_features` for Random Forest, and tested depths ranging from 10 to 30, splits between 2 and 10, and leaf sizes between 1 and 4. For Random Forest, `n_estimators` was varied between 50 and 300.

The XGBoost classifier was optimized by adjusting `n_estimators` (50 to 500), `max_depth` (4 to 10), `learning_rate` (0.01 to 0.6), `subsample` (0.3 to 0.9), `colsample_bytree` (0.5 to 0.9), and `min_child_weight` (1 to 4). For the Bernoulli Naive Bayes model, we explored alpha values between 0.01 and 2.0, binarize thresholds (0.0 to 1.0), and toggled the `fit_prior` option.

Adjusting these hyperparameters led to substantial improvements in macro-average recall and F1-score, particularly for minority classes. For example, limiting the tree depth and increasing the number of estimators improved model generalization, while using smaller learning rates in XGBoost helped control overfitting. The final selected hyperparameters for each model are summarized in Table 4. These values were selected based on their performance across cross-validation folds, prioritizing macro-average recall to ensure balanced detection across all 34 classes.

In order to evaluate the model, average macro recall and accuracy metrics were selected. Accuracy is the proportion of true predictions to the total number of predictions made.

Table 4. Optimized hyperparameters and their search ranges using RandomizedSearchCV.

| Model | Hyperparameter | Selected | Range |
|-----------------------|--------------------------------|----------|----------------------------|
| Decision Tree | <code>max_depth</code> | 20 | [10, 20, 30, 40] |
| | <code>min_samples_split</code> | 4 | [2, 5, 10] |
| | <code>min_samples_leaf</code> | 2 | [1, 2, 4] |
| | <code>criterion</code> | gini | ['gini', 'entropy'] |
| Random Forest | <code>n_estimators</code> | 150 | [50, 100, 200, 300] |
| | <code>max_depth</code> | 25 | [10, 20, 30] |
| | <code>min_samples_split</code> | 5 | [2, 5, 10] |
| | <code>min_samples_leaf</code> | 2 | [1, 2, 4] |
| | <code>criterion</code> | gini | ['gini', 'entropy'] |
| Bernoulli Naive Bayes | <code>max_features</code> | 'sqrt' | [None, 'sqrt', 'log2'] |
| | <code>alpha</code> | 150 | [0.01, 0.1, 0.5, 1.0, 2.0] |
| | <code>binarize</code> | 25 | [0.0, 0.5, 1.0] |
| XGBoost | <code>fit_prior</code> | 5 | [True, False] |
| | <code>n_estimators</code> | 200 | [50, 100, ..., 500] |
| | <code>max_depth</code> | 6 | [4, 6, 8, 10] |
| | <code>learning_rate</code> | 0.1 | [0.01, ..., 0.6] |
| | <code>subsample</code> | 0.8 | [0.3, ..., 0.9] |
| | <code>colsample_bytree</code> | 0.9 | [0.5, ..., 0.9] |
| | <code>min_child_weight</code> | 2 | [1, 2, 3, 4] |

This metric proves advantageous in scenarios where classes are evenly distributed, signifying that they possess comparable instance quantities [Bispo et al., 2024].

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (4)$$

Accuracy is an important metric. It is defined by Equation 4, in which TP refers to true positive outcomes, TN refers to true negatives, FP false positives, and FN false negatives. However, evaluating accuracy in isolation can be misleading when dealing with imbalanced datasets, in this cases, Recall, also known as sensitivity and Precision, also known as specificity reveal more about the model than accuracy. For this reason, this study also utilizes other metrics in conjunction with accuracy in order to improve a comprehensive understanding of a model's performance, overcoming these limitations.

Recall, as defined by Equation 5, is the ratio of true positives to the total amount of positive samples within the testing dataset. This measurement evaluates the model's performance in identifying all the true positive cases. Recall is the fraction of actual positive cases that are correctly classified, also known as true positive rate.

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

Other usual metrics were also calculated in order to better evaluate the model performance. Precision is the frequency of true positives among all positive classifications, as represented in Equation 6.

$$Precision = \frac{TP}{TP + FP} \quad (6)$$

The F1-score is defined as a harmonic mean of precision and recall, given by Equation 7.

$$F1 - score = \frac{2 * Precision * Recall}{Precision + Recall} \quad (7)$$

This study selected recall and accuracy as the most significant metrics for model evaluation and prioritized recall over accuracy because the potential negative impact of false negatives outweighs those of false positives, in their particular use case. In this context, a false negative occurs when a malicious packet is mistakenly classified as benign. This can have severe consequences, as it allows malicious activities to proceed undetected. For instance, if a malware distribution attempt is misclassified, it could lead to widespread infections.

4 Results

This section describes the performance of RandomForest, DecisionTree, XGBoost and BernoulliNB classification models in malicious activity within IoT network traffic and discusses the results obtained at each development stage. As one of the first steps, a feature selection approach based on the decision tree was employed for best performance and to mitigate bias and overfitting. The 46 features were ranked by their importance scores, and the top features were selected based on a threshold of three decimal places, resulting in of 35 variables to be considered by the model as illustrated in Figure 5.

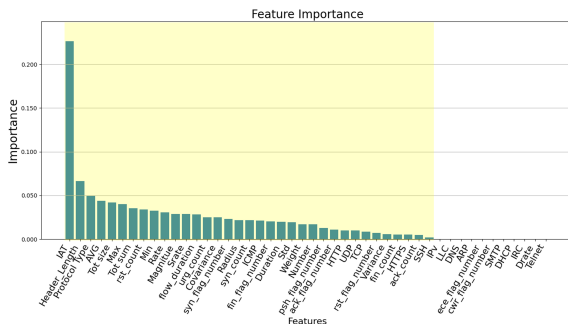


Figure 5. CICIoT2023 features ordered by importance

As shown in Figure 5, the Inter-Arrival Time (IAT) is the measure of the time elapsed between the arrival of the current and previous packet, and the result is the most important feature in the model. Another important variable is the header length, the measure of the header given in bytes, the total size of the packet, and the Protocol Type, which are also figures among the most important features.

After feature selection, the dataset was split into train and test datasets. Some techniques were applied over the training dataset to handle imbalance issues as the database utilized for this study presents a significantly highly imbalanced distribution across 34 labels, as illustrated in Figure 6. This figure graphically represents the distribution of the total labeled traffic lines, expressed as a percentage for each label. A sample of at most 15 thousand events was created, and SMOTE was employed to generate synthetic samples for the under-represented classes, resulting in a training dataset consisted instances. For another train dataset using class weights, the proportion of each class in the training dataset was also calculated. With 34 classes that are highly imbalanced, under-sampling would significantly reduce the number of samples, potentially leading to a biased model.

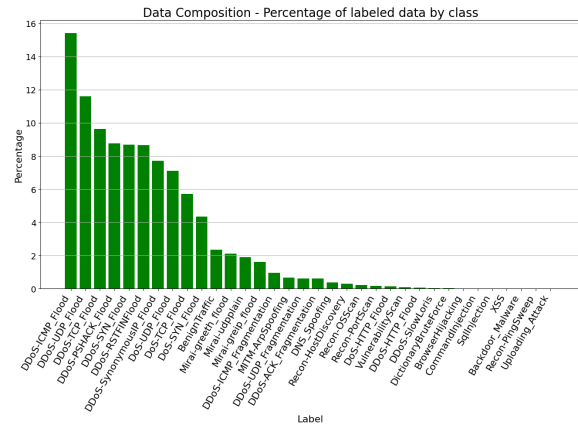


Figure 6. Data distribution of CICIoT2023

Three models, BNB, RF, and DT, were experimented on: the imbalanced dataset, the weighted dataset, and the over-sampled. For each model, Randomized Search Cross-Validation was employed to efficiently tune the hyperparameters with the aim of best recall and avoiding overfitting. Following hyperparameter optimization, each model was trained separately on balanced and imbalanced datasets before being assessed on a held-out test set. Even without over-sampling, models achieved high average results, but when examining the recall metrics macro average and by class, it was observed that the model performed weakly for some positive minority classes, as demonstrated in Table 5.

As shown in Table 5, the Bernoulli Naive Bayes Classifier underperformed in both experiments, achieving 68.35% accuracy and 44.97% in its best case. This means that while the model could correctly predict malicious traffic targeting IoT devices, it often made incorrect predictions, leading to a high number of false positives and false negatives, which is the way the mode underperformed among the three models tested in this study. Table 5 also presents that applying the SMOTE technique to the training datasets of Random Forest, XGBoost and Decision Tree models significantly enhanced their macro-average recall performance. Decision Tree model trained on the unbalanced dataset achieved high accuracy by achieving 99,48% what means that in the test dataset, the model correctly predicted 99,48% of overall positive cases and the model achieved 86,15% on the Macro Recall Average.

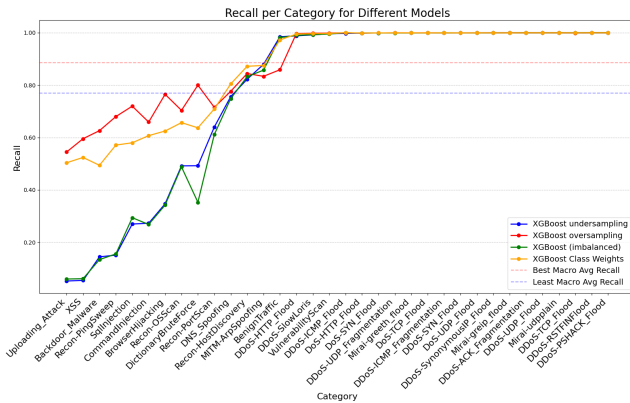
The Random Forest model trained on the unbalanced dataset achieved the highest overall accuracy, correctly predicting 99,54% of all traffic, and the model achieved and 79,71% macro recall average, which means this model had best performance on the majority classes. within the CI-CIoT23 database. However, the Random Forest model trained on the balanced dataset demonstrated superior macro-average recall (87.58%), indicating better performance in identifying specific attack types, working with satisfactory performance also in the minority classes.

The XGBoost model trained on the unbalanced dataset achieved 99.45% accuracy, and 76,96% macro recall average. However, the XGBoost trained on the balanced dataset demonstrated superior macro-average recall (88.56%), indicating the best overall performance in identifying specific attack types, working with satisfactory performance also in the minority classes.

Table 5. Performance comparison of models trained with oversampling techniques.

| Model | Accuracy | Recall (Macro Average) | Precision (Macro Average) | F1-score |
|----------------------|----------|------------------------|---------------------------|----------|
| BNB imbalanced | 68.35 | 44.92 | 38.83 | 67.88 |
| BNB (SMOTE) | 68.35 | 44.19 | 39.06 | 67.84 |
| BNB weighted | 68.36 | 44.99 | 38.84 | 67.88 |
| BNB undersampled | 68.35 | 44.88 | 38.79 | 35.37 |
| DTC imbalanced | 99.48 | 86.15 | 88.89 | 99.47 |
| DTC (SMOTE) | 99.03 | 87.34 | 74.35 | 99.17 |
| DTC weighted | 99.29 | 87.42 | 76.79 | 99.36 |
| DTC undersampled | 99.43 | 86.15 | 88.42 | 99.43 |
| RFC imbalanced | 99.54 | 79.72 | 77.21 | 99.51 |
| RFC (SMOTE) | 98.95 | 87.58 | 71.61 | 99.08 |
| RFC weighted | 99.50 | 84.64 | 92.53 | 99.49 |
| RFC undersampled | 99.06 | 85.18 | 77.04 | 79.52 |
| XGBoost imbalanced | 99.45 | 76.96 | 87.59 | 81.28 |
| XGBoost (SMOTE) | 99.25 | 88.56 | 76.85 | 99.34 |
| XGBoost weighted | 99.56 | 86.54 | 94.58 | 99.55 |
| XGBoost undersampled | 99.44 | 77.45 | 87.56 | 99.41 |

Figure 7 presents a comparative analysis of recall values for XGBoost models trained on the original imbalanced dataset, corresponding to the blue line, a XGBoost model trained on a balanced dataset using the SMOTE algorithm, represented by the red line, and the green line, representing the model trained with class weights. The vertical dashed red and blue lines represent the best and the least macro average recall among the analyzed models, respectively.

**Figure 7.** Evaluation of recall per classes on XGBoost trained with balanced and unbalanced dataset

As illustrated in Figure 7, all models achieved similar performance in the majority classes. However, the XGBoost model trained with oversampling generally achieved higher recall values compared to the model training on the imbalanced dataset, achieving better performances on predicting specific attacks including Uploading Attacks, Cross-site Scripting (XSS), Backdoor Malware, Reconnaissance PingSweep, SQL Injection, Command Injection, Browser Hijacking, Operational System Scan Reconnaissance (OSScan Recon), Dictionary Brute Force. The results suggest that oversampling techniques, specifically SMOTE, effectively improved the model's ability to identify instances from the minority classes. Additionally, training the model with class weights also led to better performance on minority classes

compared to the imbalanced model, highlighting the negative impact of class imbalance on the original dataset, as the model was biased towards the majority class, resulting in lower recall for minority classes. While the overall accuracy and weighted average recall were satisfactory, there was a significant disparity in performance between the majority and minority classes.

5 Conclusions

This study explored the efficacy of machine learning models in detecting malicious traffic targeting IoT devices. By evaluating the performance of Bernoulli Naive Bayes, Random Forest, XGBoost and Decision Tree classifiers on the CIIoT2023 dataset, we demonstrated the potential of machine learning techniques in safeguarding IoT environments.

Our findings indicate that XGBoost and Random Forest classifiers outperformed the other models, achieving high accuracy and recall scores, respectively in detecting a wide range of malicious activities. This superior performance can be attributed to its ability to handle complex relationships between features and its resistance to over-fitting.

Moreover, we found that addressing class imbalance through the use of oversampling techniques, specifically SMOTE, significantly improved the models' ability to detect minority class instances. By artificially increasing the number of samples in underrepresented classes, we mitigated the bias towards the majority class and enhanced the overall performance of the models.

To complement these findings with a broader perspective, we analyzed how our work compares with previous studies in the field. Compared to the related studies reviewed in Section 2, our approach addresses several limitations commonly observed in the literature. While many prior works are limited to binary classification or focus on a small subset of attack types, we target multiclass detection of 34 distinct attacks using lightweight models suitable for deployment in real IoT environments. Furthermore, our work dif-

fers by explicitly incorporating resampling techniques to mitigate class imbalance and by evaluating performance using macro-average metrics, ensuring fair assessment across both majority and minority classes.

Building upon this comparative analysis, we identified several directions for further research. While our results are promising, several avenues for future research remain. Exploring more advanced deep learning techniques, such as convolutional neural networks and recurrent neural networks, could potentially lead to even more accurate and robust detection models. Additionally, investigating the impact of different feature engineering techniques and hyperparameter tuning strategies could further optimize the performance of machine learning models for IoT security.

Furthermore, we emphasize the practical applicability of our findings. Beyond the experimental results, the proposed models present a viable solution for integration into real-world IoT security frameworks. Due to their low computational complexity, classifiers such as XGBoost, Decision-Tree and Random Forest—when properly tuned and combined with oversampling techniques—can be embedded in edge or fog computing nodes to provide near real-time intrusion detection capabilities. These models can serve as core components of lightweight intrusion detection systems (IDS) tailored to resource-constrained IoT environments. However, some limitations must be considered. The need for labeled data for supervised learning may restrict scalability, and model retraining is necessary to adapt to evolving attack patterns. Additionally, the performance of oversampling techniques like SMOTE depends on the representativeness of the minority class samples, which may vary across deployments. Future research should focus on online learning approaches and hybrid architectures to overcome these challenges and improve adaptability in dynamic IoT scenarios.

In light of this, we also explored complementary data balancing strategies. As part of our extended evaluation, we also incorporated Random Undersampling as an alternative strategy to address the class imbalance in the CIIoT2023 dataset. The results, included in the experimental section, provide a comparative view of how undersampling performs relative to SMOTE in multiclass IoT traffic classification. While SMOTE yielded superior performance in our tests, undersampling demonstrated reasonable results with reduced training time, making it a viable option for real-time applications. Future work may explore other resampling techniques, such as Borderline-SMOTE, SMOTE-ENN, and ADASYN, to further improve the detection of minority classes and increase the robustness of the proposed models.

Ultimately, our findings contribute to the broader objective of enhancing cybersecurity in IoT ecosystems. In conclusion, this study highlights the potential of machine learning for enhancing IoT security. By developing effective models for detecting malicious traffic, we can contribute to a more secure and resilient IoT ecosystem. Future research should focus on addressing the limitations of current approaches and exploring new techniques to improve the detection of emerging threats.

Declarations

Acknowledgements

The authors would like to thank the University of Brasilia for the support.

Funding

This research received no funding

Authors' Contributions

All the authors contributed equally to this work. All authors read and approved the final manuscript.

Competing interests

The authors declare that they have no competing interests.

Availability of data and materials

This work uses the CIIoT2023, a publicly available dataset.

References

- Abdelmoumin, G., Rawat, D. B., and Rahman, A. (2022). On the performance of machine learning models for anomaly-based intelligent intrusion detection systems for the internet of things. *IEEE Internet of Things Journal*, 9(6):4280–4290. DOI: 10.1109/JIOT.2021.3103829.
- Abdul Samad, S. R., Balasubramanian, S., Al-Kaabi, A. S., Sharma, B., Chowdhury, S., Mehbodniya, A., Webber, J. L., and Bostani, A. (2023). Analysis of the performance impact of fine-tuned machine learning model for phishing url detection. *Electronics*, 12(7):1642. DOI: 10.3390/electronics12071642.
- Algabroun, H. and Håkansson, L. (2025). Parametric machine learning-based adaptive sampling algorithm for efficient iot data collection in environmental monitoring. *Journal of Network and Systems Management*, 33(5):1–22. DOI: 10.1007/s10922-024-09881-1.
- Amaouche, S., Guezzaz, A., Benkirane, S., Azrou, M., Khattak, S. B. A., Farman, H., and Nasralla, M. M. (2023). Fscb-ids: Feature selection and minority class balancing for attacks detection in vanets. *Applied sciences*, 13(13):7488. DOI: 10.3390/app13137488.
- Arshad, A., Jabeen, M., Ubaid, S., Raza, A., Abualigah, L., Aldiabat, K., and Jia, H. (2023). A novel ensemble method for enhancing internet of things device security against botnet attacks. *Decision Analytics Journal*, 8:100307. DOI: 10.1016/j.dajour.2023.100307.
- Attou, H., Mohy-eddine, M., Guezzaz, A., Benkirane, S., Azrou, M., Alabdultif, A., and Almusallam, N. (2023). Towards an intelligent intrusion detection system to detect malicious activities in cloud computing. *Applied Sciences*, 13(17):9588. DOI: 10.3390/app13179588.

- Awad, M., Fraihat, S., Salameh, K., and Al Redhaei, A. (2022). Examining the suitability of netflow features in detecting iot network intrusions. *Sensors*, 22(16):6164. DOI: 10.3390/s22166164.
- Behnke, M., Briner, N., Cullen, D., Schwerdtfeger, K., Warren, J., Basnet, R., and Doleck, T. (2021). Feature engineering and machine learning model comparison for malicious activity detection in the dns-over-https protocol. *IEEE Access*, 9:129902–129916. DOI: 10.1109/ACCESS.2021.3113294.
- Bispo, G. D., Vergara, G. F., Saiki, G. M., Martins, P. H. d. S., Coelho, J. G., Rodrigues, G. A. P., Oliveira, M. N. d., Mosquera, L. R., Gonçalves, V. P., Neumann, C., et al. (2024). Automatic literature mapping selection: Classification of papers on industry productivity. *Applied Sciences*, 14(9):3679. DOI: 10.3390/app14093679.
- Büyükececi, M. and Okur, M. C. (2022). A comprehensive review of feature selection and feature selection stability in machine learning. *Gazi University Journal of Science*, 36(4):1506–1520. DOI: 10.35378/gujs.993763.
- Chalé, M. and Bastian, N. D. (2022). Generating realistic cyber data for training and evaluating machine learning classifiers for network intrusion detection systems. *Expert Systems with Applications*, 207:117936. DOI: 10.1016/j.eswa.2022.117936.
- Chaudhary, P., Singh, A., and Gupta, B. (2025). Dynamic multiphase ddos attack identification and mitigation framework to secure sdn-based fog-empowered consumer iot networks. *Computers and Electrical Engineering*, 123:110226. DOI: 10.1016/j.compeleceng.2025.110226.
- Coscia, A., Dentamaro, V., Galantucci, S., Maci, A., and Pirlo, G. (2024). Automatic decision tree-based nids ruleset generation for dos/ddos attacks. *Journal of Information Security and Applications*, 82:103736. DOI: 10.1016/j.jisa.2024.103736.
- Dasari, S. and Kaluri, R. (2024). An effective classification of ddos attacks in a distributed network by adopting hierarchical machine learning and hyperparameters optimization techniques. *IEEE Access*, 12:10834–10845. DOI: 10.1109/ACCESS.2024.3352281.
- Diwan, T. D., Choubey, S., Hota, H., Goyal, S., Jamal, S. S., Shukla, P. K., and Tiwari, B. (2021). Feature entropy estimation (fee) for malicious iot traffic and detection using machine learning. *Mobile Information Systems*, 2021(1):8091363. DOI: 10.1155/2021/8091363.
- Hou, J., Qu, L., and Shi, W. (2019). A survey on internet of things security from data perspectives. *Computer Networks*, 148:295–306. DOI: 10.1016/j.comnet.2018.11.026.
- Jeon, S.-E., Oh, Y.-S., Lee, Y.-J., and Lee, I.-G. (2024). Suboptimal feature selection techniques for effective malicious traffic detection on lightweight devices. *CMES-Computer Modeling in Engineering & Sciences*, 140(2). DOI: 10.32604/cmescs.2024.047239.
- Ji, I. H., Lee, J. H., Kang, M. J., Park, W. J., Jeon, S. H., and Seo, J. T. (2024). Artificial intelligence-based anomaly detection technology over encrypted traffic: a systematic literature review. *Sensors*, 24(3):898. DOI: 10.3390/s24030898.
- Li, J., Othman, M. S., Chen, H., and Yusuf, L. M. (2024). Optimizing iot intrusion detection system: feature selection versus feature extraction in machine learning. *Journal of Big Data*, 11(1):36. DOI: 10.1186/s40537-024-00892-y.
- Liu, Y., Wang, J., Li, J., Niu, S., and Song, H. (2021). Machine learning for the detection and identification of internet of things devices: A survey. *IEEE Internet of Things Journal*, 9(1):298–320. DOI: 10.1109/JIOT.2021.3099028.
- Mittal, S., Dhall, A., Sharma, T., and Sharma, T. (2025). Deep learning ensemble for iot malware detection with explainable ai and class imbalance mitigation. *Engineering Applications of Artificial Intelligence*, 127:107847. DOI: 10.1016/j.engappai.2024.109560.
- Musleh, D., Alotaibi, M., Alhaidari, F., Rahman, A., and Mohammad, R. M. (2023). Intrusion detection system using feature extraction with machine learning algorithms in iot. *Journal of Sensor and Actuator Networks*, 12(2). DOI: 10.3390/jsan12020029.
- Neto, E. C. P., Dadkhah, S., Ferreira, R., Zohourian, A., Lu, R., and Ghorbani, A. A. (2023). Ciciot2023: A real-time dataset and benchmark for large-scale attacks in iot environment. *Sensors*, 23(13):5941. DOI: 10.3390/s23135941.
- Noman, H. A. and Abu-Sharkh, O. M. F. (2023). Code injection attacks in wireless-based internet of things (iot): A comprehensive review and practical implementations. *Sensors*, 23(13). DOI: 10.3390/s23136067.
- Pedreira, V., Barros, D., and Pinto, P. (2021). A review of attacks, vulnerabilities, and defenses in industry 4.0 with new challenges on data sovereignty ahead. *Sensors*, 21(15). DOI: 10.3390/s21155189.
- Pimenta Rodrigues, G. A., Marques Serrano, A. L., de Oliveira Albuquerque, R., Mayumi Saiki, G., Santedicola Ribeiro, S., Sandoval Orozco, A. L., and García Villalba, L. J. (2024). Mapping of data breaches in companies listed on the nyse and nasdaq: Insights and implications. *Results in Engineering*, 21:101893. DOI: 10.1016/j.rineng.2024.101893.
- Pimenta Rodrigues, G. A., Marques Serrano, A. L., Lopes Espiñeira Lemos, A. N., Canedo, E. D., Mendonça, F. L. L. d., de Oliveira Albuquerque, R., Sandoval Orozco, A. L., and García Villalba, L. J. (2024). Understanding data breach from a global perspective: Incident visualization and data protection law review. *Data*, 9(2). DOI: 10.3390/data9020027.
- Rimal, Y., Sharma, N., and Alsadoon, A. (2024). The accuracy of machine learning models relies on hyperparameter tuning: student result classification using random forest, randomized search, grid search, bayesian, genetic, and optuna algorithms. *Multimedia Tools and Applications*, pages 1–16. DOI: 10.1007/s11042-024-18426-2.
- Rodrigues, G. A. P., Fernandes, P. A. G., Serrano, A. L. M., Filho, G. P. R., Vergara, G. F., Bispo, G. D., Albuquerque, R. d. O., and Gonçalves, V. P. (2025). From rockyou to rockyou2024: Analyzing password patterns across generations, their use in industrial systems and vulnerability to password guessing attacks. *Journal of Internet Services and Applications*, 16(1):69–86. DOI:

- 10.5753/jisa.2025.5034.
- Sharma, A., Mansotra, V., and Singh, K. (2023). Detection of mirai botnet attacks on iot devices using deep learning. *Journal of Scientific Research and Technology*, pages 174–187. DOI: 10.5281/zenodo.8330561.
- Thaseen, I. S., Mohanraj, V., Ramachandran, S., Sanapala, K., and Yeo, S.-S. (2021). A hadoop based framework integrating machine learning classifiers for anomaly detection in the internet of things. *Electronics*, 10(16):1955. DOI: 10.3390/electronics10161955.
- Vergara, G. F., Giacomelli, P., Serrano, A. L. M., Mendonça, F. L. L. d., Rodrigues, G. A. P., Bispo, G. D., Gonçalves, V. P., Albuquerque, R. d. O., and Sousa Júnior, R. T. d. (2024). Stego-stfan: A novel neural network for video steganography. *Computers*, 13(7). DOI: 10.3390/computers13070180.
- Yalçın, N., Çakır, S., and Üaldı, S. (2024). Attack detection using artificial intelligence methods for scada security. *IEEE Internet of Things Journal*. DOI: 10.1109/JIOT.2024.3447876.
- Zhang, Y. and Liu, Q. (2022). On iot intrusion detection based on data augmentation for enhancing learning on unbalanced samples. *Future Generation Computer Systems*, 133:213–227. DOI: 10.1016/j.future.2022.03.007.