computers

MDPI

Article

Toward Improved Machine Learning-Based Intrusion Detection for Internet of Things Traffic

Sarah Alkadi, Saad Al-Ahmadi and Mohamed Maher Ben Ismail *

Department of Computer Science, College of Computer and Information Sciences, King Saud University, Riyadh 11362, Saudi Arabia; sara.alqadi@gmail.com (S.A.); salahmadi@ksu.edu.sa (S.A.-A.)
* Correspondence: mbenismail@ksu.edu.sa

Abstract: The rapid development of Internet of Things (IoT) networks has revealed multiple security issues. On the other hand, machine learning (ML) has proven its efficiency in building intrusion detection systems (IDSs) intended to reinforce the security of IoT networks. In fact, the successful design and implementation of such techniques require the use of effective methods in terms of data and model quality. This paper encloses an empirical impact analysis for the latter in the context of a multi-class classification scenario. A series of experiments were conducted using six ML models, along with four benchmarking datasets, including UNSW-NB15, BOT-IoT, ToN-IoT, and Edge-IIoT. The proposed framework investigates the marginal benefit of employing data preprocessing and model configurations considering IoT limitations. In fact, the empirical findings indicate that the accuracy of ML-based IDS detection rapidly increases when methods that use quality data and models are deployed. Specifically, data cleaning, transformation, normalization, and dimensionality reduction, along with model parameter tuning, exhibit significant potential to minimize computational complexity and yield better performance. In addition, MLP- and clustering-based algorithms outperformed the remaining models, and the obtained accuracy reached up to 99.97%. One should note that the performance of the challenger models was assessed using similar test sets, and this was compared to the results achieved using the relevant pieces of research.

Keywords: intrusion detection; Internet of Things; data quality; model quality; machine learning

Citation: Alkadi, S.; Al-Ahmadi, S.; Ben Ismail, M.M. Toward Improved Machine Learning-Based Intrusion Detection for Internet of Things Traffic. *Computers* **2023**, *12*, 148. https://doi.org/10.3390/computers12080148

Academic Editors: Phivos Mylonas, Katia Lida Kermanidis, Manolis Maragoudakis and Paolo Bellavista

Received: 12 June 2023
Revised: 13 July 2023
Accepted: 18 July 2023
Published: 27 July 2023

1. Introduction

The continuous growth witnessed in various computing fields has promoted several emerging technologies. In particular, the Internet of Things (IoT), in which wired and wireless communications are coupled between devices, has emerged as a paradigm shift regarding the future of the Internet. Diverse IoT devices are capable of sensing, collecting, and communicating in an automated manner. They have been adopted within different areas and services, including healthcare, manufacturing, transportation, education, and smart cities [1–3]. In fact, the number of IoT devices is expected to grow from 0.3 billion in 2003 to over 100 billion in 2040 [4]. This growth reflects the drastic impact of IoT technologies on our daily life and the associated risk of security vulnerabilities. IoT devices generate a massive amount of data that can be exploited to analyze the overall performance of the IoT network. Advanced analyses of the generated data can be conducted to spot security attacks and detect malicious events [1–3].

In this context, machine learning (ML) provides a great contribution to the efforts intended to make IoT networks more secure. Despite the promising results recently reported, the security attacks that target IoT networks, along with the concerns related to resource and processing constraints, have arisen as challenging obstacles facing typical IoT frameworks [1]. Since 2008, many pieces of research have promoted the utilization of ML techniques in intrusion detection systems (IDSs) and have yielded accurate detections of less suspicious traffic with low-cardinality intrusions [1]. Unlike signature-based IDSs

that detect attacks based on the signatures of attacks, anomaly IDSs detect zero-day attacks using machine learning techniques [2]. IDSs can also be categorized into (i) network-based IDSs (NIDSs) at the network level and (ii) host-based IDSs (HIDSs) at the operation system level. A network-based IDS (NIDS) is typically used to monitor the network traffic, whereas the latter version is used to monitor specific malicious system activities, including traffic, application logs, system calls, file-system changes, and others [3].

Mining for hidden patterns and insights in the data can guide the detection task conducted using ML models. In particular, mining techniques consider the different aspects of the data, such as size, data modality, and data processing speeds. Consequently, both machine learning models and data quality play a critical role and affect IDS performance. One should note that handling datasets specifically in security applications creates issues such as incompleteness, duplication, timeliness, and lack of diversity. These issues represent real obstacles to designing accurate ML-based IDSs [1].

Several studies [5–11] were carried out to investigate the quality of ML-based models for IDSs regarding the IoT. The relevant works consider different quality criteria, such as model creation, validation, and optimization. They also employ different ML algorithms covering both conventional and deep learning techniques for intrusion detection in low-power IoT networks. In particular, decision tree (DT), k-means clustering, naive Bayes (NB), artificial neural networks (ANN), recurrent neural network (RNN), convolutional neural networks (CNNs), genetic algorithms (GA), and other ML techniques were extensively investigated [5–11]. Other research [12–14] studied the effects of data in terms of quality, specifications, and issues in related datasets. Examples of popular datasets for IDS-based IoT networks include ToN-IoT, BOT-IoT, and others. These benchmarking datasets were proposed and used for data-related quality assurance purposes. The related works presented useful techniques for handling data issues regardless of model complexity [12–14]. However, these works considered either the data or the model quality-related issues for IDS-based IoT networks. This reflects the need for a comprehensive analysis of both perspectives.

Accordingly, this research is motivated by the need to investigate the impact of both data and model quality in terms of the methods used for enhancing ML-based IDSs for IoT networks. Particularly, experimental scenarios were designed based on six ML models along with four benchmarking datasets to assess ML-based IDSs for IoT networks. This research intends to answer the following research questions:

- What are the main criteria relevant to preparing and selecting proper datasets that preserve the required quality for IDS performance in an IoT context?
- How would quality assurance factors, including both data and model perspectives, impact multi-class ML-based IDSs within an IoT context?
- How would quality assurance methods enhance the performance of ML-based IDSs?
- Can a lightweight framework designed based on model quality assurance methods enhance the IDS detection rate in an IoT context?

Accordingly, the main contributions of this study can be summarized as follows:

- Introducing dataset selection criteria to ensure high-quality data-driven modeling for IDSs within an IoT environment;
- A comprehensive investigation and analysis of the trade-offs regarding quality assurance challenges and solutions in ML-based IDSs considering an IoT environment;
- Improved generalization for the selected supervised ML techniques considering IoT resource-constrained devices;
- A lightweight framework for enhancing the IDS detection rate in an IoT context through employing selected data and model quality assurance methods.

To the best of our knowledge, this research represents the first study that comprehensively investigates data and models quality assurance methods for an effective IDS within an IoT context. The research findings can be perceived as new insights and directions toward improving IDS performance with consideration of IoT constraints.

The rest of the article is organized as follows. In Section 2, works related to ML-based IDSs, data collection and pre-processing, and datasets for IoT-based IDSs are presented. The best practices for efficient IDSs in IoT networks from the perspectives of both models and data are provided in Section 3. Section 4 outlines the proposed approach, and Section 5 depicts the experiential design, including datasets, performance measures, and results. In Section 6, a discussion about the performed experiments and findings is conducted. Finally, the conclusions and future directions are provided in Section 7.

2. Related Works

2.1. ML-Based Intrusion Detection Systems for IoT Networks

Since 1980, IDSs have been introduced as a primary solution that can monitor host-based or network-based traffic to detect intrusion attempts [3]. Typically, IDSs can be grouped into three main categories: (i) signature-based, (ii) anomaly-based, and (iii) hybrid-based IDSs. As mentioned earlier, signature-based IDSs use a database of known attack signatures to flag intrusion patterns, which makes zero-day attacks undetectable. For the anomaly-based category, the detection is carried out using ML techniques, whereby normal behaviors are collected and analyzed to identify potential abnormalities so that zero-day attacks can be identified. Lastly, the hybrid approach relies on both mechanisms, known attack signatures and ML techniques, and exploits their advantages to enhance the overall detection rate [14].

Despite the widespread use of IDSs in traditional networks, there are still several issues when it comes to IoT environments and their specifications. These specifications compose barriers, ranging from computing capabilities to storage capacities, that affect the deployment of IoT-based IDSs. In terms of computing capabilities, the IoT contains large-scale nodes that pose a greater number of real-time distribution and processing issues. However, IoT computing-constraint issues create a situation where maintaining the work of IDS agents is challenging [14]. Additionally, the multi-hop architecture of the IoT poses an issue due to storage capacity constraints regarding the IoT. Moreover, the specified protocols for IoT networks produce new security and privacy challenges. Such protocols include IPv6, Low-Power Wireless Personal Area Network (6LoWPAN), Constrained Application Protocol (CoAP), and others. Furthermore, administrating IoT networks poses new issues since large amounts of alerts are generated by IDSs, which cannot be analyzed manually. Several enhancements need to be considered to overcome these issues, such as alert correlation, data visualization, and early detection [3].

Accordingly, several pieces of research have been proposed to investigate IDS adoption for IoT networks using ML-based solutions. In fact, ML provides data-driven IoT-intelligent solutions that are suitable for IoT security specifications. They exhibit a promising ability to detect unknown attacks early and in real-time [15]. In this context, a set of instances with a range of features is fed into the ML-based IDS. Those features are correlated with either system events or network packets representing labeled incidents. Thus, labeled data are considered crucial for building detection models using ML-based IDSs [16,17]. ML algorithms are designed to learn the association between the considered features and the pre-defined class labels. In addition, several data quality issues can be identified while handling ML input data for IDSs. Such issues include data inconsistency, missing values, a lack of labeled data, duplication, a lack of variety, imbalances, and noise [16]. The pieces of research relevant to ML-based IDSs within the IoT have undertaken investigations using databases from the ACM, Google Scholar, IEEE, Science Direct, Scopus, and Springer. The main focus of the selected publications is ML categorization, ML-based IDSs for the IoT, quality assurance for ML-based IDSs, and IoT security. The chosen timeframe of the selected publications is between 2019 and 2023. The surveyed works are summarized in Table 1, where the relevant information, such as the learning algorithms, datasets, and evaluation metrics, is depicted.

Table 1. A summary of ML-based IDS-related works regarding the IoT.

Reference	Year	ML Technique	Dataset	Evaluation Metric	Classification
Tareq et al. [12]	2022	DenseNet Inception Time	ToN-IoT Edge-IIoT UNSW-NB15	Accuracy, Precision, Recall, F1-measure	Multi-class
Koroniotis et al. [18]	2019	RNN, SVM LSTM	BOT-IoT	Accuracy, Precision, Recall	Multi-class
Kanimozhi et al. [19]	2019	ANN, RF-DT	UNSW-NB15	Accuracy, Precision, Recall, F1-measure	Multi-class
Nawir et al. [20]	2019	NB, MLP DT	UNSW-NB15	Accuracy	Binary
Alsaedi et al. [21]	2020	LR, LDA KNN, RF CART, NB SVM, LSTM	ToN-IoT	Accuracy, Precision, Recall, F1-measure	Multi-class
Kasongo et al. [22]	2020	ANN, LR KNN, SVM DT	UNSW-NB15	Accuracy, Precision, Recall, F1-measure	Multi-class
Thaseen et al. [23]	2020	C4.5, NB RF, MLP SVM, CART KNN, ANN Ensemble Learning	BOT-IoT	Accuracy, Precision, Recall, F1-measure, Specificity, AUC.	Multi-class
Sugi et al. [24]	2020	LSTM, KNN	BOT-IoT	Accuracy	Multi-class
Sarhan et al. [25]	2021	DFF, RF	ToN-IoT	Accuracy, AUC F1-measure, Detection Rate, False Alarm Rate, Prediction Time	Multi-class
Ferrag et al. [26]	2021	DT, RF SVM, KNN DNN,	Edge-IIoT	Accuracy, Precision, Recall, F1-measure	Multi-class
Fatani et al. [27]	2021	CNN	BOT-IoT	Accuracy, Precision, Recall	Multi-class
Yin et al. [28]	2023	MLP	UNSW-NB15	Accuracy, Precision, Recall, F1-measure, AUC, FPR	Multi-class
Moustafa et al. [29]	2016	ANN	UNSW-NB15	Accuracy, Precision, Recall	Multi-class
Gad et al. [30]	2021	LR, NB, DT, RF, AdaBoost, KNN, SVM, XGBoost	ToN-IoT	Accuracy, Precision, Recall	Multi-class

2.2. Quality Assurance Challenges of ML-Based IDSs

Quality assurance regarding ML-based IDSs poses many challenges from both data and model perspectives. Research work coins the term "data quality" as a metric of data used to guarantee the proper building of ML systems. It can be described as a multi-dimensional concept covering both the qualitative and quantitative aspects of data. In terms of data quality challenges, these can be categorized into label-, instance-, and feature-based challenges [16]. In terms of label-based issues, mislabeled data and imbalanced data are typical examples in this category. KDD'99 [31] is an example of an imbalanced IDS dataset that includes millions of records with very few data points representing malicious traffic. Having such an imbalanced level of distribution leads to an inefficient decision boundary

since most of the data points are benign. ML models rely highly on the accuracy of training data, which, in turn, (if they are mislabeled) comprise the predictive accuracy of models and, thereby, lower the confidence of it [32,33]. Research has shown that balancing data could improve neural network performance in minor classes with few instances [34]. Another type of label-based issue is the difficulty of labeling data due to huge volumes and noisy labels. A terabyte of data is labeled in a manual manner by a knowledge expert within the IDS field, meaning only a limited number of datasets are available for benchmarking [35].

The second category is mapped to instance-based issues where a shortage of data is investigated. Such a problem includes data scarcity, where no strong distinction can be made. This happens due to the lack of malicious traffic that needs to be learned when compared to benign traffic behavior. It can also happen due to having many missing values and outliers, which negatively affects the performance of the ML model. Different ML algorithms are sensitive to outliers, such as clustering and regression analysis [36,37].

The last category relates to feature-based issues, covering the noise, redundancy, feature sparsity, and correlation matters of data. Current IDSs primarily consider the development of the ML model and dismiss the investigation of detailed feature selection. The poor analysis of feature selection can cause limitations in terms of capturing the IoT traffic in terms of semantic relationships and detecting unknown attack types [36,37].

With regard to model quality, this also represents an important element since the building of the model is an integrated process that is drastically impacted by several factors, such as model selection and fitting, the training process, and hyper-parameter optimization. In other words, the careful selection of these criteria contributes toward a better generalization capability in the ML model [38]. For model selection, the complexities of space and time represent critical criteria in the IoT environment, where computational resources are scarce and require lightweight IDS solutions. Accordingly, adopting deep learning techniques is quite challenging due to limitations in terms of power utilization, memory consumption, and data processing resources. Therefore, conventional ML techniques are preferred for the IoT environment because of their robustness, stable performance, and relative simplicity [39,40]. Hansson et al. [41] reported the wide adoption of branching logic, such as is found in decision trees (DTs), within several ML-based applications.

In terms of model fitting and training, unbiasedness and robustness prediction represent sensitive factors where errors can be found due to some bias and/or variance. Bias represents the difference between the expected and the predicted values. Thus, a model with high bias reflects a data underfitting problem and yields considerable misclassifications regarding training and test instances. On the other hand, variance pays attention to the model's sensitivity toward changes in the training dataset. Models with high variance typically overfit data and exhibit a low generalization capability. This gives high error rates when using the test data, whereas it yields considerably good performance when using the training data. Regularization or early stopping can be used to overcome overfitting problems [37,38,40].

Finally, hyper-parameter optimization represents another issue that deals with selecting the perfect settings for these parameters iteratively. This is considered computationally challenging since a new dimension to the search space is added when every new hyper-parameter is defined. Moreover, insufficient knowledge about the classification problem domain creates further issues in the optimization process. Examples of hyper-parameters are DT depth, the kernel type of the support vector machine (SVM), and others [40].

2.3. Data Collection, Pre-Processing, and Selection for IDSs Regarding the IoT

IoT networks are associated with many objects that sense and collect huge amounts of data through a sensor, gateway, or router. The collection of IoT data is a crucial step when building a ML-based IDS. In fact, several sources are utilized to collect different formats of data instead of having only a single data source. Moreover, several challenges, such as inaccessible physical locations, changing environments, decentralized systems, and data being updated, altered, or deleted while the IoT is rebooted, are faced [42,43].

Furthermore, several pre-processing techniques are required to preserve good data quality and thereby enhance the detection accuracy of IDSs [44]. Such techniques tackle feature and pattern transformations for an efficient numerical representation. Moreover, the optimal selection of highly relevant features can be employed for dimensionality reductions, thereby preserving the meaningful properties of the data [25]. Additionally, removing outliers, omitting redundant data, fixing imbalances, ensuring real-time traces, and avoiding data scarcity and overlapping also contributes to high-quality performance [16].

It is worth noting that inefficient pre-processing has a direct impact on data quality in terms of bias, noise, sparsity, and data overlap [14]. For example, data imbalance can lead to a sampling bias and, thereby, a misinterpretation of performance. Inferring a meaningful conclusion becomes challenging since the dataset does not reflect the true data distribution effectively. If some classes are predominant, then any potentially low false-positive rates can affect the classification results. Moreover, a high amount of sampling bias is found within the security domain due to the challenges of data collection and the rarity of publicly available datasets. Another type of bias is data mislabeling, where the ground-truth labels are inaccurate, leading to unstable ML-based security system performance. As such, it is crucial to consider labeling uncertainty to avoid the inherent bias in ML-based IDSs [45].

In terms of IoT datasets, the relevant datasets at present are constrained by their reduced size and limited quality [43]. In fact, the current IDS datasets are characterized by several issues. For example, NSL-KDD [42] is not comprehensive and contains many redundant records. These drawbacks become even more important when the dataset is considered in an IoT context. Moreover, protecting IoT devices in an IDS context needs extensive experiments to find and adopt any tailored solutions. The authors in [46] refer to this problem in regard to privacy, such as the system owners not sharing the security information publicly [46]. This led to several efforts to create new publicly labeled datasets. Authors have worked on incorporating both benign and malicious instances of either real or simulated IoT traffic into new benchmark datasets [47]. Małowidzki et al. [48] and Ring et al. [49] identified a group of essential features for a good cybersecurity dataset. They mentioned features such as being realistic, being up-to-date, covering all typical attacks, being labeled, being flow-based, and system working hours.

As such, the dataset selection process is associated with several properties that contribute toward building benchmark datasets. Such properties consist of general information, duration, context, labels, and features. Regarding general information, many elements, including data timeframe, data availability, data size, data format, and metadata, are considered. These elements also identify the total number of packets, flows, logs, and instances, as well as the associated format. Moreover, they use metadata to present attributes like IP addresses, hosts, network structures, network configurations, and current attacks. In terms of duration, data are used with a timestamp to reflect their collection time, such as daytime, nighttime, weekdays, or weekends. For the context aspect, it indicates the scenario of collecting the network data by taking into consideration information about the monitored system, such as the environment, operating systems, and processes. Moreover, data labels, which are identified as either benign or malicious, require an accurate process for correct labeling for all training, testing, and validation data. Finally, the features associated with the input data are categorized into qualitative or quantitative types, such as network features, packet features, n-gram features, and so on. Dimensionality-reduction techniques can be applied to the feature set to enhance detection performance [16,36].

Consequently, a detailed list of dataset selection criteria was proposed by Gharib et al. [50], which is presented in Table 2. Identifying such criteria can contribute to new reliable network intrusion datasets where the compliance issues related to these criteria are addressed, thereby leading to good generalization performance. For example, the Canadian Institute for Cybersecurity (CIC) at the University of New Brunswick follows Gharib et al. [50] criteria, where improvements in terms of duplications and uncertainties are adopted for their datasets [51]. Accordingly, the chosen datasets for this paper were evaluated to comply with these criteria.

Table 2. Dataset selection criteria introduced in [50].

No.	Criteria	Brief
1	Complete Network Configuration	Realistic network configuration with all essential equipment such as PCs, servers, routers, and firewalls.
2	Complete Traffic	Contains a sequence of packets that originate from a source (to a destination) and can be realistic or pseudo-realistic with both real and simulated world traffic.
3	Labeled Dataset	All dataset instances are tagged with correct labels for a valid and reliable analysis.
4	Complete Interaction	All network interactions, such as within or between internal LANs, are recorded for better interpretation and evaluation of the prediction results.
5	Complete Capture	All network traffic without removing any non-functional or not labeled parts is captured to better calculate the false-positive rate of an IDS.
6	Available Protocols	All protocols of network traffic are recorded and traced, such as HTTP, FTP, VOIP, and other protocols.
7	Attack Diversity	Different attacks and threat scenarios are supported by the offline dataset, including a variety of categories.
8	Anonymity	Datasets without payload information for privacy concerns that affect the usefulness of the dataset.
9	Heterogeneity	Building datasets from different sources such as network traffic, operating systems logs, or network equipment logs.
10	Feature set	A group of related features from several data sources, such as traffic or logs.
11	Metadata	A complete documentation of the network configuration, operating systems, attack scenarios, and others.

3. Best Practices for Efficient IDSs in IoT Networks

Since quality issues affect the reliability, efficiency, and robustness of IDS performance, different quality requirements have been recently proposed [16,52]. Particularly, IoT-based IDSs entail several quality requirements to enhance their detection performance. These requirements can be grouped into two main categories: (i) data-related and (ii) model-related quality requirements. Data-related requirements cover the needed pre-processing methods, whereas the model-related requirements tackle the proper selection of detection in ML models, the performance measures, and the related hyper-parameters [16,52].

3.1. Data-Related Quality Assurance Solutions

Several solutions have been proposed and implemented to handle the aforementioned issues. For example, solutions related to label-based issues might include oversampling and under-sampling, optimal feature extraction, and genetic programming. For instance-based issues, the solutions might include adversarial example augmentation, transfer learning, and reinforcement learning. For feature-based issues, the solutions might include feature selection, feature normalization, dimensionality reduction, and redundancy elimination methods [30]. Consequently, some well-known data-related solutions can be selected as follows.

3.1.1. Class Sampling

Sampling methods are popular solutions intended to tackle the issue of imbalanced data. They can be classified into under-sampling and over-sampling methods. On the one hand, under-sampling is a reduction mechanism performed on dataset instances to balance class distribution by eliminating the majority of class instances. It can be classified into two categories: (i) fixed ratio under-sampling and (ii) cleaning under-sampling. Fixed ratio under-sampling is a statistical measure that relies on random selection methods. It calculates the proportions of the numbers of a given class over the total number of other classes. Accordingly, major class instances are calculated and randomly down-sampled to a specific number of records; this number is obtained using data exploration and analysis techniques. On the other hand, under-sampling relies on different mechanisms, such as clustering, nearest-neighbor analysis, or classification accuracy. One should note that no specific number of target classes is defined due to performing cleaning when considering the feature space. Overall, under-sampling penalizes the over-representation of major

classes without affecting the minor classes and decreasing the imbalanced cases. However, under-sampling might cause a loss of useful data for the classification process [51,53].

In addition, over-sampling can be perceived as a replication mechanism performed on dataset instances to balance class distribution by increasing the minority class instances. Such balancing can be performed either using data-balancing methods, such as random up-sampling, or feature-space methods, such as the synthetic minority over-sampling technique (SMOTE) [54]. The random up-sampling works on replicating minor class instances randomly to balance their distribution. However, it leads to potential overfitting problems. SMOTE avoids this problem by increasing the minor class instances without performing replication. Linear interpolating is employed between the close minor class instances using k nearest neighbors to create new ones. This overcomes the overfitting obstacle where the classifier's decision boundaries are moved from the space of the minority classes and up-sample in specific dimensions. Both under-sampling and over-sampling can be combined to remove all class instances from the training set that are misclassified [51].

3.1.2. Feature Pre-Processing and Selection

Dataset instances are associated with a set of attributes that can be quantitative features, qualitative features, n-gram features, network features, and packet features [16]. However, redundancy, noises, outliers, and missing and irrelevant values in the feature set degrade detection accuracy. Several methods are proposed to overcome the performance inhibitors [55].

These methods include feature selection, which plays an important role in the enhancement of detection performance for the machine learning classifiers. It is employed to manage computational complexity, reduce the data dimensions, and avoid data duplications, thereby lowering the rate of false alarms [56]. According to Shetye [57], three methods of feature selection are recommended in several contexts, including imbalanced data issues. Such methods are grouped as follows: (i) filter-based, which works on analyzing the correlation and relevance, (ii) wrapper-based, which employs the ML algorithm in eliminating recursive features, and (iii) embedded-based, which iteratively extracts features using the methods of regularization and optimization. In such a context, SelectKBest [58] is a well-known solution that represents a filter-based method used to extract the first k features with the highest scores. The scores are calculated using a specific function, such as an ANOVA F-measure, the information gain ratio, or X^2. The features with high scores contribute the most toward the quality of the training dataset. However, SelectKBest dismisses the rare classes with strong consideration for the largest classes. This requires the experimental design to first consider finding the rarest class with important features, followed by adding the features needed for every class [55,56].

Normalization is also a well-known method that handles the issue of having features on different scales and the potential strong bias with large-scale features. Normalizing the features is performed on both the training and testing sets. It gives better convergence, adds suitable regularization, and decreases the generalization error. As such, the features are converted into a suitable range using different techniques like MinMax scaling, StandardScaler, and others [36,59,60]. Another method consists of a transformation, which deals with feature engineering by deriving new attributes from existing ones using some domain knowledge. A baseline can be employed to assess the usability and suitability of new features. The new features can be derived by combining, encoding, augmenting, discretization, and other methods. This involves converting the features from categorical into numerical representations or vice versa using mechanisms such as one-hot encoding. Moreover, this involves discretizing continuous features into ranges and augmenting the features with other features to form new dimensions [56,59]. Furthermore, noise reduction is another pre-processing task that is conducted to filter out irrelevant and unwanted instances. However, removing the noise requires careful analysis and evaluation to avoid removing typical instances [59]. Finally, a data imputation method is used that handles the missing, NaN, and unique values to ensure the preparation of a completely meaningful dataset. The imputation process might calculate interpolated values, mean or median

values, or a special value symbol. It might also include value substitution using model predictions, matrix factorization, or performing multiple imputations. However, multiple experiments need to be conducted to find out the most proper techniques [59].

3.2. Model-Related Quality Assurance Solutions

Assuring the quality of ML models involves many aspects that can be grouped into two main categories: the selection of the ML technique and performance evaluation. In terms of ML technique selection, dataset availability and the required learning task both define the proper selection of the model. With regard to performance evaluation, several metrics have been introduced in the literature to study the effectiveness of ML models. Consequently, the well-known model-related solutions are presented below:

3.2.1. ML Technique Selection

Various conventional and deep learning techniques are being adopted today in different fields. The model selection process encounters several considerations, such as the data type and the learning problem. The model needs to be tailored to incorporate the suitable features and characteristics of the dataset for better solutions. The no free lunch theorem confirms this fact, as ML models cannot perform the same when used on different problems. Using simpler models with lower capacities as a baseline is recommended as a starting attempt, followed by enhancing the capacity gradually. It also includes the validation aspect to ensure the appropriateness of the model parameters without increasing unnecessary complexity and isolation. Accordingly, ensuring domain knowledge contributes toward boosting model quality [59].

For learning problems, several aspects need to be considered while training the model, including the objective, optimization, regularization, and cross-validation. The objective of the learning problem is derived from the target application to measure the model's performance. On the other hand, optimization works on tuning the model hyper-parameters and applying cost-sensitive learning to support achieving the objective. Cost-sensitive learning shows an optimization solution where the weight of misclassification errors is calculated based on the class weight. Regularization attempts to ensure a bias-variance trade-off and avoid related problems, such as overfitting or underfitting. Finally, cross-validation is used to support the proper generalization of the ML model by splitting data into training, validation, and testing sets [51,59].

3.2.2. Performance Evaluation

The performance of IDSs is typically affected by the quality of both the datasets and models. Several performance metrics have been employed for evaluating the classification and detection tasks. The most popular evaluation metrics are derived from the confusion matrix, such as accuracy, precision, recall, F1-measure, and others. They are simple, clear, and widely used, which eases comparisons with other related research work. However, those metrics are very sensitive to the classes' representation within the dataset. They can be affected by minor changes in the proportions of the classes. Other measures have been introduced by authors to tackle some of the specifications of either the datasets or models. Such measures include the index of balanced accuracy (IBA) for calculating the geometric mean of recall G^{-} that is used for imbalanced classes. Another metric is the balanced accuracy score (BAS), which calculates the recall and discards the precision. Prediction bias also can be used to assure performance quality as the difference between the expected prediction and true prediction regarding model accuracy. Finally, variance is measured by repeatedly computing the model's prediction using training data to track its changes [51].

Another important aspect of performance evaluation is the validation of model performance. In other words, it is important to have an additional test set that is disjointed from the validation and training sets. The test set should reflect the real scenarios of the original datasets, including all dataset dimensionalities, and avoid dependencies on the training set [59]. In the case of dataset availability limitations, a cross-validation technique

is proposed to overcome such challenges. Different types of this technique can be found, such as leave-one-out cross-validation (LOOCV) and k-fold cross-validation. The first type splits a dataset of size n into two subsets of size 1 and $n - 1$ in a repeated manner $n - 1$ times. The second type is more efficient in terms of computation power, where random splits are performed repeatedly for the dataset into k-folds of equal size [37].

4. Proposed Approach

In this section, we outline the proposed framework intended to assess the performance of the classification tasks. Specifically, the main objective of the proposed framework is to investigate the impact of data and model quality on enhancing the detection rate of IDSs. It works on maintaining the trade-off between the quality factors and performance efficiency when building ML-based IDSs for the IoT. Such constrained IDSs deal with several types of traffic data that require tailored solutions. Accordingly, both quality perspectives need to be carefully considered to balance the relevant constraints in terms of time, cost, and performance. A multi-class scenario is designed to yield the identification of multiple pre-defined labels. These labels include different types of popular attacks targeting IoT networks. Moreover, further consideration of IoT network limitations is given through time and space complexity analysis. Accordingly, typical machine learning algorithms are associated with a selected number of IoT-oriented datasets.

The proposed framework, including several components, is depicted in Figure 1. As seen, the first component consists of benchmark datasets enclosing labeled traffic flows and their related attributes. The considered datasets are detailed in Section 5. Then, the data are fed into the pre-processing component, which incorporates several critical mechanisms for a high-quality learning process. Specifically, four main steps are deployed. Namely, cleaning, transformation, normalization, and feature selection are conducted. Data cleaning includes the removal of "NaN", duplicate, and noisy values, while data transformation targets the categorical features and represents them using a numerical format. Data normalization solves the issue of having features on different scales and putting them in a range with a mean of 0 and a standard deviation of 1. Finally, feature selection is achieved using the SelectKBest method to identify the relevant features.

Next, the dataset is split into training and test sets at 70% and 30%, respectively. According to the related surveys and review papers, a set of ML techniques have been adopted due to their effectiveness in the security domain. These techniques have demonstrated good and effective performance when used for IDS design in the IoT environment. They include the following:

- Decision tree (DT): A family of tree-based algorithms used for both classification and regression problems. They apply certain rules inferred from the data features to decide the value of newly examined inputs. The trees are split into several branches and leaves based on the provided rules [1,61].
- Random forest (RF): A family of tree-based algorithms used for training an army of trees instead of one. These trees are built up randomly from different training subsets for both classification and regression tasks. They are scalable and can handle large datasets, minimizing the overall generalization error with high accuracy [1,61].
- Support vector machine (SVM): A family of non-probabilistic algorithms used for both classification and regression problems. It performs prediction by identifying the dividing hyper-plane that separates the inputs with the maximum margin. Both binary and multi-class systems can be handled and classified into a suitable dimensional space. SVM performs both linear and non-linear classification using a kernel trick [1,61].
- Naïve Bayes (NB): A family of probabilistic classification algorithms based on Bayes' theorem for both binary and multi-class inputs. Naïve represents an oversimplified assumption so as to calculate the probabilities of attributes. They are assumed to be conditionally independent and can be high-dimensional attributes [1,61].
- K nearest neighbor (KNN): A family of clustering algorithms used for both classification and regression problems. It works on combining new unseen datapoints with

similar given points by searching the closest neighbors in the feature space of the available dataset. A distance metric is used to find the K nearest neighbors, such as Euclidean distance, L∞ norm, and others [1,61].

- Multi-layer perceptron (MLP): A family of basic artificial neural networks consisting of three layers: input, hidden, and output layers, which allows for solving both linearly separable and non-linearly separable problems. It is a feed-forward neural network that connects neurons in a forward manner without having loops. An activation function and a set of weights are employed, whereby the weights can be fine-tuned using a supervised learning technique called backpropagation [62].

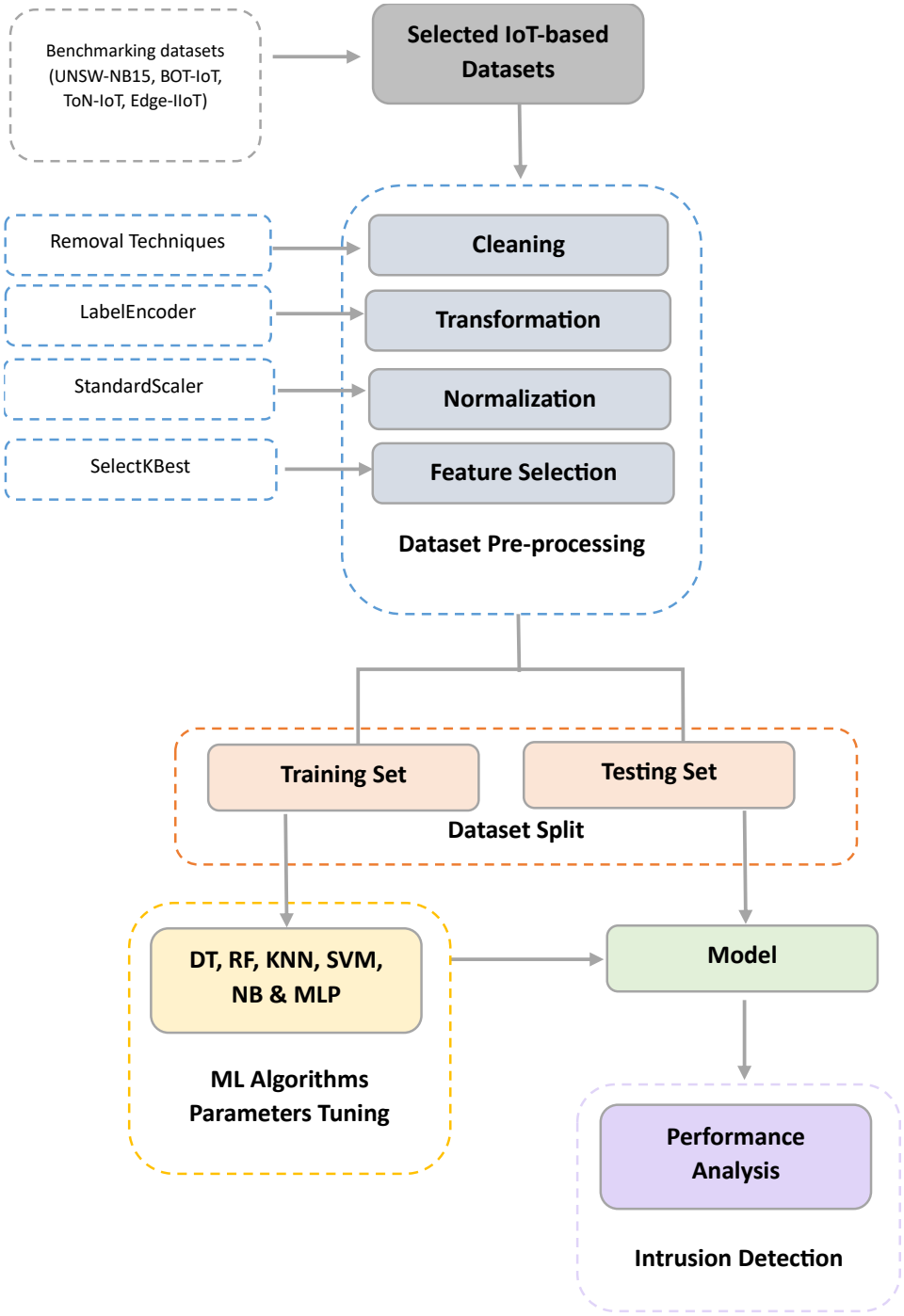

Figure 1. The proposed methodology.

Since an exhaustive search of the best settings of the candidate models' parameters is impractical, the GridSearch [63] method is adopted for model hyper-parameter tuning. In fact, GridSearch can be introduced as a generic technique used for hyper-parameter optimization and configuration at distinct levels. Specifically, it performs a complete search over the space of the hyper-parameter subset in accordance with the chosen ML algorithm. In particular, a boundary specification is needed to limit the values for each hyper-parameter of interest, followed by training and testing models for such parameters. Accordingly, the model is trained using the training set, which incorporates the hyper-parameter specifications set by the GridSearch method. Finally, a performance analysis is conducted on the test set to measure the effectiveness of the quality assurance techniques adopted by the multi-class ML-model-based IDS system.

To sum up, the proposed framework considers the essential quality factors that affect detection performance. Those factors are driven by the perspectives of the data and model, such as data source, data completeness, data diversity, model choice, feature space size, and others. They can be employed to optimize ML-based IDS behavior, taking into account IoT-relevant constraints.

5. Experiments

5.1. Datasets

Several datasets are used by the research community in the context of IDSs to evaluate the overall performance of the considered ML models. However, some of these datasets lack considerable information and need to incorporate new emerging attack types. Well-known datasets, such as KDD'99 [31], or similar sets are developed specifically for a wired network environment that is not intended for the IoT ecosystem. Thus, the creation of optimized IDSs targeting the IoT is not possible using these datasets. Accordingly, benchmark datasets are required that are modern, context-related, and updated with the latest novel attacks for the sake of better performance [31,64].

For this research, four publicly available IoT-based IDS datasets were considered. These datasets were recently released for IoT applications, including a suitable amount of traffic. They represent real or simulated network traffic for academic purposes that were compiled according to the proposed criteria mentioned by Gharib et al. [50]. For classification purposes, all four datasets are divided into 70% training and 30% testing sets to avoid overfitting and ensure proper generalization. A summary of all four datasets is reported in Table 3.

Table 3. Overview of the datasets chosen for this research.

Dataset	Year	Attacks	Features	Total Records	IoT Devices
UNSW-NB15 [38]	2015	10 attacks	45	2 million	NA
BOT-IoT [18]	2019	5 attacks	43	72 million	Simulated
ToN-IoT [35]	2020	9 attacks	31	22 million	Simulated
Edge-IIoT [36]	2022	14 attacks	61	20 million	More than 10 devices

5.1.1. UNSW-NB15 Dataset

The UNSW-NB15 dataset [65] is frequently adopted by research works for IDSs. The dataset was released by the Cyber Range Lab of the Australian Center for Cyber Security (ACCS) in 2015. One should mention that the PerfectStorm tool was utilized to create benign traffic. On the other hand, nine attack scenarios were adopted, including DoS, generic, reconnaissance, fuzzes, shellcode, exploits, analysis, worms, and backdoors. For feature extraction, the Argus [65] and Bro-IDS [65] tools were employed to extract 49 network traffic features. The dataset is not dedicated to IoT-based IDSs, but it has gained wide popularity in that context. It exhibits a class imbalance of 2,218,761 and 321,283 benign and malicious traces, respectively [65]. With regard to the compliance criteria introduced in [50], the 11 arguments are evaluated and presented in Table 4.

Table 4. Overview of the UNSW-NB15 dataset [65].

Criteria	Complied	Criteria	Complied
Complete Network Configuration	Yes	Attack Diversity	10
Complete Traffic	Yes	Anonymity	No
Labeled Dataset	Yes	Heterogeneity	Yes
Complete Interaction	Yes	Feature set	45
Complete Capture	Yes	Metadata	No
Available Protocols	TCP, UDP, ICMP and others		

In these datasets, several pre-processing steps are performed for quality assurance purposes: (a) combining the training and test datasets due to having some different attack category labels, followed by splitting it into training and testing sets later; (b) the removal of several values, such as NaN values, those classes with few instances, and duplicated rows within the classes; (c) the transformation of categorical features, such as encoding protocol type, services, states, and labels; (d) selecting the best features using SelectKBest [58] with K = 30; and (e) oversampling minor classes that are represented by malicious traffic and that are also bootstrapped to overcome the issue of imbalanced class records. A summary of the applied pre-processing steps is presented in Table 5.

Table 5. Pre-processing results for UNSW-NB15 dataset.

Method	Examples of Results	
	Before	After
Combining training and test datasets	Training dataset = 82,331 records Test dataset = 175,340 records	Combined dataset = 257,673 records
Removal of NaN values	Rows = 257,673	Rows= 257,673
	Columns = 45	Columns = 43
Transformation of categorical features (Ex: Attack category values)	'Analysis',	0
	'Backdoor',	1
	'DoS',	2
	'Exploits',	3
	'Fuzzers',	4
	'Generic',	5
	'Normal',	6
	'Reconnaissance',	7
	'Shellcode',	8
	'Worms'	9
Selection of best features	All	SelectKBest K = 30
Removal of duplicate values	Class 0 len: 2031	Class 0 len: 450
	Class 1 len: 1879	Class 1 len: 350
	Class 2 len: 5499	Class 2 len: 3623
	Class 3 len: 27,433	Class 3 len: 25,548
	Class 4 len: 20,954	Class 4 len: 19,044
	Class 5 len: 7599	Class 5 len: 7324
	Class 6 len: 85,027	Class 6 len: 85,024
	Class 7 len: 9991	Class 7 len: 9991
	Class 8 len: 1456	Class 8 len: 1456
	Class 9 len: 171	Class 9 len: 171

5.1.2. BOT-IoT Dataset

The BOT-IoT dataset [18] was developed for IoT networks in the Research Cyber Range lab of UNSW Canberra. It includes real and simulated traffic for both benign and malicious traces generated by IoT-based botnets. In terms of traffic, the Node-red tool is employed for simulating the network activities of five types of IoT devices, such as garage doors, weather monitoring systems, refrigerators, lights, and thermostats. The attacks are categorized into DoS, DDoS, port scanning, OS fingerprinting, information theft, and Keylogging attacks. In addition, two types of network flows are used, including the parameters of real protocols and simulated ones. For the IoT environment, the Message Queuing Telemetry Transport (MQTT) protocol is used, as it is a lightweight messaging protocol. More than 72 million records make up the dataset, indicating several attack types with a size of 16.7 GB in the CSV format. One should note that a smaller version of the dataset, including around 3 million records, was also used in previous works [18,66]. With regard to the compliance criteria provided in [50], the 11 arguments are evaluated and presented in Table 6.

Table 6. Overview of the BOT-IoT dataset [18].

Criteria	Complied	Criteria	Complied
Complete Network Configuration	Yes	Attack Diversity	5
Complete Traffic	Yes	Anonymity	Yes
Labeled Dataset	Yes	Heterogeneity	Yes
Complete Interaction	Yes	Feature set	43
Complete Capture	Yes	Metadata	No
Available Protocols	TCP, MQTT, ARP, IGMP, and others		

Only the 10 best feature subsets of the BOT-IoT dataset were employed for the experiments. This subset is composed of only 10 out of 43 independent features. It was developed by the UNSW research team and was derived by calculating the correlation coefficient and joint entropy for the total number of features [67]. In this dataset, several pre-processing steps are performed for quality assurance, such as the pre-processing steps in Section 5.1.1.

5.1.3. ToN-IoT Dataset

The ToN-IoT dataset [68] is a heterogeneous collection of data with multiple network traffic features published by the Cyber Range Lab of the Australian Centre for Cyber Security (ACCS) in 2019. The dataset is typically used to connect several resources, such as cloud layers, edges, blur, virtual machines, and physical systems. It contains several features based on the adopted environments, such as the IoT, IIoT-connected devices, operating system logs, and system network traffic. The traffic type and attack type in the case of malicious traffic are examples of such features. Several attack scenarios were found, such as injection, ransomware, DoS, distributed DoS (DDoS), man-in-the-middle (MITM), cross-site scripting (XSS), backdoors, password cracking, and scanning attacks. With regard to the compliance criteria identified in [50], the 11 arguments are evaluated and presented in Table 7.

The ToN-IoT dataset is employed for experimental design. It contains several subsets: IoT, Linux, Windows 7, Windows 10, and Network subsets with different feature sets [68]. For the sake of consistency, only the network subset was employed for similar feature sets with the other aforementioned datasets. Accordingly, several pre-processing steps are performed for quality assurance purposes, such as the pre-processing steps in Section 5.1.1.

Table 7. Overview of the ToN-IoT dataset [68].

Criteria	Complied	Criteria	Complied
Complete Network Configuration	Yes	Attack Diversity	9
Complete Traffic	Yes	Anonymity	Yes
Labeled Dataset	Yes	Heterogeneity	Yes
Complete Interaction	No	Feature set	IoT: 44 Linux: 36 Windows 7: 135 Windows 10: 127 Network: 46
Complete Capture	Yes	Metadata	Yes
Available Protocols	TCP, UDP, DNS, HTTP, SSL, and others		

5.1.4. Edge-IIoT Dataset

The Edge-IIoT dataset [69] is a newly released IoT and IIoT dataset intended to support research in the cybersecurity field. It represents real traffic that is generated from more than 10 types of IoT devices. Such devices include digital sensors that are low-cost for sensing temperature and humidity, soil moisture sensors, ultrasonic sensors, heart rate sensors, flame sensors, water level detection sensors, pH sensors, and others. The Edge-IIoT dataset encompasses both benign and malicious traffic. The latter covers around 14 attack scenarios related to IoT services. These scenarios are grouped into five categories of threats: DoS and DDoS attacks, man-in-the-middle (MITM) attacks, information gathering, injection attacks, and malware attacks [26]. With regard to the compliance criteria introduced in [50], the eleven arguments are evaluated and presented in Table 8.

Table 8. Overview of Edge-IIoT dataset [69].

Criteria	Complied	Criteria	Complied
Complete Network Configuration	Yes	Attack Diversity	14
Complete Traffic	Yes	Anonymity	Yes
Labeled Dataset	Yes	Heterogeneity	Yes
Complete Interaction	No	Feature set	61
Complete Capture	Yes	Metadata	Yes
Available Protocols	TCP, ARP, DNS, ICMP, HTTP, and others		

The ML-Edge-IIoT dataset is employed for experimental design. The subset is composed of 61 features and was nominated to be used for evaluating conventional ML algorithms [69]. In the selected datasets, several pre-processing steps are performed for quality assurance purposes, as mentioned for the UNSW-NB15 dataset.

5.2. Performance Measures

Several performance measures have been used in the literature to tackle the model evaluation task. Specifically, five standard metrics, namely, accuracy, precision, recall, *F*1-*measure*, and AUC score, are derived indirectly from a confusion matrix [14]. In fact, the confusion matrix includes four main values: true positive (*TP*), true negative (*TN*), false positive (*FP*), and false negative (*FN*). *TP* and *TN* represent the successful classification of

benign and malicious inputs, respectively. The misclassified benign and malicious inputs can be represented by *FP* and *FN*, respectively.

Accuracy represents one of the derived metrics that measures the detection capability of a system to identify benign and malicious inputs. It can be mathematically expressed as follows:

$$Accuracy = \frac{TP + TN}{TP + FN + TN + FP} \times 100 \quad (1)$$

Precision identifies the percentage of positive predictions that are correctly classified. It can be mathematically expressed as follows:

$$Precision = \frac{TP}{TP + FP} \times 100 \quad (2)$$

Recall can be defined as the percentage of positive predictions over all the positive instances. It shows the ability of IDSs to correctly detect malicious inputs and can be mathematically expressed as follows:

$$Recall = \frac{TP}{TP + FN} \times 100 \quad (3)$$

However, precision and recall lack the utilization of *TN* and, thereby, neglect the correct classification of negative inputs within the overall IDS detection rate.

In terms of the remaining metrics, the *F1-measure* gives the harmonic mean between *recall* and *precision*. The macro version of this measure is not affected by class weight since all the classes are weighted equally. It helps to handle data quality issues, such as data imbalance problems.

$$F1\text{-}measure = 2 \times \frac{Precison \times Recall}{Precison + Recall} \quad (4)$$

Finally, the AUC score computes the area under the ROC curve to determine the prediction performance between both the normal and intrusion classes when compared to random prediction [16].

To sum up, the best model performances can be achieved when obtaining the highest scores in all the five measures: F1, accuracy, precision, recall, and AUC score, but with the lowest scores in FPR and FNR. FPR reflects the number of benign instances that are incorrectly classified as intrusion. FNR reflects the number of intrusion instances that are incorrectly classified as normal. As such, an IDS with a high FPR represents a weakness in identifying the actual characteristics of the normal class. However, an IDS with a high FNR is more threatening and can cause more damage to the network environment [1,16].

5.3. Results

In the following, we report the results and findings obtained using the four main benchmarking datasets described in the previous section. The traffic flow features of the datasets were considered for their increasing use because of being more resource-efficient and suitable to the ML detection task [70]. The detailed results obtained using the five pre-defined ML algorithms are reported in particular. These algorithms are chosen to reduce the computational limitations associated with IoT devices while achieving better performance [71]. ML algorithms are employed to perform multi-class classification with several pre-processing steps. Such steps include removing missing values, removing NaN values, removing duplicate records, encoding categorical features, feature selection, and feature normalization. This is followed by a classification task over the complete datasets. The supervised ML classifiers are evaluated using Equations (1)–(4), where Equation (1) is used to measure the overall performance. Accordingly, both the training and testing set are evaluated in terms of forecasting the accuracy of the classifiers. Such evaluation can investigate potential overfitting, underfitting, and time-consuming model configuration.

5.3.1. Classification Results Obtained Using Pre-Processed UNSW-NB15 Dataset

The UNSW-NB15 dataset was employed in this experiment to first evaluate the candidate models' performance in a generic IDS scenario. A multi-class classification is performed, covering 10 types of attack classes. Figure 2 shows the class distribution over the dataset instances. In terms of performance evaluation, the dataset is partitioned into 80% training and 20% testing sets. For quality assurance, several feature-based mechanisms are applied to enhance detection performance. Feature conversion, feature normalization, and feature selection were employed by using LabelEncoder, StandardScaler, and SelectKbest, respectively. Only a subset of the 30 best features was used in the performance analysis.

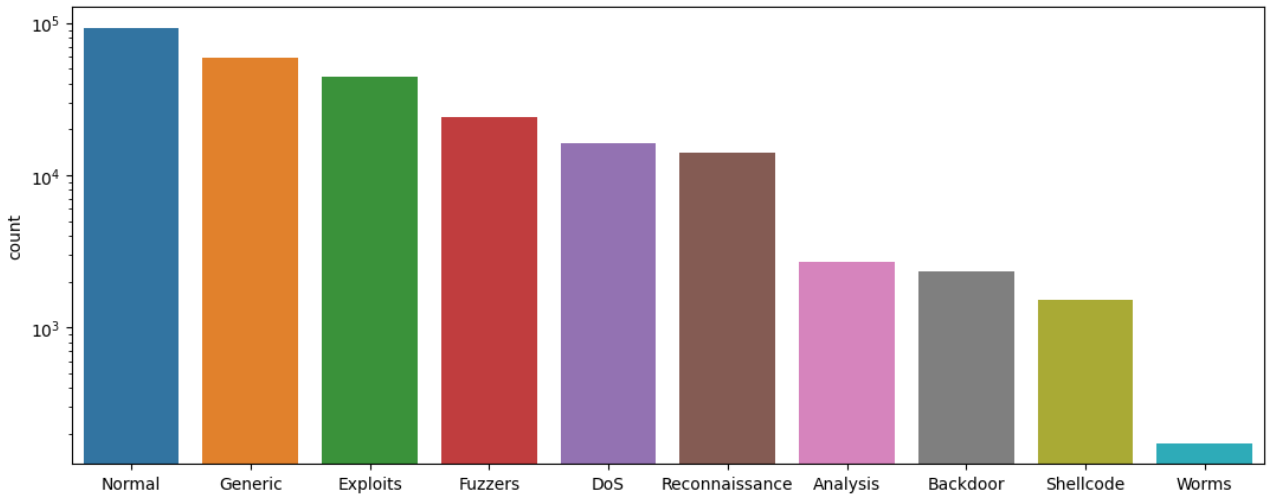

Figure 2. Class distribution for the UNSW-NB15 dataset.

In terms of performance evaluation, the pre-mentioned performance measures were utilized to evaluate the detection rate. The reported results show a variation specifically for NB, which yielded only 26% accuracy. The other models' performance ranged from 77% to 86% for DT and KNN, respectively. Moreover, the accuracy of extremely rare classes, such as Analysis and Backdoor, is very low compared to the dominant classes, like Normal and Generic. The top accuracy of 89% was achieved by the MLP model. The details of the performance of each label classification can be seen in Table 9.

Table 9. Performance results obtained by related works using the UNSW-NB15 dataset.

Research	Year	ML Technique	Accuracy	Feature Selection	Class
Kanimozhi et al. [19]	2019	**ANN** RF-DT	**89.00%** 75.62%	Recursive Feature Elimination (RFE)	Multi
Kasongo et al. [22]	2020	**ANN** LR KNN SVM DT	**77.51%** 65.29% 72.30% 61.53% 67.57%	XGBoost-based feature selection	Multi
Yin et al. [28]	2023	**MLP**	**84.24%**	Information Gain (IG) and Random Forest (RF)	Multi
Moustafa et al. [29]	2016	**ANN**	**81.34%**	Chi-square $\left(Chi^2\right)$	Multi
The proposed method	2023	DT RF KNN SVM NB **MLP**	77.44% 79.50% 86.16% 84.02% 26.08% **89.00%**	ANOVA F-value	Multi

Moreover, several research works utilized the UNSW-NB15 dataset to measure the IDS detection rate. Accordingly, a comparison of these research works using the proposed model was also conducted. The results reported in Table 9 reflect the superiority of all the proposed conventional ML models in classifying multi-class attacks when compared to other studies. However, some research studies achieved higher accuracy since they adopted deep learning (DL) models. DL models require high implementation costs in terms of memory and computational power, which is very challenging in IoT resource-constrained devices [71]. Moreover, MLP obtained the highest accuracy when compared to the same conventional ML types of models, either from the results of other works or the proposed candidate models.

5.3.2. Classification Results Obtained Using the Pre-Processed BOT-IoT Dataset

The BOT-IoT dataset is utilized for its specificity with IoT networks. It contains several subsets; only the 10 best feature subsets were adopted, including its training and testing collections. The same candidate models were evaluated on a five attacks classification problem. Figure 3 shows the class distribution for the BOT-IoT dataset.

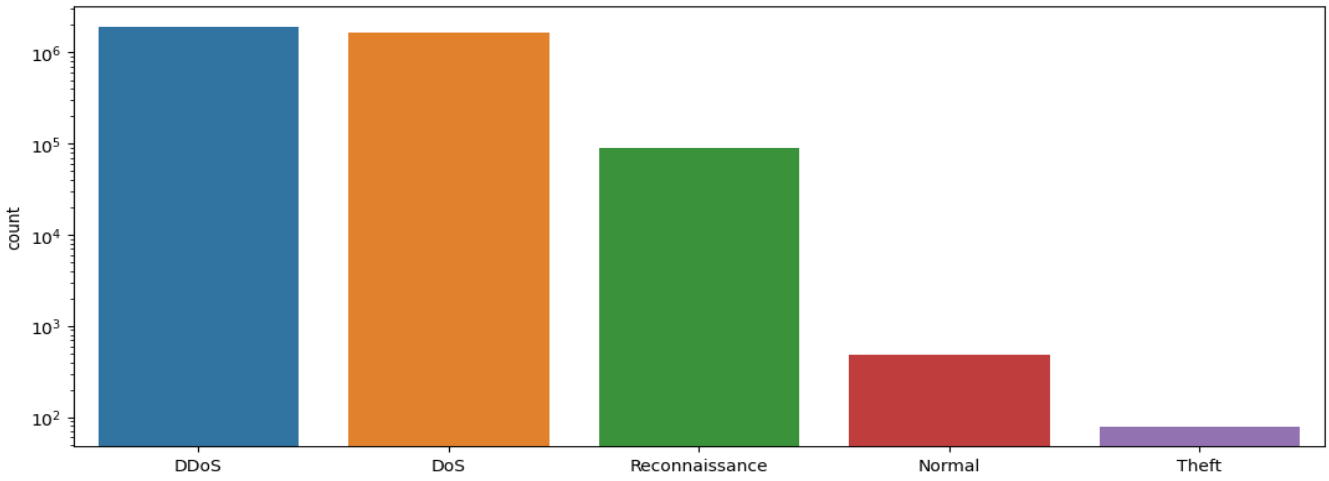

Figure 3. Class distribution for the BOT-IoT dataset.

For evaluation purposes, the dataset was split into 70% and 30% for the training and testing subsets, respectively. In terms of pre-processing, the feature-based mechanisms were used to ensure better detection rates. This step covers feature conversion, feature normalization, and feature selection. Accordingly, LabelEncoder, StandardScaler, and SelectKbest were applied based on the proposed framework. It is worth noting that only 10 of the best features were used in the classification task.

In terms of performance evaluation, the six candidate models were used to address the attack detection problem and, thereby, the detection rate. The reported results show minor variations for the NB and SVM classifiers, which yielded 71% and 89% accuracy, respectively. They achieved the lowest accuracy when compared to the detection rates of other models, which ranged from 98% to 99%. The best accuracy of 99.94% was achieved by the KNN model.

In terms of the comparative analysis, multiple research works have employed the BOT-IoT dataset for evaluating their proposed IDS. As such, a comparison between these research works, and the proposed model was conducted. As depicted in Table 10, the reported results reflect the superiority of our candidate ML models in classifying multi-class attacks when compared to other studies. Moreover, KNN gave the highest accuracy compared to the same conventional ML types in either the results of other works or from the proposed candidate models.

Table 10. Results achieved by related works using the BOT-IoT dataset.

Research	Year	ML Technique	Accuracy	Feature Selection	Class
Koroniotis et al. [18]	2019	SVM **RNN** **LSTM**	88.3% **99.7%** **99.7%**	Correlation Coefficient	Multi
Thaseen et al. [23]	2020	C4.5 NB RF MLP SVM CART KNN ANN **Ensemble Learning**	92.0% 87.6% 92.7% 87.4% 89.5% 80.3% 88.4% 97.0% **99.0%**	Correlation Coefficient	Multi
Sugi et al. [24]	2020	**LSTM** KNN	**77.51%** 92.29%	Information Gain (IG)	Multi
Fatani et al. [27]	2021	**CNN**	**99.47%**	Transient Search Optimization (TSO)	Multi
The proposed method	2023	DT RF **KNN** SVM NB MLP	99.88% 99.89% **99.94%** 89.00% 71.00% 98.00%	ANOVA F-value	Multi

5.3.3. Classification Results Obtained Using the Pre-Processed ToN-IoT Dataset

The pre-mentioned ToN-IoT dataset consists of several subsets, where the network subset was selected for the sake of consistency with the other benchmarking datasets. A multi-class classification scenario was performed covering 10 types of target classes. Figure 4 shows the class distribution over the dataset instances.

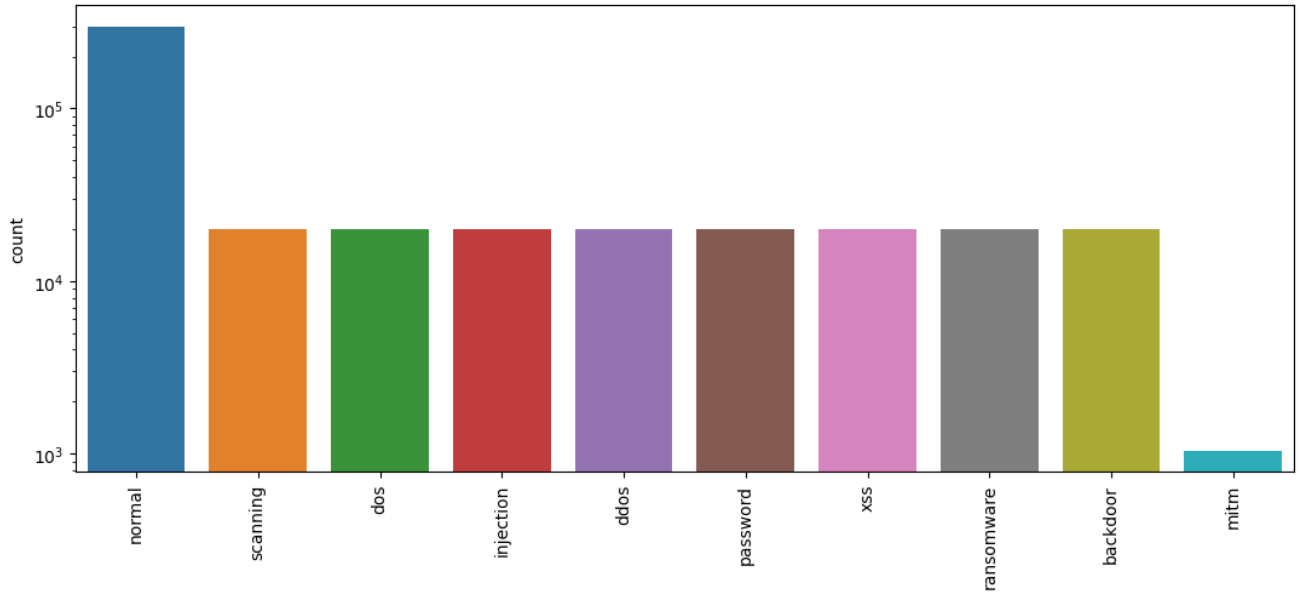

Figure 4. The distribution of the ToN-IoT dataset for 10 attack classes.

The dataset was divided into 80% training and 20% testing prior to the performance evaluation. Feature conversion and normalization using LabelEncoder and Standard-

Scaler were applied, respectively. Their needs come from the variety of dataset variables found in the dataset with different scales. Moreover, only the subset from the 30 best features achieved using the SelectKbest feature selection method was used for the classification tasks.

In terms of performance evaluation, several competing results were achieved by the selected models, ranging from 98% to 99% accuracy, except for SVM with 87%. The highest accuracy of 99.97% was achieved by the MLP classifier, followed by RF and KNN, with slight differences. The lowest accuracy of 86% was obtained by a linear kernel SVM.

Several pieces of research that investigated the ToN-IoT dataset were considered for a comparative analysis, which was conducted to assess the proposed model's performance. The results reported in Table 11 reflect the superiority of all the proposed models in classifying multi-class attacks when compared to other studies. Moreover, MLP yielded the highest accuracy compared to the other research works, as well as the remaining candidate models.

Table 11. Results obtained using the related works and the ToN-IoT dataset.

Research	Year	ML Technique	Accuracy	Feature Selection	Class
Tareq et al. [12]	2022	DenseNet **Inception Time**	98.57% **99.65%**	-	Multi
Sarhan et al. [20]	2021	DFF **RF**	96.58% **97.49%**	Chi-square $\left(Chi^2\right)$	Multi
Alsaedi et al. [21]	2020	LR LDA KNN RF **CART** NB SVM LSTM	61% 62% 72% 71% **77%** 54% 60% 68%	-	Multi
Gad et al. [30]	2021	LR NB DT RF AdaBoost KNN SVM **XGBoost**	85.9% 69.2% 97.2% 97.2% 90.6% 98.2% 86.0% **98.3%**	Chi-square $\left(Chi^2\right)$	Multi
The proposed method	2023	DT RF KNN SVM NB **MLP**	98.06% 99.95% 99.87% 87.22% 99.39% **99.97%**	ANOVA F-value	Multi

5.3.4. Classification Results Obtained Using the Pre-Processed Edge-IIoT Dataset

The pre-mentioned Edge-IoT dataset contains multiple subsets, whereby a subset, namely the "ML-EdgeIIoT" dataset, is chosen for the sake of the type of the candidate models. In particular, a multi-class classification scenario targeting 15 types of classes was conducted to assess the performance of the candidate models. Figure 5 shows the class distribution over the dataset instances. The dataset is divided into 80% training and 20% testing subsets prior to the pre-processing phase. Due to the variety of the feature scales, feature conversion and normalization using LabelEncoder and StandardScaler were applied, respectively. Moreover, the SelectKbest feature selection method was adopted, yielding the best features for enhancing the detection rate.

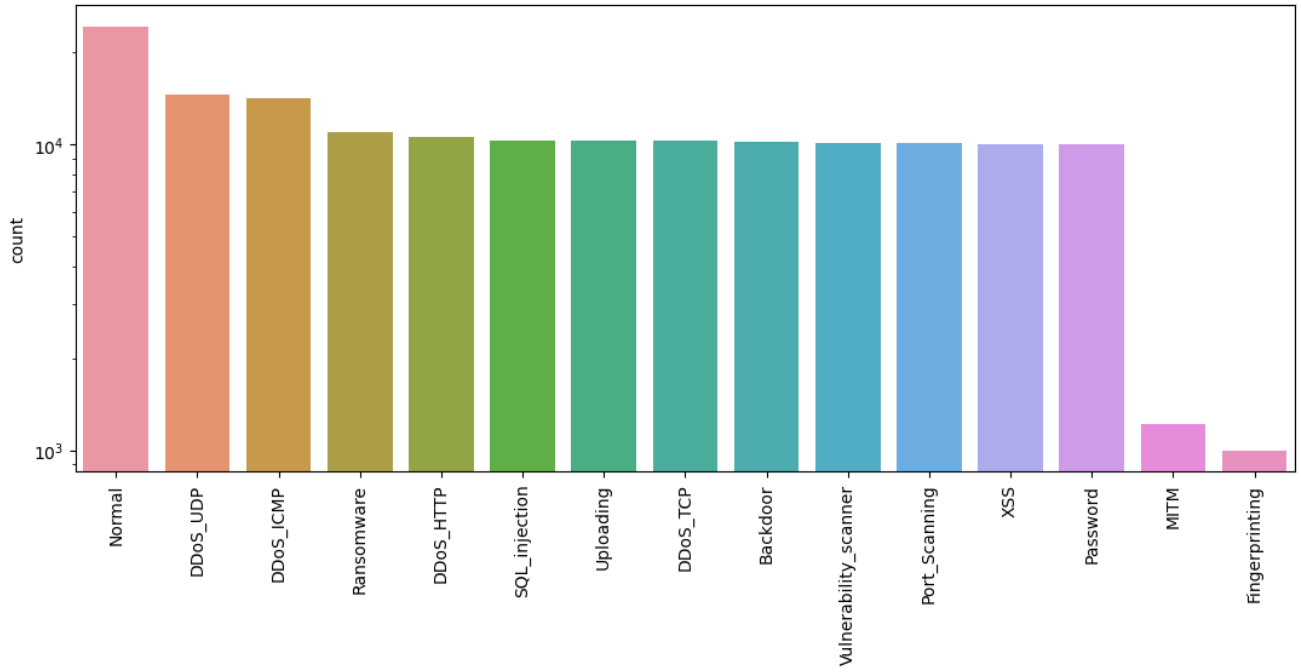

Figure 5. Class distribution of the Edge-IIoT dataset, including 15 attack classes.

For evaluation purposes, four performance measures were employed: the accuracy, precision, recall, and F1-measure. The reported results show very competitive performance, ranging from 95% to 99%. The best accuracy reached 99.97% for the RF classifier, followed by DT, MLP, and KNN, with slight differences. The lowest accuracy values were 97% and 95.01% for SVM and NB, respectively.

Table 12 shows a comparison between the proposed methods and the related studies. The tabulated results reflect the superiority of all proposed models in classifying multi-class attacks when compared to the results obtained using the related research. As seen, RF achieved the highest accuracy when compared to the related works, as well as the remaining candidate models.

Table 12. Results obtained using related works and the Edge-IIoT dataset.

Research	Year	ML Technique	Accuracy	Feature Selection	Class
Ferrag et al. [26]	2021	DT RF SVM KNN **DNN**	67.11% 80.83% 77.61% 79.18% **94.67%**	Generic Model (RF)	Multi
Tareq et al. [12]	2022	**Inception Time**	**94.94%**	-	Multi
The proposed method	2023	DT **RF** KNN SVM NB MLP	99.87% **99.97%** 99.65% 97.00% 95.01% 99.87%	ANOVA F-value	Multi

6. Discussion

In this study, several ML techniques were adapted to assess IDS performance and model the traffic flow in the IoT environment. The candidate models include DT, RF, KNN, SVM, NB, and MLP. They were employed in the context of a multi-class classification scenario. In addition, four benchmarking datasets, namely, UNSW-NB15, BOT-IoT, ToN-IoT, and Edge-IIoT, were selected based on the criteria proposed by Gharib et al. [50].

For IDS quality improvement, several data pre-processing techniques were applied. Specifically, the instance-based and feature-based approaches were investigated. Instance-based pre-processing deals with data cleaning and the corresponding removal techniques. On the other hand, feature-based pre-processing includes feature transformation, normalization, and dimensionality reduction through proper feature selection. Accordingly, several records were removed from the selected datasets. The removed records contain NaN, duplicate, network flow identifiers, and noisy values. In addition, feature transformation was applied to all the categorical features of the selected datasets to obtain numerical representations of the datasets. On the other hand, encoding is performed over multiple important features, such as encoding protocol type, services, states, attack categories, attack subcategories, network protocols, DNS-related features, HTTP-related features, SSL-related features, and so on. Feature normalization was applied to all instances by using the standardization method StandardScaler with outputs with a mean of 0 and standard deviation of 1. This was performed in conjunction with dimensionality reduction for all datasets, including 30 to 40 features. The latter was carried out using SelectKBest [58], which relies on the ANOVA F-measure. Such feature selection contributes to reducing the feature vector by selecting the most relevant features. An additional condition is applied to the selection with a defined threshold to filter out less important feature selections. For instance, the feature vector of the ToN IoT dataset was reduced from 44 to 30 and then minimized further to 14 only. These methods are proven to enhance overall IDS detection and reduce complexity after several iterations of experiments.

Another aspect of IDS quality improvement was proposed to enhance the performance of the candidate models. For model parameter tuning, the GridSearch method [63] was applied to find the optimal parameter values for better detection performance. This method has drastically improved the results, such as with the increment of the tree-based methods, where performance was around 9% higher. The four metrics: accuracy, precision, recall, and F1-measure, were also used to measure the efficiency of the models with regard to individual label detection, as well as overall detection. Different attacks were correctly classified, whereas in some cases, models underperformed in terms of minority classes, such as Backdoor and Worms in UNSW-NB15.

Based on the obtained results, one can claim that MLP is the best-performing classifier on the four benchmarking datasets. It achieved 86% accuracy on UNSW-NB15, while it performed even better on the other datasets with 99% accuracy. Overall, all the candidate models yielded satisfactory results using all dedicated IoT-based datasets. On the other hand, they proved to be less efficient for UNSW-NB15. Tree-based techniques are the easiest methods in terms of implementation since their interpretation is obvious using conditional rules. Moreover, the results achieved within the proposed framework are superior when compared to those obtained in [12,18–22,26–28,72]. A detailed comparison of the results obtained using the related works and each dataset for the multi-class classification task are reported in Tables 9–12.

The confusion matrices obtained using the best-performing models, along with all four benchmarking datasets, are presented in Figure 6. As can be seen, the diagonals reflect the performance of each best model with respect to each class in the different datasets. The darker the square becomes, the higher the achieved accuracy is of a given class.

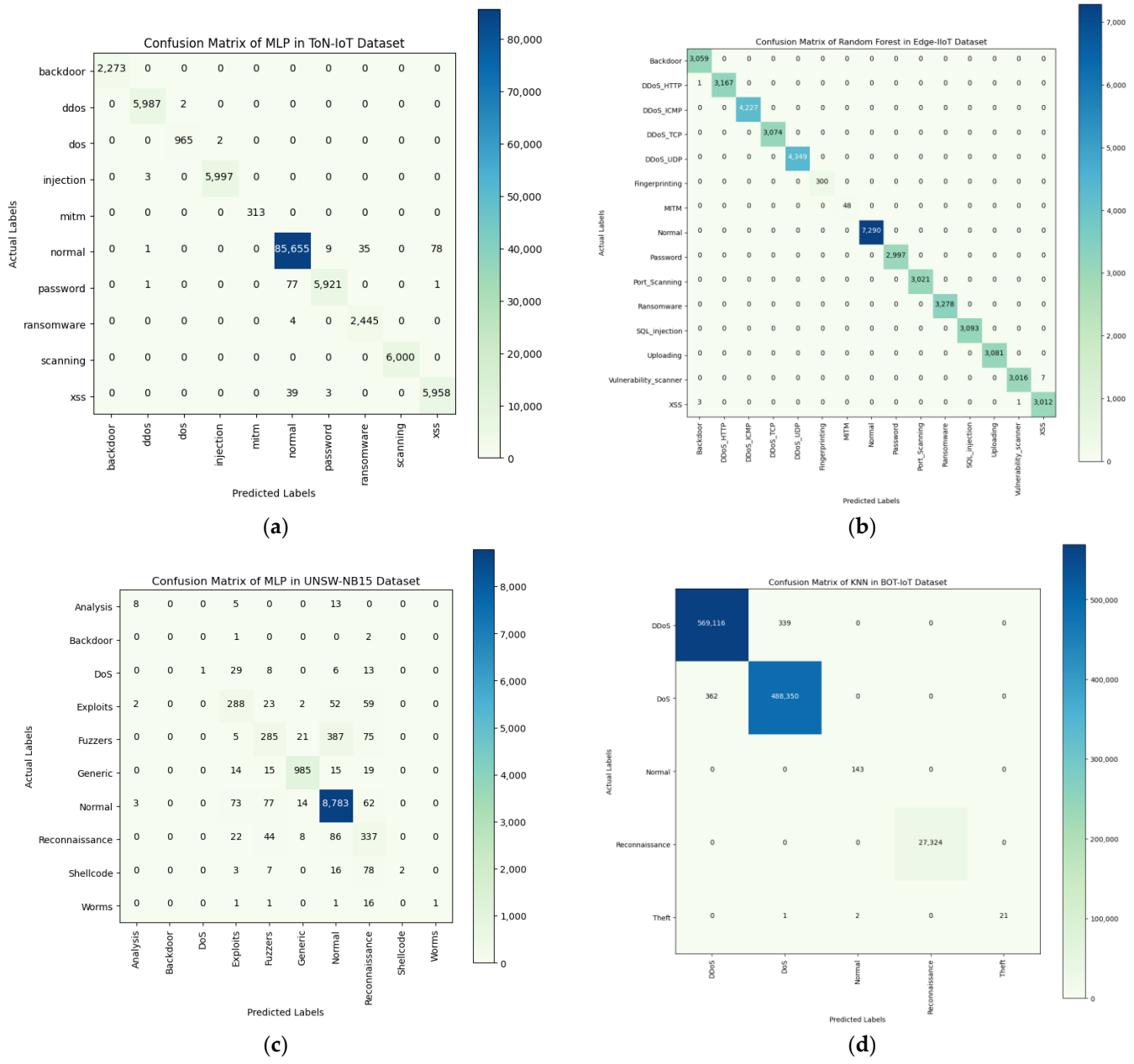

Figure 6. The confusion matrices obtained using the best performing models: (**a**) MLP and ToN-IoT datasets; (**b**) RF and Edge-IIoT datasets; (**c**) MLP and UNSW-NB15 datasets; (**d**) and KNN and BOT-IoT datasets.

7. Conclusions and Future Works

Day by day, the IoT environment is exposed to an increasing number of cyber-attacks. Enhancing the performance of IDSs contributes toward the better protection of IoT cross-platform applications and connected devices. Various researchers have devoted efforts to developing a secure, lightweight framework for the IoT using ML techniques. Two main elements play an important role in assuring the quality of ML-based IDSs: dataset pre-processing and model generalization capability. The major contribution of this work consists of designing a framework for quality ML-based IDSs, taking into consideration IoT environment specifications.

Several dimensions relevant to both data quality and model quality were explored for more accurate intrusion detection. Accordingly, several experiments were conducted on four benchmarking datasets using six different classification techniques, namely decision

tree, random forest, k-nearest neighbors, support vector machine, naive Bayes, and multi-layer perceptron. The considered algorithms were assessed using datasets, including UNSW-NM15, BOT-IoT, ToN-IoT, and Edge-IIoT, in a multi-class classification scenario. Moreover, a review of each considered dataset was performed according to the criteria recommended by Gharib et al. [50]. The performance of the proposed framework was assessed using precision, recall, F1-measure, and accuracy. The reported results proved the superiority of the MLP technique when using both the UNSW-NB15 and ToN-IoT datasets. When using the relevant pre-processing techniques, MLP and RF achieved the highest accuracy of 99.97% using a selected set of features on the ToN-IoT and Edge-IIoT datasets, respectively. Moreover, KNN yielded superior performance when associated with the BOT-IoT dataset, reaching 99.94% accuracy. Overall, there was an increase in terms of accuracy for all proposed models using the four benchmarking datasets when compared to related works.

Based on the obtained results, one can claim that (i) data pre-processing techniques, such as transformation and normalization, enhance the detection rate significantly; (ii) dimensionality reduction and/or feature selection represent crucial pre-processing steps that require careful selection and yield a less complex model; (iii) hyper-parameter tuning should be conducted for all candidate models using the GridSerach method; and (iv) most of the candidate models achieve good performance on the datasets after applying the pre-processing steps. This reflects the importance of quality methods, such as data pre-processing and model tuning, for performance enhancement.

Regarding future work, more investigations would need to be carried out to incorporate techniques such as adversarial machine learning (AML) for quality improvement purposes. In fact, AML, as an emerging research field that has gained wide popularity in image processing, is still growing in terms of intrusion detection scenarios and, more specifically, in an IoT context. Particularly, AML deals with the generation of adversarial examples to measure detection performance when typical classifiers are deceived by such inputs. The consideration of AML methods for the proposed framework can contribute to hardening ML-based IDSs. Specifically, this can be achieved by associating adversarial attack scenarios and AML-based defensive mechanisms iteratively until the best overall detection performance is reached. In other words, designing a coherent framework that combines both attacks and defense mechanisms that reflect the "defense-in-depth" concept would represent a promising research direction to further improve the findings in this article. This would also yield an additional cost for the adversary in terms of knowledge, time, and computational resources to evade such a solution [39,72]. Moreover, AML techniques can be employed in data-driven modeling scenarios, representing an active learning method to target human actions in terms of unsafe behavior. The target action features are expressed in mathematical formalisms to ease the use of formal methods. Formal methods are works from the computer science field that formulate mathematical specifications, designs, and verifications to form proof obligations that are met by systems. This reflects the importance of formal methods to enable trustworthy AI-based system assurance and verification to achieve better accuracy and convergence in terms of prediction performance. Consequently, further investigation in such a field can be proposed for future considerations [73,74].

Author Contributions: Conceptualization, S.A., S.A.-A. and M.M.B.I.; methodology, S.A.; software, S.A.; validation, S.A. and M.M.B.I.; formal analysis, S.A.; investigation, S.A.; resources, S.A., S.A.-A. and M.M.B.I.; data curation, S.A.; writing—original draft preparation, S.A.; writing—review and editing, S.A., S.A.-A. and M.M.B.I.; visualization, S.A.; supervision, S.A.-A. and M.M.B.I.; project administration, M.M.B.I.; funding acquisition, S.A.-A. All authors have read and agreed to the published version of the manuscript.

Funding: The authors would like to thank the Deanship of Scientific Research at King Saud University for funding and supporting this research through the DSR Graduate Students Research Support (GSR) initiative.

Institutional Review Board Statement: Not applicable.

Data Availability Statement: All data has been present in main text.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Mahdavinejad, M.S.; Rezvan, M.; Barekatain, M.; Adibi, P.; Barnaghi, P.; Sheth, A.P. Machine learning for Internet of Things data analysis: A survey. *Digit. Commun. Netw.* **2018**, *4*, 161–175. [CrossRef]
2. Almseidin, M.; Alzubi, M.; Kovacs, S.; Alkasassbeh, M. Evaluation of machine learning algorithms for intrusion detection system. In Proceedings of the 2017 IEEE 15th International Symposium on Intelligent Systems and Informatics (SISY), Subotica, Serbia, 14–16 September 2017; pp. 277–282.
3. Zarpelão, B.B.; Miani, R.S.; Kawakani, C.T.; de Alvarenga, S.C. A survey of intrusion detection in Internet of Things. *J. Netw. Comput. Appl.* **2017**, *84*, 25–37. [CrossRef]
4. Capra, M.; Peloso, R.; Masera, G.; Ruo Roch, M.; Martina, M. Edge computing: A survey on the hardware requirements in the internet of things world. *Future Internet* **2019**, *11*, 100. [CrossRef]
5. Shukla, P. ML-IDS: A machine learning approach to detect wormhole attacks in Internet of Things. In Proceedings of the 2017 Intelligent Systems Conference (IntelliSys), London, UK, 7–8 September 2017; pp. 234–240.
6. Viegas, E.; Santin, A.; Oliveira, L.; Franca, A.; Jasinski, R.; Pedroni, V. A reliable and energy-efficient classifier combination scheme for intrusion detection in embedded systems. *Comput. Secur.* **2018**, *78*, 16–32. [CrossRef]
7. Canedo, J.; Skjellum, A. Using machine learning to secure IoT systems. In Proceedings of the 2016 14th Annual Conference on Privacy, Security and Trust (PST), Auckland, New Zealand, 12–14 December 2016; pp. 219–222.
8. Kim, J.; Kim, J.; Thu HL, T.; Kim, H. Long short term memory recurrent neural network classifier for intrusion detection. In Proceedings of the 2016 International Conference on Platform Technology and Service (PlatCon), Jeju, Republic of Korea, 15–17 February 2016; pp. 1–5.
9. Saeed, A.; Ahmadinia, A.; Javed, A.; Larijani, H. Intelligent intrusion detection in low-power IoTs. *ACM Trans. Internet Technol. (TOIT)* **2016**, *16*, 1–25. [CrossRef]
10. Zhang, J.; Qin, Z.; Yin, H.; Ou, L.; Zhang, K. A feature-hybrid malware variants detection using CNN based opcode embedding and BPNN based API embedding. *Comput. Secur.* **2019**, *84*, 376–392. [CrossRef]
11. Agarap, A.F. Towards Building an Intelligent Anti-Malware System: A Deep Learning Approach Using Support Vector Machine (SVM) for Malware Classification, No. 1. 2017. Available online: http://arxiv.org/abs/1801.00318 (accessed on 13 April 2022).
12. Tareq, I.; Elbagoury, B.M.; El-Regaily, S.; El-Horbaty, E.-S.M. Analysis of ToN-IoT, UNW-NB15, and Edge-IIoT Datasets Using DL in Cybersecurity for IoT. *Appl. Sci.* **2022**, *12*, 9572. [CrossRef]
13. Nesa, N.; Ghosh, T.; Banerjee, I. Non-parametric sequence-based learning approach for outlier detection in IoT. *Future Gener. Comput. Syst.* **2018**, *82*, 412–421. [CrossRef]
14. Khan, H.U.; Sohail, M.; Ali, F.; Nazir, S.; Ghadi, Y.Y.; Ullah, I. Prioritizing the Multi-criterial Features based on Comparative Approaches for Enhancing Security of IoT devices. *Phys. Commun.* **2023**, *59*, 102084. [CrossRef]
15. Mazhar, T.; Talpur, D.B.; Shloul, T.A.; Ghadi, Y.Y.; Haq, I.; Ullah, I.; Ouahada, K.; Hamam, H. Analysis of IoT Security Challenges and Its Solutions Using Artificial Intelligence. *Brain Sci.* **2023**, *13*, 683. [CrossRef] [PubMed]
16. Tran, N.; Chen, H.; Bhuyan, J.; Ding, J. Data Curation and Quality Evaluation for Machine Learning-Based Cyber Intrusion Detection. *IEEE Access* **2022**, *10*, 121900–121923. [CrossRef]
17. Si-Ahmed, A.; Al-Garadi, M.A.; Boustia, N. Survey of Machine Learning Based Intrusion Detection Methods for Internet of Medical Things. *arXiv* **2022**, arXiv:2202.09657. [CrossRef]
18. Koroniotis, N.; Moustafa, N.; Sitnikova, E.; Turnbull, B. Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-IoT dataset. *Future Gener. Comput. Syst.* **2019**, *100*, 779–796. [CrossRef]
19. Kanimozhi, V. Jacob PUNSW-NB15 dataset feature selection network intrusion detection using deep learning. *Int. J. Recent Technol. Eng.* **2019**, *7*, 443–446.
20. Nawir, M.; Amir, A.; Yaakob, N.; Lynn, O.B. Effective and efficient network anomaly detection system using machine learning algorithm. *Bull. Electr. Eng. Inform.* **2019**, *8*, 46–51. [CrossRef]
21. Alsaedi, A.; Moustafa, N.; Tari, Z.; Mahmood, A.; Anwar, A. TON_IoT telemetry dataset: A new generation dataset of IoT and IIoT for data-driven intrusion detection systems. *IEEE Access* **2020**, *8*, 165130–165150. [CrossRef]
22. Kasongo, S.M.; Sun, Y. Performance analysis of intrusion detection systems using a feature selection method on the UNSW-NB15 dataset. *J. Big Data* **2020**, *7*, 105. [CrossRef]
23. Thaseen, I.S.; Mohanraj, V.; Ramachandran, S.; Sanapala, K.; Yeo, S.S. A hadoop based framework integrating machine learning classifiers for anomaly detection in the internet of things. *Electronics* **2021**, *10*, 1955. [CrossRef]
24. Sugi, S.S.S.; Ratna, S.R. Investigation of machine learning techniques in intrusion detection system for IoT network. In Proceedings of the 2020 3rd International Conference on Intelligent Sustainable Systems (ICISS), Thoothukudi, India, 3–5 December 2020; pp. 1164–1167.
25. Sarhan, M.; Layeghy, S.; Portmann, M. Feature Analysis for Machine Learning-based IoT Intrusion Detection. *arXiv* **2021**, arXiv:2108.12732.

26. Ferrag, M.A.; Friha, O.; Hamouda, D.; Maglaras, L.; Janicke, H. Edge-IIoTset: A new comprehensive realistic cyber security dataset of IoT and IIoT applications for centralized and federated learning. *IEEE Access* **2022**, *10*, 40281–40306. [CrossRef]
27. Fatani, A.; Abd Elaziz, M.; Dahou, A.; Al-Qaness, M.A.; Lu, S. IoT intrusion detection system using deep learning and enhanced transient search optimization. *IEEE Access* **2021**, *9*, 123448–123464. [CrossRef]
28. Yin, Y.; Jang-Jaccard, J.; Xu, W.; Singh, A.; Zhu, J.; Sabrina, F.; Kwak, J. IGRF-RFE: A hybrid feature selection method for MLP-based network intrusion detection on UNSW-NB15 Dataset. *J. Big Data* **2023**, *10*, 15. [CrossRef]
29. Moustafa, N.; Slay, J. The evaluation of Network Anomaly Detection Systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set. *Inf. Secur. J. A Glob. Perspect.* **2016**, *25*, 18–31. [CrossRef]
30. Gad, A.R.; Nashat, A.A.; Barkat, T.M. Intrusion detection system using machine learning for vehicular ad hoc networks based on ToN-IoT dataset. *IEEE Access* **2021**, *9*, 142206–142217. [CrossRef]
31. KDD Cup 1999. Available online: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html (accessed on 1 February 2023).
32. Amit, I.; Matherly, J.; Hewlett, W.; Xu, Z.; Meshi, Y.; Weinberger, Y. Machine learning in cyber-security-problems, challenges and data sets. *arXiv* **2018**, arXiv:1812.07858.
33. Kayacik, H.G.; Zincir-Heywood, A.N.; Heywood, M.I. Selecting features for intrusion detection: A feature relevance analysis on KDD 99 intrusion detection datasets. In Proceedings of the Third Annual Conference on Privacy, Security and Trust, Saint Andrews, NB, Canada, 12–14 October 2005; Volume 94, pp. 1722–1723.
34. Sahu, A.; Mao, Z.; Davis, K.; Goulart, A.E. Data processing and model selection for machine learning-based network intrusion detection. In Proceedings of the 2020 IEEE International Workshop Technical Committee on Communications Quality and Reliability (CQR), Stevenson, WA, USA, 14 May 2020; pp. 1–6.
35. Buczak, A.L.; Guven, E. A survey of data mining and machine learning methods for cyber security intrusion detection. *IEEE Commun. Surv. Tutor.* **2015**, *18*, 1153–1176. [CrossRef]
36. Yang, X.; Peng, G.; Zhang, D.; Lv, Y. An Enhanced Intrusion Detection System for IoT Networks Based on Deep Learning and Knowledge Graph. *Secur. Commun. Netw.* **2022**, *2022*, 4748528. [CrossRef]
37. Gudivada, V.; Apon, A.; Ding, J. Data quality considerations for big data and machine learning: Going beyond data cleaning and transformations. *Int. J. Adv. Softw.* **2017**, *10*, 1–20.
38. Cawley, G.C.; Talbot, N.L. On over-fitting in model selection and subsequent selection bias in performance evaluation. *J. Mach. Learn. Res.* **2010**, *11*, 2079–2107.
39. Alkadi, S.; Al-Ahmadi, S.; Ismail MM, B. Better Safe Than Never: A Survey on Adversarial Machine Learning Applications towards IoT Environment. *Appl. Sci.* **2023**, *13*, 6001. [CrossRef]
40. Paleyes, A.; Urma, R.G.; Lawrence, N.D. Challenges in deploying machine learning: A survey of case studies. *ACM Comput. Surv.* **2022**, *55*, 1–29. [CrossRef]
41. Hansson, K.; Yella, S.; Dougherty, M.; Fleyeh, H. Machine learning algorithms in heavy process manufacturing. *Am. J. Intell. Syst.* **2016**, *6*, 1–13.
42. Monasterios, Y.D.P. *Adversarial Machine Learning: A Comparative Study on Contemporary Intrusion Detection Datasets*; The University of Toledo: Toledo, OH, USA, 2020.
43. Khraisat, A.; Alazab, A. A critical review of intrusion detection systems in the internet of things: Techniques, deployment strategy, validation strategy, attacks, public datasets and challenges. *Cybersecurity* **2021**, *4*, 18. [CrossRef]
44. Luo, Z.; Zhao, S.; Lu, Z.; Sagduyu, Y.E.; Xu, J. Adversarial machine learning based partial-model attack in IoT. In Proceedings of the 2nd ACM Workshop on Wireless Security and Machine Learning, Linz, Austria, 13 July 2020; pp. 13–18.
45. Arp, D.; Quiring, E.; Pendlebury, F.; Warnecke, A.; Pierazzi, F.; Wressnegger, C.; Cavallaro, L.; Rieck, K. Dos and Don'ts of Machine Learning in Computer Security. In Proceedings of the 31st USENIX Security Symposium (USENIX Security 22), Boston, MA, USA, 10–12 August 2022; pp. 3971–3988.
46. Garc´ıa, S.; Zunino, A.; Campo, M. Survey on network-based botnet detection methods. *Secur. Commun. Netw.* **2014**, *7*, 878–903. [CrossRef]
47. Ge, M.; Syed, N.F.; Fu, X.; Baig, Z.; Robles-Kelly, A. Towards a deep learning-driven intrusion detection approach for Internet of Things. *Comput. Netw.* **2021**, *186*, 107784. [CrossRef]
48. Małowidzki, M.; Berezinski, P.; Mazur, M. Network intrusion detection: Half a kingdom for a good dataset. In Proceedings of the NATO STO SAS-139 Workshop, Lisbon, Portugal, 23 April 2015.
49. Ring, M.; Wunderlich, S.; Scheuring, D.; Landes, D.; Hotho, A. A survey of network-based intrusion detection data sets. *Comput. Secur.* **2019**, *86*, 147–167. [CrossRef]
50. Gharib, A.; Sharafaldin, I.; Lashkari, A.H.; Ghorbani, A.A. An evaluation framework for intrusion detection dataset. In Proceedings of the 2016 International Conference on Information Science and Security (ICISS), Pattaya, Thailand, 19–22 December 2016; IEEE: Pattaya, Thailand, 2016; pp. 1–6. [CrossRef]
51. Bulavas, V.; Marcinkevičius, V.; Rumiński, J. Study of multi-class classification algorithms' performance on highly imbalanced network intrusion datasets. *Informatica* **2021**, *32*, 441–475. [CrossRef]
52. Fan, W. Data quality: From theory to practice. *Acm Sigmod Rec.* **2015**, *44*, 7–18.
53. Lemaitre, G.; Nogueira, F.; Aridas, C.K. Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. *J. Mach. Learn. Res.* **2016**, *18*, 1–5.

54. Pears, R.; Finlay, J.; Connor, A.M. Synthetic Minority Over-sampling TEchnique (SMOTE) for Predicting Software Build Outcomes. *arXiv* **2014**, arXiv:1407.2330.
55. Daza, L.; Acuna, E. Feature selection based on a data quality measure. In Proceedings of the World Congress on Engineering, London, UK, 2–4 July 2008; Volume 2, pp. 1095–1099.
56. Khraisat, A.; Gondal, I.; Vamplew, P.; Kamruzzaman, J. Survey of intrusion detection systems: Techniques, datasets and challenges. *Cybersecurity* **2019**, *2*, 20. [CrossRef]
57. Shetye, A. Feature Selection with Sklearn and Pandas. 2019. Available online: https://towardsdatascience.com/featureselection-with-pandas-e3690ad8504b (accessed on 20 January 2023).
58. Pilnenskiy, N.; Smetannikov, I. Modern Implementations of Feature Selection Algorithms and Their Perspectives. In Proceedings of the 2019 25th Conference of Open Innovations Association (FRUCT), Helsinki, Finland, 5–8 November 2019; pp. 250–256.
59. Studer, S.; Bui, T.B.; Drescher, C.; Hanuschkin, A.; Winkler, L.; Peters, S.; Müller, K.R. Towards CRISP-ML (Q): A machine learning process model with quality assurance methodology. *Mach. Learn. Knowl. Extr.* **2021**, *3*, 392–413. [CrossRef]
60. Banaamah, A.M.; Ahmad, I. Intrusion Detection in IoT Using Deep Learning. *Sensors* **2022**, *22*, 8417. [CrossRef]
61. Hussain, F.; Hussain, R.; Hassan, S.A.; Hossain, E. Machine learning in IoT security: Current solutions and future challenges. *IEEE Commun. Surv. Tutor.* **2020**, *22*, 1686–1721. [CrossRef]
62. Qayyum, A.; Usama, M.; Qadir, J.; Al-Fuqaha, A. Securing connected & autonomous vehicles: Challenges posed by adversarial machine learning and the way forward. *IEEE Commun. Surv. Tutor.* **2020**, *22*, 998–1026.
63. Liashchynskyi, P.; Liashchynskyi, P. Grid Search, Random Search, Genetic Algorithm: A Big Comparison for NAS. *arXiv* **2019**, arXiv:1912.06059.
64. Ahmad, Z.; Shahid Khan, A.; Wai Shiang, C.; Abdullah, J.; Ahmad, F. Network intrusion detection system: A systematic study of machine learning and deep learning approaches. *Trans. Emerg. Telecommun. Technol.* **2021**, *32*, e4150.
65. Moustafa, N.; Slay, J. UNSW-NB15: A comprehensive data set for network intrusion detection systems (unsw-nb15 network data set). In Proceedings of the 2015 Military Communications and Information Systems Conference (MilCIS), Canberra, ACT, Australia, 10–12 November 2015. [CrossRef]
66. Papadopoulos, P.; Thornewill von Essen, O.; Pitropakis, N.; Chrysoulas, C.; Mylonas, A.; Buchanan, W.J. Launching adversarial attacks against network intrusion detection systems for iot. *J. Cybersecur. Priv.* **2021**, *1*, 252–273. [CrossRef]
67. Peterson, J.M.; Leevy, J.L.; Khoshgoftaar, T.M. A review and analysis of the bot-iot dataset. In Proceedings of the 2021 IEEE International Conference on Service-Oriented System Engineering (SOSE), Oxford, UK, 23–26 August 2021; pp. 20–27.
68. Moustafa, N. Ton-Iot Datasets 2019. Available online: https://ieee-dataport.org/documents/toniot-datasets (accessed on 15 January 2023).
69. Ferrag, M.A. EdgeIIoT Dataset 2022. Available online: https://www.kaggle.com/datasets/mohamedamineferrag/edgeiiotset-cybersecurity-dataset-of-iot-iiot (accessed on 15 January 2023).
70. Rodríguez, M.; Alesanco, Á.; Mehavilla, L.; García, J. Evaluation of Machine Learning Techniques for Traffic Flow-Based Intrusion Detection. *Sensors* **2022**, *22*, 9326. [CrossRef] [PubMed]
71. Al-Garadi, M.A.; Mohamed, A.; Al-Ali, A.K.; Du, X.; Ali, I.; Guizani, M. A survey of machine and deep learning methods for internet of things (IoT) security. *IEEE Commun. Surv. Tutor.* **2020**, *22*, 1646–1685. [CrossRef]
72. Martins, N.; Cruz, J.M.; Cruz, T.; Abreu, P.H. Adversarial machine learning applied to intrusion and malware scenarios: A systematic review. *IEEE Access* **2020**, *8*, 35403–35419. [CrossRef]
73. Deshmukh, J.V.; Sankaranarayanan, S. Formal techniques for verification and testing of cyber-physical systems. In *Design Automation of Cyber-Physical Systems*; Springer: Cham, Switzerland, 2019; pp. 69–105. [CrossRef]
74. Seshia, S.A.; Sadigh, D.; Sastry, S.S. Toward verified artificial intelligence. *Commun. ACM* **2022**, *65*, 46–55. [CrossRef]