*electronics*

MDPI

*Article*

# Using Feature Selection Enhancement to Evaluate Attack Detection in the Internet of Things Environment

**Khawlah Harahsheh *, Rami Al-Naimat and Chung-Hao Chen**

Department of Electrical & Computer Engineering, Old Dominion University, Norfolk, VA 23529, USA; ramiplan@yahoo.com (R.A.-N.); cxchen@odu.edu (C.-H.C.)
* Correspondence: khawlah_moh@hotmail.com or khara001@odu.edu

**Abstract:** The rapid evolution of technology has given rise to a connected world where billions of devices interact seamlessly, forming what is known as the Internet of Things (IoT). While the IoT offers incredible convenience and efficiency, it presents a significant challenge to cybersecurity and is characterized by various power, capacity, and computational process limitations. Machine learning techniques, particularly those encompassing supervised classification techniques, offer a systematic approach to training models using labeled datasets. These techniques enable intrusion detection systems (IDSs) to discern patterns indicative of potential attacks amidst the vast amounts of IoT data. Our investigation delves into various aspects of supervised classification, including feature selection, model training, and evaluation methodologies, to comprehensively evaluate their impact on attack detection effectiveness. The key features selected to improve IDS efficiency and reduce dataset size, thereby decreasing the time required for attack detection, are drawn from the extensive network dataset. This paper introduces an enhanced feature selection method designed to reduce the computational overhead on IoT resources while simultaneously strengthening intrusion detection capabilities within the IoT environment. The experimental results based on the InSDN dataset demonstrate that our proposed methodology achieves the highest accuracy with the fewest number of features and has a low computational cost. Specifically, we attain a 99.99% accuracy with 11 features and a computational time of 0.8599 s.

**Keywords:** machine learning; feature selection; intrusion detection system; IoT; cybersecurity; wireless security; SDN



**Citation:** Harahsheh, K.; Al-Naimat, R.; Chen, C.-H. Using Feature Selection Enhancement to Evaluate Attack Detection in the Internet of Things Environment. *Electronics* **2024**, *13*, 1678. https://doi.org/10.3390/electronics13091678

Academic Editor: Aryya Gangopadhyay

Received: 4 March 2024
Revised: 17 April 2024
Accepted: 24 April 2024
Published: 26 April 2024



## 1. Introduction

As the Internet of Things (IoT) continues to proliferate, ensuring the security of interconnected devices and networks becomes increasingly crucial. A complete IoT system includes devices, sensors, networks, software, and other essential components necessary for operation and interconnection. Devices and sensors of this nature often have low resource requirements and multiple security vulnerabilities from manufacturers [1]. It is forecasted that by 2030, there will be approximately 500 billion IoT devices connected to the internet [2]. Edge devices, gateways, cloud servers, data analysis tools, and user interfaces represent the main components of an IoT system. In addition to having sensors and controllers, edge devices can also perform initial data reviews before transmitting it to the cloud.

Cybersecurity is a critical concern in today's digital age due to the increasing number of cyberattacks and the growing sophistication of cybercriminals. Ensuring the security of computer systems, networks, and data is of utmost importance, particularly for IoT devices. IoT environments are vulnerable to various cyber threats and attacks, making intrusion detection a fundamental aspect of IoT security. There are many security flaws in IoT devices, often originating from the manufacturer, along with insufficient defense against

cyberattacks in edge network areas. Additionally, the importance of the data collected by IoT devices contributes to increased cyberattacks in IoT systems.

Traditional rule-based IDSs have limitations in detecting emerging and unknown attacks. This has led to the integration of machine learning techniques into IDSs, enabling them to adapt and evolve to ever-changing security threats. Supervised classification refers to a type of machine learning algorithm that learns from labeled data to classify input into predefined categories or classes. In this approach, classification algorithms are trained on labeled data to classify network traffic as normal or malicious. By analyzing the patterns and features in labeled data, these algorithms can identify suspicious behavior in unseen traffic. This makes supervised learning a powerful tool for IDS to distinguish legitimate activity from cyberattacks, including device spoofing, data tampering, and denial-of-service attacks on the Internet of Things (IoT).

The dynamic and heterogeneous nature of IoT networks presents a unique challenge. Classification algorithms offer a viable solution by dynamically adapting to evolving attack landscapes. This iterative process involves continuous learning from labeled data, enabling these algorithms to enhance their accuracy in detecting and mitigating IoT-specific threats. However, a critical factor influencing performance is data dimensionality or the number of features used. In resource-constrained IoT devices, selecting the most relevant features becomes crucial to optimize the performance of machine learning models for IDSs. This is especially important as securing these environments with traditional techniques can be challenging.

The sequential steps involved in the intrusion detection process using machine learning are depicted in Figure 1. It begins with dataset collection, where data relevant to both normal and malicious activities are gathered. Subsequently, data preprocessing ensures the data's cleanliness, transformation, and formatting for analysis. Following this, feature selection identifies the most informative attributes for model training. Model selection involves choosing an appropriate algorithm tailored to the problem at hand, while model training allows the model to learn from the preprocessed data. Model evaluation assesses the model's performance using validation datasets, and model optimization fine-tunes the model for enhanced accuracy.

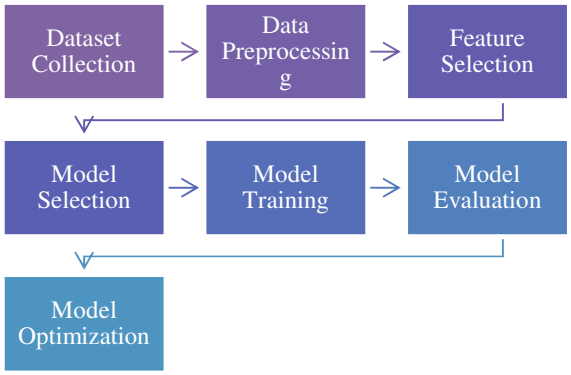


**Figure 1.** Intrusion detection steps using machine learning.

It becomes imperative to identify and select the most relevant data features to enhance the performance of a machine learning model, particularly for IDSs in IoT devices where securing these environments with conventional algorithms and techniques poses a challenge. IoT devices often have limitations, such as low computing capabilities, limited power sources, restricted storage, or restricted memory capacity. Traditional feature selection methods face optimization difficulties and have high computational complexity [2]. The limitations of IoT devices lead to a higher likelihood of attacks or flaws in their security protocols [3]. These limitations and challenges of IoT encompass various categories, including hardware constraints, software issues, network concerns, and security vulnerabilities.

It is crucial to thoroughly consider all of these limitations and challenges before embarking on the development of any systems and security solutions.

Feature selection involves searching within the training data to enhance the classification accuracy. It is crucial to identify and select the most relevant features in the data to improve the performance of the ML model, particularly in anomaly-based IDSs [4]. This process entails selecting a subset of input features from the original set of features in the training data. Feature selection holds significant importance, especially as the data scale increases, to improve the accuracy and efficiency of machine learning models and prevent overfitting [5]. One of the main challenges in ML is to identify the group of relevant features that strike a balance between high accuracy and low time complexity [6]. Three primary approaches can be used to classify the many feature selection techniques into general categories [7]:

- Filter Methods: Filter approaches do not need to train a machine learning model because they rely on statistical metrics. They evaluate each feature's importance separately without referencing the learning algorithm. Common filter methods include correlation-based feature selection, the chi-squared test, and information gain and mutual information.
- Wrapper Methods: By training and evaluating machine learning models on various feature combinations, wrapper approaches assess feature subsets. Although they require more computing power than filter approaches, they may produce feature subsets that are better. Common wrapper methods include forward selection, backward elimination, and recursive feature elimination (RFE).
- Embedded Methods: Feature selection is a crucial step in the model training process that is carried out by embedded methods. Usually, they are employed with algorithms that facilitate feature selection by default. Common embedded methods include L1 regularization (Lasso), tree-based methods, and feature importance from Support Vector Machines (SVMs).

The contributions of this work are as follows:

- Innovative Approach to Feature Selection: This study presents a novel hybrid feature selection methodology tailored specifically to IoT environments. This approach progresses through several defined stages, each contributing to selecting the most relevant features for intrusion detection systems (IDSs) in IoT contexts.
- Caching Mechanism: This study introduces a new caching mechanism to improve the efficiency of feature selection. The system identifies and caches significant features, streamlining subsequent iterations by retrieving cached features when the dataset's feature count remains unchanged.
- Benchmarking Against Outdated Practices: This study highlights the limitations of using non-compatible and outdated datasets, like the KDD'99, in current IDS research. Our results underscore the importance of updated and relevant datasets for developing more accurate and efficient IDS solutions tailored to modern IoT ecosystems.

In this study, the latest InSDN dataset, published in 2020, is used. The rest of this paper is organized as follows: Section 2 reviews the related research on enhancing feature selection. Section 3 provides an overview of the proposed method, the dataset used, and the four stages for enhancing feature selection. Section 4 presents the experimental results and discussion, followed by Section 5, which discusses future work and challenges. Finally, Section 6 concludes this paper.

## 2. Related Work

An IDS plays a critical role in safeguarding networks by identifying and classifying malicious activities. ML techniques have emerged as powerful tools for enhancing the capabilities of IDSs, particularly in the dynamic and heterogeneous environments presented by the IoT. This review explores the application of supervised and unsupervised

learning algorithms in ML-based IDSs, with a focus on feature selection techniques to optimize performance.

### A. Detection System based on Machine Learning

An IDS utilizes one of three detection methods to identify malicious activities. Firstly, signature-based detection compares possible attack signatures with stored ones from previous attacks [8]. While effective with a low false positive rate, it struggles against zero-day attacks where the signature is absent. Secondly, anomaly detection identifies abnormal behavior that deviates from regular baseline operations [9]. This method outperforms signature-based detection in detecting zero-day attacks due to the uniqueness of operational baselines for individual networks. However, it often suffers from a high rate of false positive alarms due to legitimate network behaviors being flagged as anomalies. Thirdly, hybrid detection combines signature-based and anomaly techniques to alleviate their weaknesses [10]. To address zero-day attacks and high false positive problems, multiple algorithms must concurrently process events to determine anomality while simultaneously matching signatures to previously recorded attacks.

Machine learning methods in IDSs fall into two categories based on the training method. Supervised learning algorithms, such as Logistic Regression, Gaussian Naive Bayes, Support Vector Machines (SVMs), and Random Forest, are trained on labeled data [11,12]. Unsupervised learning algorithms, including K-Means Clustering and Single Linkage Clustering, do not rely on labeled data and can autonomously classify traffic patterns [13].

### B. Feature selection methods for IDS

An essential aspect of ML-based IDSs is their ability to process large amounts of data quickly to detect attacks in real time. However, not all features contribute equally to the classification process, and some may introduce noise or are highly correlated with others [14]. Feature selection techniques aim to improve classification performance by selecting only the most significant features, thus enabling ML-based IDSs to make predictions more efficiently.

Several studies have explored feature selection methods for IDSs, each contributing unique insights to enhance detection accuracy. Thaseen et al. [15] proposed an intrusion detection method that combines chi-square feature selection with a Multi-Class Support Vector Machine (MSVM) to minimize the processing time while maximizing the classification accuracy for network attacks. Kumar et al. [16] introduced the gain ratio feature selection technique, which, when coupled with an updated Naive Bayes classifier, outperformed existing methods in classification performance. Pham et al. [17] presented an IDS framework utilizing gain ratio feature selection with an ensemble bagging model, demonstrating improved classification accuracy and reduced false alarm rates compared to alternative approaches. Shahbaz et al. [18] investigated a feature selection approach based on Correlation-Based Feature Selection (CFS) and Symmetrical Uncertainty (SU), showcasing enhanced classification accuracy relative to conventional methods like CFS, information gain, and chi-square techniques. Le et al. [19] introduced a hybrid Sequence Forward Selection (SFS) algorithm coupled with a Decision Tree model, achieving significantly improved prediction performance and reduced false alarm rates across various neural network architectures.

Additionally, Taher et al. [20] proposed a wrapper feature selection method with an Artificial Neural Network (ANN) and Support Vector Machine (SVM), yielding superior detection rates compared to existing models. Furthermore, Yeshalem et al. [21] introduced a Bootstrap-Based Homogeneous Ensemble Feature Selection (BHmEFS) method, which selected a subset of relevant features to enhance classification accuracy, while Bostani et al. [22] devised a hybrid feature selection method combining a Binary Gravitational Search Algorithm (BGSA) with Mutual Information (MI), achieving higher accuracy and detection rates than standard wrapper-based and filter-based techniques. Finally, Zhang et al. [23] proposed a two-level network intrusion detection model integrating the ReliefF algorithm

and borderline Synthetic Minority Oversampling Technique (SMOTE), demonstrating improved detection accuracy, particularly for minority samples. Despite these advancements, single feature selection's potential impact on prediction performance remains underexplored, prompting the development of the Hybrid Ensemble Feature Selection (HEFS) approach to retain critical candidate features and enhance prediction performance in NIDS.

### C. InSDN Dataset

Many researchers are interested in securing the IoT and wireless networks, and in this subsection, the focus is on the research that uses the InSDN dataset, and it compares those results with the results of this study. Numerous published studies rely on datasets that are either incompatible or outdated. For example, the KDDCup99 dataset, which was published in 1999, has been used in recent research [24,25]. Additionally, the Nsl-KDD dataset, published in 2009, has been used in studies such as [4,26–30]. However, using outdated datasets in an intrusion detection system (IDS) for machine learning may not accurately reflect the current threat landscape. New types of attacks and vulnerabilities may have emerged since the dataset was created, making the model less effective in identifying contemporary threats. Outdated datasets can also introduce disadvantages, such as limited relevance to current threats, obsolete features, a lack of diversity, mismatch with real-world scenarios, and reduced model performance, which can lead to a false sense of security. Therefore, in this section, we discuss several methodologies proposed in recent studies that utilized the InSDN dataset, which was published in 2020, such as [31–35].

D. Firdaus et al. [34] proposed a DDoS detection method using machine learning with an Ensemble Algorithm. Their study consisted of two methodologies: clustering and classification and Ensemble Algorithm clustering and classification. The researchers utilized Ensemble Algorithm K-means++ and Random Forest to achieve high detection accuracy and efficiency, obtaining a remarkable accuracy of 100% using 15 features out of the original dataset's 84. Conversely, other researchers, such as those in [31,35,36], also achieved very high accuracies of 99.42%, 99.9961%, and 99.94%, respectively, but they employed 48 or 56 features. A. Ibrahimy et al. [36] used feature correlation to reduce the number of features in the InSDN dataset, while V. Hnamte and J. Hussain [35] employed two 1D convolutional layers to capture important features from the input data. These layers are connected to a flattened layer, which converts the output of the convolutional layers into a 1D array. Subsequently, four fully connected dense layers are utilized, activated by the rectified linear unit (ReLU) activation function.

The researchers in [32,33] achieved an accuracy of 98.98% with 30 features and 98% with 20 features, respectively. A. Zainudin et al. [32] employed the LightGBM feature selection technique, which utilized three main modules: pre-block, depth-wiseConv, and stacked GRU. The depth-wiseConv module utilized a residual connection to enhance training model performance and address the issue of gradient vanishing. Additionally, a factorized convolution architecture was utilized to create a lightweight model structure.

## 3. Material and Methods

The number of features in an intrusion detection system (IDS) significantly influences its accuracy and performance. A larger feature set can provide a more detailed representation of the data, potentially leading to the better detection of complex intrusion patterns. However, it can also introduce noise and irrelevant information, which can confuse the model, leading to overfitting and reduced generalization capabilities. Conversely, a smaller, well-curated feature set can result in a more streamlined and interpretable model, reducing the computational load and enabling the IDS to operate more efficiently, especially for IDSs in IoT environments and resource limitations.

This paper introduces an improved feature selection methodology aimed at achieving high accuracy while utilizing a reduced number of features. The focus on a lower feature count is particularly relevant for wireless networks, which often have resource constraints. In this section, we present a hybrid feature selection approach designed to enhance the performance of intrusion detection system (IDS) classification. We demonstrate

the application of our methodology on the InSDN dataset. Software-defined networking (SDN) has been developed to reduce network complexity by centralizing the control and management of the entire network [31]. SDN-based Industrial Internet of Things (IIoT) networks utilize a centralized controller, which can make them vulnerable to DoS/DDoS attacks and susceptible to single points of failure [37].

### *A. Dataset*

The dataset used in this research is called the InSDN dataset, which was published in 2020 by M. Elsayd et al. [31]. The purpose of creating the InSDN dataset was to reduce its size compared to other IDS datasets. The researchers collected real-world network traffic from an SDN environment and classified it based on the presence of DDoS attacks. Unlike NSL-KDD and KDD99, InSDN provides a realistic representation of traffic in an SDN environment [34].

The dataset consists of a total of 343,889 data records with 84 features. Of these, 127,828 records correspond to ordinary traffic, while 216,061 records correspond to attack traffic in both OSV and metaspoitable files. InSDN includes various attack scenarios, such as SYN, TCP, UDP, and ICMP floods, as well as Slowloris attacks. Table 1 presents the distribution of samples within the InSDN dataset.

**Table 1.** Distribution of samples in InSDN dataset.

| CSV File | Class | Sample | Percentage | Total |
|---|---|---|---|---|
| **Normal_data.csv** | Normal traffic | 68,424 | 20% | 68,424 |
| **OVS.csv** | DoS | 52,471 | 39.76% | 136,743 |
| | DDoS | 48,413 | | |
| | Probe | 36,372 | | |
| | BFA | 1110 | | |
| | Web_attack | 192 | | |
| | Botnet | 164 | | |
| **metasploitable-2.csv** | DoS | 1145 | 40.34% | 138,772 |
| | DDoS | 73,529 | | |
| | Probe | 61,757 | | |
| | BFA | 295 | | |
| | Exploitation (R2L) | 17 | | |
| **Total** | | | | 343,889 |

The InSDN dataset was generated using multiple virtual machines with an SDN network architecture. The standard Ubuntu system represents regular users, while the Kali system represents attackers performing various types of attacks on the SDN network [35].

### *B. Hybrid feature selection approach*

In this paper, we present a sophisticated hybrid feature selection approach specifically devised for the nuanced requirements of IoT environments. The proposed methodology unfolds across several defined stages:

1. Initial Stage (Data Preprocessing);
2. Second Stage (Dimension Reduction);
3. Feature Selection Stage;
4. Final Stage (Model Evaluation);
5. Caching Mechanism.

Each stage contributes to the overarching goal of selecting the most pertinent features for use in the IDS. The resulting features are of pivotal importance in machine learning,

directly influencing the model's learning efficacy and predictive accuracy. Our methodological process, depicted in Figure 2, underscores the critical nature of feature selection in enhancing IDS performance within IoT contexts.

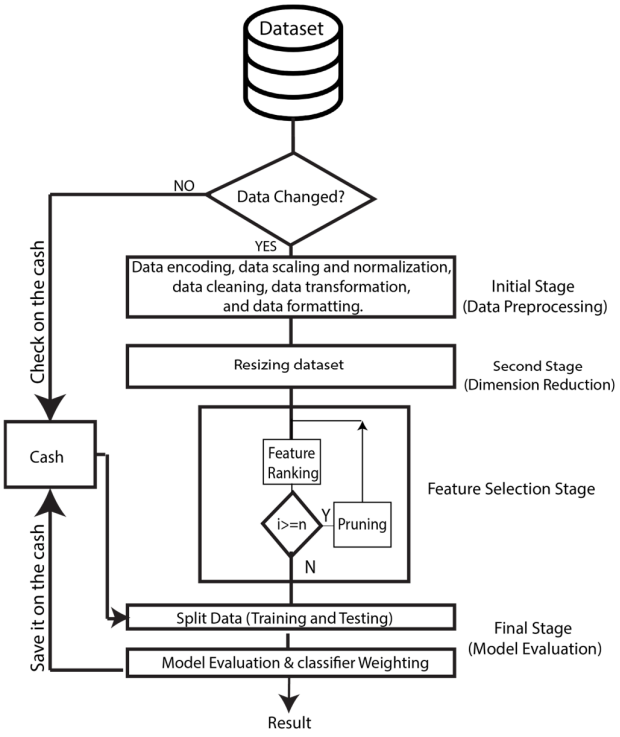


**Figure 2.** The whole process of the proposed method.

Incorporating a caching mechanism significantly improves the efficiency of our feature selection methodology. Throughout our proposed approach, the system meticulously identifies and caches the most significant features for subsequent iterations. After each iteration, the system checks the dataset's feature count. Unchanged counts prompt the retrieval of previously selected features from the cache, streamlining the process to the final evaluation stage. Conversely, any change in feature count mandates a fresh execution through all four stages, commencing with data preprocessing and culminating in model evaluation.

1. *Initial Stage (Data Preprocessing)*

The data processing stage—which includes data encoding, scaling, normalization, cleaning, transformation, and formatting—is essential for preparing the raw dataset and creating a refined dataset where the intrinsic patterns can be more easily and accurately discerned by the IDS, leading to improved detection accuracy and performance. Data encoding converts categorical data into a numerical format, which is necessary because most machine learning algorithms can only interpret numerical values. This ensures that valuable categorical information, such as protocol types or service names, is retained and can be used in the detection process. Data scaling and normalization are critical steps that adjust the range of the feature data. Scaling alters the range of the data to a common scale without distorting the differences in the ranges of values, while normalization adjusts the data in such a way that their distribution will have a mean value of 0 and a standard deviation of 1. These steps are crucial to prevent features with larger scales from dominating those with smaller scales, thus ensuring that the IDS model treats all features equally.

Data cleaning is the process of detecting and correcting (or removing) corrupt or inaccurate records from a dataset, which might otherwise lead to incorrect conclusions. Data transformation involves converting data into a suitable format or structure for analysis, which could include generating polynomial features or interaction terms if needed. Finally, data formatting ensures that all data are consistently and correctly formatted, such as

ensuring data times are in a uniform format, which is fundamental for a time-series analysis in intrusion detection. One major challenge in IDS is dealing with high-dimensional and imbalanced data, which increases the cost of machine learning [38]. Well-preprocessed data lower the chance of overfitting, aid in the model's learning of pertinent patterns, and produce predictions that are more reliable and accurate.

2. *Second Stage (Dimension Reduction)*

The second stage is the dataset resizing stage, which is a crucial step in data preprocessing; there are various methods used for dimensionality reduction, such as Principal Component Analysis (PCA), to suit different needs. A Linear Discriminant Analysis (LDA) excels in classification tasks by using label information to maintain class distinctions. t-Distributed Stochastic Neighbor Embedding (t-SNE) shines in visualizing complex data but can be computationally demanding and sensitive to hyperparameters. Uniform Manifold Approximation and Projection (UMAP) offers a faster alternative to t-SNE, and it is suitable for large datasets while preserving the data structure. Autoencoders (AEs), leveraging neural networks, are powerful for non-linear transformations but require substantial data and computational resources. Random Projection provides a speedy, simple approach, though it may yield less precise results due to its stochastic nature. Finally, Isomap is effective for unfolding manifolds by maintaining geodesic distances, albeit at a higher computational cost. The choice among these techniques hinges on the specific dataset and analytical objectives, as well as balancing factors like visualization clarity, computational efficiency, and interpretability. In this stage, we use PCA because it performs data resizing at a faster speed compared with the other methods, while the other methods gave us good results but took more computational time.

After employing the dimension reduction algorithm in our proposed method, the preprocessed data will be fed into the feature selection stage. Each line will undergo feature ranking and pruning after a specific number of iterations. The combination process may involve selecting the best features based on scoring metrics or aggregating results from different workers. The outcome of the feature selection stage will be cached, which offers a time-saving advantage for subsequent runs on similar datasets or during iterative model development.

3. *Feature Selection Stage*

The act of choosing a subset of the most pertinent features (input variables or attributes) from the initial collection of features is known as feature selection in machine learning. Improving the machine learning model's efficiency, decreasing overfitting, and improving performance are the goals. Filter methods select features that are independent of one another and are heavily dependent on the output, while wrapper techniques strive to maximize some specified criteria with respect to the feature set as part of the selection process [39].

In our proposed method, we utilize the filter method because we aim to enhance the speed of our IDS. A higher speed translates to lower power usage, reduced capacity requirements, and a faster computational time. Filter methods are generally faster compared to wrapper methods. They preprocess the data independently of the learning algorithm. Common filter methods include mutual information, the chi-squared test, and correlation-based feature selection. These methods can be highly efficient, particularly when dealing with high-dimensional data. They employ statistical or correlation-based techniques to rapidly evaluate the relevance of individual features without the need to train a machine learning model.

Common metrics for feature selection include mutual information, the chi-squared test, and correlation coefficients. In the study by the researchers of [40], both filter and wrapper methods were compared, and it was found that wrapper methods often require more computing resources but provide higher accuracy. On the other hand, filter methods require fewer resources but lack optimization capabilities. Filter methods are known for their ease of implementation and faster execution compared to wrapper and hybrid

methods. However, wrapper methods outperform filter methods in terms of accuracy, albeit at the cost of a slower execution [41]. Another study by Zhang et al. [27] also compared these feature selection methods and found that wrapper methods are closely tied to the classifier, which uses the performance of the classifier as an objective function to evaluate the current feature subset. Wrapper methods train a new model for each feature subset, resulting in a relatively higher computational cost.

The provided rank feature pseudo code outlines a procedure for ranking features using a combination of a variance threshold method and a Random Forest classifier (Algorithm 1). The pseudo-code stops short of detailing the application of the Random Forest classifier; however, in typical usage, the Random Forest would be employed post-variance threshold application to rank the features according to their importance. This importance is gleaned from how much the features contribute to the accuracy of the Random Forest model. By combining the variance threshold with Random Forest ranking, the algorithm aims to enhance the feature selection process, potentially improving the predictive power of the subsequent machine learning models and offering a deeper understanding of the features' influence within the dataset. The algorithm commences by initializing a VarianceThreshold object with a predefined threshold value, set here at 0.1. This threshold acts as a filter to remove features with low variance under the assumption that features with a lower variance may contain less information. The variance_selector is then fitted to the dataset, denoted as X, where it computes the variance for each feature.

**Algorithm 1:** Rank features using a Random Forest classifier pseudocode.

```
1. # Step 1: Initialize the VarianceThreshold object with a threshold value
2. threshold_value = 0.1
3. variance_selector = VarianceThreshold(threshold = threshold_value)
4. # Step 2: Fit the selector to data
5. variance_selector.fit(X)
6. # Step 3: Get the indices of the features selected
7. selected_feature_indices = variance_selector.get_support(indices = True)
8. # Step 4: Use the selected features
9. selected_features = X[: selected_feature_indices]
```

Subsequently, the algorithm retrieves the indices of the features that surpass the variance threshold by invoking the get_support method with indices=True, which returns an array of indices. These indices represent the features that have been deemed important enough to retain based on their variance. In the final step, the algorithm selects the features from dataset X using the obtained indices, creating a subset of X that only includes features with significant variance. This subset, denoted as selected_features, can be used for further processing or directly in the Random Forest classifier for ranking based on feature importance.

We used filter methods to select features based on their statistical properties. Common filter methods include the following:

- Correlation: This involves calculating the correlation between features and remove highly correlated or redundant ones.
- Variance Threshold: This involves removing features with low variance as they often contain little information.
- Statistical Tests: This involves utilizing statistical tests (e.g., chi-squared test and ANOVA) to select features that significantly contribute to the classification.

You can use several different methods and approaches to increase the feature selection process speed. Choosing features can be computationally costly, particularly when working with big datasets or intricate feature spaces. It is challenging for IoT characteristics to exploit the features and attributes of IDS to ensure self-protection [4]. The following are

some methods used in the proposed method (Figure 2) to enhance the speed of the feature selection process:

- Feature Ranking: In this manner, it might not be necessary to assess every feature in the dataset; instead, the most promising features should be concentrated on first.
- Pruning: Early in the feature subset evaluation process, any subsets that do not seem promising should be pruned or removed. This will expedite the feature selection process and shrink the search space. In the proposed methodology, feature ranking is performed in each round, while pruning is conducted every n number of iterations.
- Caching: Implementing a caching mechanism can significantly enhance the efficiency of our feature selection methodology. During our proposed method journey, the system meticulously identifies and selects the most significant features, which are then cached for subsequent iterations. The system checks the dataset's feature count after each iteration. If there is no change in the number of features, the system will swiftly retrieve the previously selected features from the cache, skipping unnecessary steps and advancing directly to the final evaluation stage. Conversely, any alteration in feature count necessitates a fresh execution through all four stages—beginning with data preprocessing and culminating in model evaluation.

*4. ML Model*

It is important to note that while these methods are generally faster, they may not always yield the best subset of features for a specific problem. When choosing the feature selection method, factors such as the nature of the data, the complexity of the problem, the desired trade-off between the speed and model performance, and the specific goals of the analysis should be considered. Sometimes, a slightly slower method may yield better results in terms of model accuracy and generalization.

Using Random Forest (RF) or L1 regularization (Lasso) can offer faster feature selection compared to other techniques. With Random Forest, one can assess feature importance, which is often faster than running more computationally intensive wrapper methods like recursive feature elimination with cross-validation (RFECV). Random Forest models can achieve a high prediction accuracy with a small overhead in terms of computational resource usage [42]. In our proposed method, we utilize Random Forest as the primary supervised machine learning model for detecting attacks. Random Forest offers several advantages that make it well-suited for this task. Notably, it excels in handling both continuous and categorical data, making it versatile for various types of features commonly encountered in intrusion detection datasets. Additionally, Random Forest has robust mechanisms for addressing missing and outlier values, which are common challenges in real-world data. Furthermore, its ability to parallelize computations enables shorter training times, contributing to the efficiency of our detection system [43].

L1 regularization (Lasso) is commonly used in linear models to encourage sparsity in feature selection. It can be relatively fast, especially when combined with optimization techniques like coordinate descent.

*5. Final Stage (Model Evaluation)*

By selecting the most relevant features and eliminating irrelevant or redundant ones, one can enhance the performance of the IDS. This reduces the computational complexity and improves its ability to accurately detect and classify intrusions. Metrics for evaluating intrusion detection systems (IDS) are used to assess the performance of IDS solutions in detecting and preventing network or system intrusions and security breaches. These metrics help security professionals understand the effectiveness and efficiency of their IDS solutions. Below are some common IDS evaluation metrics:

- True Positive (TP): This is the number of intrusions or attacks correctly detected using the IDS.
- True Negative (TN): This is the number of normal or non-malicious network activities correctly detected using the IDS.

- False Positive (FP): This is the number of normal activities incorrectly classified as intrusions or attacks using the IDS. This is also known as a "false alarm".
- False Negative (FN): This is the number of intrusions or attacks that the IDS failed to detect or classify as normal. This is also known as a "missed detection".
- Accuracy: This is the ratio of correctly identified instances (TP + TN) to the total number of instances. It provides an overall measure of the IDS's correctness.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{1}$$

- Precision: Precision is defined as the ratio of true positives (TP) to the total number of instances classified as positive using the IDS (TP + FP). It measures the ability of the IDS to avoid false positives.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{2}$$

- Recall (Sensitivity or True Positive Rate): This is the ratio of true positives (TPs) to the total number of actual positive instances (TP + FN). It measures the IDS's ability to identify all positive instances.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} = \text{Recall}_{\text{Positive}} \tag{3}$$

- F1 Score: This is the harmonic mean of precision and recall. It balances the trade-off between false positives and false negatives [1].

$$\text{F1 Score} = 2 * \frac{\text{precision} * \text{Recall}}{\text{precision} + \text{Recall}} \tag{4}$$

Figure 3 illustrates the performance of a multi-class classification model across nine distinct classes, ranging from 0 to 8, as detailed in the confusion matrix. The findings are as follows:

- Class 0: There are 298 samples correctly classified.
- Class 1: There are 25 samples correctly classified.
- Class 2: There are 14,395 samples correctly classified.
- Class 3: There are 9612 samples correctly classified.
- Class 4: There are 9433 samples correctly classified.
- Class 5: There are 13,493 samples correctly classified.
- Class 6: There are 19,590 samples correctly classified.
- Class 7: There are 0 samples; it appears to be a class with no correct classifications, or it might be a mislabeled row in the confusion matrix.
- Class 8: There are 20 samples correctly classified.

The confusion matrix is square, indicating that there are nine classes in total. Each class's sample count corresponds to the value along the diagonal, which represents the number of true positive predictions for each class. The matrix also provides data on misclassifications; for instance, instances where Class 2 is predicted to be Class 5 (180 instances) or Class 4 is predicted to be Class 5 (213 instances). However, there is an anomaly with Class 7, which does not shows any correct classifications, suggesting that there is an error in the model or the data, or perhaps it is a class without instances.

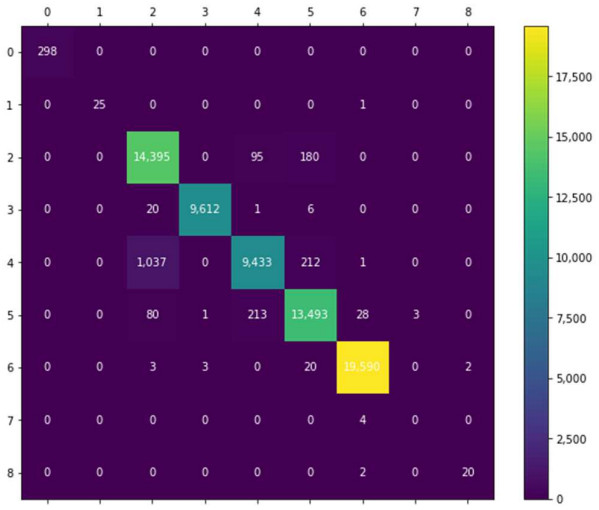


**Figure 3.** Confusion matrix for proposed module.

## 4. Experimental Results and Discussion

In this section, we present the outcomes of our extensive investigation into the effectiveness of the proposed methodology. Our experiments involve extracting crucial features from the InSDN dataset through the suggested stages. We evaluate the performance using classification matrices, which include metrics such as accuracy, loss, and AUC score.

In our proposed system, we have different stages: the initial stage (data preprocessing), second stage (dimension reduction), the feature selection stage, and the final stage (model evaluation). After each stage, we preserved the dataset for size comparison, as illustrated in Figure 4. The original InSDN dataset consists of 84 features. However, after applying the dimension reduction algorithm in the second stage, the dataset is reduced to only 18 columns. The ultimate dataset is further refined, retaining the top 11 features. The most important features are shown in Figure 5.

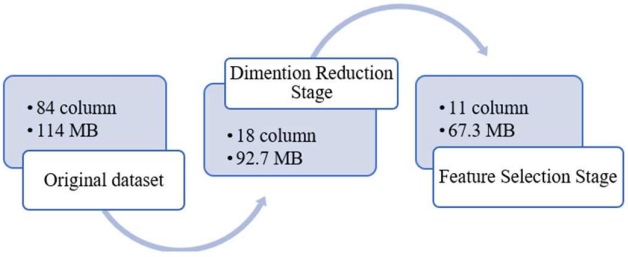


**Figure 4.** Saved data size and number of features after each stage.

The proposed model achieved excellent results with an accuracy of 99.99%, a precision of 99.99%, a recall of 99.99%, an ROC score of 0.99997, and a computational cost of 0.8599 s when using 11 features. Figure 6 depicts the module's results with actual versus predicted values.

Table 2 presents a comparison between our methodology and existing approaches. Our methodology achieves a remarkably high accuracy of 99.99% with a minimal number of features, namely 11 of them. Additionally, we achieved a high computational speed of less than a second, specifically 0.8599 s; this was achieved after the application of our novel feature selection methodology, which effectively reduced the feature set to the most informative 11 features from the original 84 features present in the InSDN dataset. The reduction in feature space is a crucial factor contributing to the notable efficiency of our intrusion detection system (IDS) in processing and classifying data. This makes our methodology suitable for wireless networks, particularly in limited resource environments such as IoT.

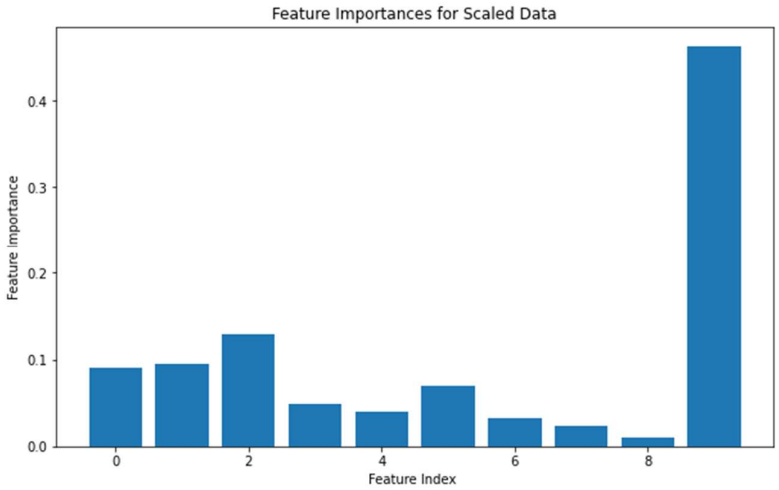


**Figure 5.** Top 8 important features of the InSDN dataset.

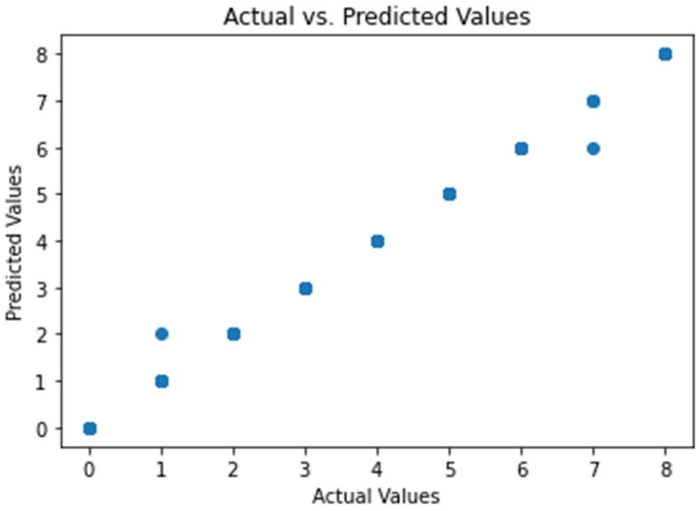


**Figure 6.** Actual vs. predicted values.

**Table 2.** Comparison between our method and other research.

| Ref Number | ML Tech. | Accuracy | # of Features | Time (s) |
|---|---|---|---|---|
| [31] | RF | 99.42% | 48 | 226.151 |
| [32] | CNN | 98.98% | 30 | 0.164 |
| [33] | RF | 98% | 20 | 20.50 |
| [34] | KNN and RF | 100% | 15 | 1.5 |
| [37] | CNN | 90.83% | 19 | 21.0506 |
| [36] | KNN | 99.9961% | 56 | 18.69 |
| [35] | DCNN | 99.94% | - | - |
| **Our paper** | **Hybrid Feature Selection** | **99.99%** | **11** | **0.8599** |

## 5. Future Work and Challenges

Our methodology incorporates a caching mechanism to optimize the feature selection process. After each round, we check the number of columns in the dataset. If the number of columns remains constant, the methodology retrieves features from the cache, bypassing intermediary stages and sending them directly to the final stage. However, if there is a change in the column counts, the proposed approach proceeds through all four stages. It extracts the most influential features, sends them to the model evaluation stage, and stores them in the cache for future use.

In our future work, we aim to enhance our methodology by conducting a comprehensive examination of the data within these columns rather than solely relying on the column count. This adjustment is necessary because there may be cases where the number of columns remains unchanged, but the content within those columns is entirely replaced. Additionally, the methodology might encounter a new dataset with a coincidental matching number of columns. Therefore, we plan to assess the data alone to ensure accurate feature selection.

Our next plan is to further improve the methodology by incorporating lightweight techniques. We intend to evaluate and test these enhancements on larger datasets to ensure their effectiveness and scalability. This way, we can address the power limitations of IoT devices while achieving optimal feature selection performance.

## 6. Conclusions

This paper introduces an advanced feature selection method to address the need for efficiency and a reduced dataset size in an IDS. The method combines multiple stages to optimize secure communication in wireless environments. The main objectives include improving model performance, reducing overfitting, and enhancing the efficiency and speed of the machine learning model. The ultimate goal is to achieve high accuracy with a minimal number of features to make the IDS more compatible with IoT environments.

The proposed methodology is applied to the InSDN dataset, and the experimental results demonstrate its effectiveness. The methodology successfully achieves the objective of a high accuracy with a minimal feature set and low computational cost. Notably, the results show an impressive 99.99% accuracy using only 11 features, with a computational time of 0.8599 s. These findings highlight the potential of the methodology to advance secure communication in wireless IoT environments.

**Author Contributions:** Methodology, K.H.; Data curation, R.A.-N.; Writing—review & editing, C.-H.C. All authors contributed equally to the conceptualization of the study. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research received no external funding.

**Data Availability Statement:** The dataset is available as in reference number [31].

**Conflicts of Interest:** The authors declare no conflicts of interest. All co-authors have seen and agree with the contents of the manuscript, and there is no financial interest to report.

## References

1. Cam, N.T.; Trung, N.G. An Intelligent Approach to Improving the Performance of Threat Detection in IoT. *IEEE Access* **2023**, *11*, 44319–44334. [CrossRef]
2. Chen, H.; Ma, X.; Huang, S. A Feature Selection Method for Intrusion Detection Based on Parallel Sparrow Search Algorithm. In Proceedings of the 2021 16th International Conference on Computer Science & Education (ICCSE), Lancaster, UK, 17–21 August 2021; pp. 685–690. [CrossRef]
3. Harahsheh, K.M.; Chen, C.H. A Survey of Using Machine Learning in IoT Security and the Challenges Faced by Researchers. *Informatica* **2023**, *47*, 1–54. [CrossRef]
4. Natarajan, B.; Bose, S.; Maheswaran, N.; Logeswari, G.; Anitha, T. A New High-Performance Feature Selection Method for Machine Learning-Based IOT Intrusion Detection. In Proceedings of the 2023 12th International Conference on Advanced Computing (ICoAC), Chennai, India, 17–19 August 2023; pp. 1–8. [CrossRef]
5. Yesaswini, A.M.; Annapurna, K. A Hybrid Approach for Intrusion Detection System to Enhance Feature Selection. In Proceedings of the 2023 Second International Conference on Augmented Intelligence and Sustainable Systems (ICAISS), Trichy, India, 23–25 August 2023; pp. 1301–1307. [CrossRef]
6. Abbas, N.; Nasser, Y.; Shehab, M.; Sharafeddine, S. Attack-Specific Feature Selection for Anomaly Detection in Software-Defined Networks. In Proceedings of the 2021 3rd IEEE Middle East and North Africa COMMunications Conference (MENACOMM), Agadir, Morocco, 3–5 December 2021; pp. 142–146. [CrossRef]
7. Tadist, K.; Najah, S.; Nikolov, N.S.; Mrabti, F.; Zahi, A. Feature selection methods and genomic big data: A systematic review. *J. Big Data* **2019**, *6*, 79. [CrossRef]

8. Santoso, B.I.; Idrus MR, S.; Gunawan, I.P. Designing Network Intrusion and Detection System using signature-based method for protecting OpenStack private cloud. In Proceedings of the 2016 6th International Annual Engineering Seminar (InAES), Yogyakarta, Indonesia, 1–3 August 2016; pp. 61–66.
9. Garg, A.; Maheshwari, P. A hybrid intrusion detection system: A review. In Proceedings of the 2016 10th International Conference on Intelligent Systems and Control (ISCO), Coimbatore, India, 7–8 January 2016; pp. 1–5.
10. Tama, B.A.; Comuzzi, M.; Rhee, K.H. TSE-IDS: A two-stage classifier ensemble for intelligent anomaly-based intrusion detection system. *IEEE Access* **2019**, *7*, 94497–94507. [CrossRef]
11. Shah, R.A.; Qian, Y.; Mahdi, G. Group feature selection via structural sparse logistic regression for IDS. In Proceedings of the 2016 IEEE 18th International Conference on High Performance Computing and Communications; IEEE 14th International Conference on Smart City; IEEE 2nd International Conference on Data Science and Systems (HPCC/SmartCity/DSS), Sydney, NSW, Australia, 12–14 December 2016; pp. 594–600.
12. Singh, S.; Kumari, K.; Gupta, S.; Dua, A.; Kumar, N. Detecting Different Attack Instances of DDoS Vulnerabilities on Edge Network of Fog Computing using Gaussian Naive Bayesian Classifier. In Proceedings of the 2020 IEEE International Conference on Communications Workshops (ICC Workshops), Dublin, Ireland, 7–11 June 2020; pp. 1–6.
13. Su, L.P.; Zhang, J.A. The improvement of cluster analysis in intrusion detection system. In Proceedings of the 2017 10th International Conference on Intelligent Computation Technology and Automation (ICICTA), Changsha, China, 9–10 October 2017; pp. 274–277.
14. Ambusaidi, M.A.; He, X.; Nanda, P.; Tan, Z. Building an intrusion detection system using a filter-based feature selection algorithm. *IEEE Trans. Comput.* **2016**, *65*, 2986–2998. [CrossRef]
15. Thaseen, I.S.; Kumar, C.A. Intrusion detection model using fusion of chi-square feature selection and multi-class SVM. *J. King Saud Univ. Comp. Inf. Sci.* **2017**, *29*, 462–472.
16. Kumar, K.; Batth, J.S. Network intrusion detection with feature selection techniques using machine-learning algorithms. *Int. J. Comput. Appl.* **2016**, *150*, 1–13. [CrossRef]
17. Pham, N.T.; Foo, E.; Suriadi, S.; Jeffrey, H.; Lahza, H.F.M.; Abramson, D. (Eds.) Improving performance of intrusion detection system using ensemble methods and feature selection. In Proceedings of the Australian Computer Science Week (ACSW), Brisbane, QLD, Australia, 29 January–2 February 2018; Association for Computing Machinery: New York, NY, USA, 2018; pp. 1–6.
18. Shahbaz, M.B.; Wang, X.; Behnad, A.; Samarabandu, J.; Satyajit Chakrabarti, H.N.S. (Eds.) On efficiency enhancement of the correlation-based feature selection for intrusion detection systems. In Proceedings of the 7th Annual Conference on Information Technology Electronics Mobile Communication (IEMCON), Vancouver, BC, Canada, 13–15 October 2016; IEEE: Piscataway, NJ, USA, 2016; pp. 1–7.
19. Le, T.-T.-H.; Kim, Y.; Kim, H. Network intrusion detection based on novel feature selection model and various recurrent neural networks. *Appl. Sci.* **2019**, *9*, 1392. [CrossRef]
20. Taher, K.A.; Jisan, B.M.Y.; Rahman, M.M.; Tanseer Ali, M.E.Z. (Eds.) Network intrusion detection using supervised machine learning technique with feature selection. In Proceedings of the International Conference on Robotics, Electrical Signal Processing Techniques (ICREST), Dhaka, Bangladesh, 5–7 January 2021; IEEE: Piscataway, NJ, USA, 2019; pp. 643–646.
21. Yeshalem, G.D.; Chen, H.; Din, B.M.Y.; Zhong Li, C.Y. (Eds.) Bootstrap-based homogeneous ensemble feature selection for network intrusion detection system. In Proceedings of the 4th International FLINS Conference on Developments Artificial Intelligence Technologies Computation in Robotics, Cologne, Germany, 18–21 August 2020; World Scientific Publishing Company: Beijing, China, 2020; pp. 16–27.
22. Bostani, H.; Sheikhan, M. Hybrid of binary gravitational search algorithm and mutual information for feature selection in intrusion detection systems. *Soft Comput.* **2017**, *21*, 2307–2324. [CrossRef]
23. Zhang, J.; Zhang, Y.; Li, K.; Guan, S. (Eds.) A network intrusion detection model based on the combination of relief and borderline-smote. In Proceedings of HPCCT & BDAI '20: Proceedings of the 2020 4th High Performance Computing and Cluster Technologies Conference & 2020 3rd International Conference on Big Data and Artificial Intelligence, Qingdao, China, 3–6 July 2020; Association for Computing Machinery: New York, NY, USA, 2020; pp. 199–203.
24. Li, Y.; Shi, K.; Qiao, F.; Luo, H. A Feature Subset Selection Method Based on the Combination of PCA and Improved GA. In Proceedings of the 2020 2nd International Conference on Machine Learning, Big Data and Business Intelligence (MLBDBI), Taiyuan, China, 23–25 October 2020; pp. 191–194. [CrossRef]
25. Hostiadi, D.P.; Atmojo, Y.P.; Huizen, R.R.; Susila, I.M.D.; Pradipta, G.A.; Liandana, I.M. A New Approach Feature Selection for Intrusion Detection System Using Correlation Analysis. In Proceedings of the 2022 4th International Conference on Cybernetics and Intelligent System (ICORIS), Prapat, Indonesia, 8–9 October 2022; pp. 1–6. [CrossRef]
26. Zhao, R.; Mu, Y.; Zou, L.; Wen, X. A Hybrid Intrusion Detection System Based on Feature Selection and Weighted Stacking Classifier. *IEEE Access* **2022**, *10*, 71414–71426. [CrossRef]
27. Zhang, Z.; Wen, J.; Zhang, J.; Cai, X.; Xie, L. A Many Objective-Based Feature Selection Model for Anomaly Detection in Cloud Environment. *IEEE Access* **2020**, *8*, 60218–60231. [CrossRef]
28. Yu, T.; Liu, Z.; Liu, Y.; Wang, H.; Adilov, N. A New Feature Selection Method for Intrusion Detection System Dataset–TSDR method. In Proceedings of the 2020 16th International Conference on Computational Intelligence and Security (CIS), Guangxi, China, 27–30 November 2020; pp. 362–365.

29. Parimala, G.; Kayalvizhi, R. An Effective Intrusion Detection System for Securing IoT Using Feature Selection and Deep Learning. In Proceedings of the 2021 International Conference on Computer Communication and Informatics (ICCCI), Coimbatore, India, 27–29 January 2021; pp. 1–4. [CrossRef]
30. Sarvari, S.; Sani, N.F.M.; Hanapi, Z.M.; Abdullah, M.T. An Efficient Anomaly Intrusion Detection Method with Feature Selection and Evolutionary Neural Network. *IEEE Access* **2020**, *8*, 70651–70663. [CrossRef]
31. Elsayed, M.S.; Le-Khac, N.-A.; Jurcut, A.D. InSDN: A Novel SDN Intrusion Dataset. *IEEE Access* **2020**, *8*, 165263–165284. [CrossRef]
32. Zainudin, A.; Akter, R.; Kim, D.-S.; Lee, J.-M. Towards Lightweight Intrusion Identification in SDN-based Industrial Cyber-Physical Systems. In Proceedings of the 2022 27th Asia Pacific Conference on Communications (APCC), Jeju Island, Republic of Korea, 19–21 October 2022; pp. 610–614. [CrossRef]
33. Yilmaz, M.N.; Bardak, B. An Explainable Anomaly Detection Benchmark of Gradient Boosting Algorithms for Network Intrusion Detection Systems. In Proceedings of the 2022 Innovations in Intelligent Systems and Applications Conference (ASYU), Antalya, Turkey, 7–9 September 2022; pp. 1–6. [CrossRef]
34. Firdaus, D.; Munadi, R.; Purwanto, Y. DDoS Attack Detection in Software Defined Network using Ensemble K-means++ and Random Forest. In Proceedings of the 2020 3rd International Seminar on Research of Information Technology and Intelligent Systems (ISRITI), Yogyakarta, Indonesia, 10–11 December 2020; pp. 164–169. [CrossRef]
35. Hnamte, V.; Hussain, J. Network Intrusion Detection using Deep Convolution Neural Network. In Proceedings of the 2023 4th International Conference for Emerging Technology (INCET), Belgaum, India, 26–28 May 2023; pp. 1–6. [CrossRef]
36. Ibrahimy, A.M.; Dewanta, F.; Aminanto, M.E. Lightweight Machine Learning Prediction Algorithm for Network Attack on Software Defined Network. In Proceedings of the 2022 IEEE Asia Pacific Conference on Wireless and Mobile (APWiMob), Bandung, Indonesia, 9–10 December 2022; pp. 1–6. [CrossRef]
37. Zainudin, A.; Akter, R.; Kim, D.-S.; Lee, J.-M. Federated Learning Inspired Low-Complexity Intrusion Detection and Classification Technique for SDN-Based Industrial CPS. *IEEE Trans. Netw. Serv. Manag.* **2023**, *20*, 2442–2459. [CrossRef]
38. Fu, G.; Li, B.; Yang, Y.; Wei, Q. A Multi-Distance Ensemble and Feature Clustering Based Feature Selection Approach for Network Intrusion Detection. In Proceedings of the 2022 International Symposium on Sensing and Instrumentation in 5G and IoT Era (ISSI), Shanghai, China, 17–18 November 2022; pp. 160–164. [CrossRef]
39. Vuong, T.-C.; Tran, H.; Trang, M.X.; Ngo, V.-D.; Luong, T.V. A Comparison of Feature Selection and Feature Extraction in Network Intrusion Detection Systems. In Proceedings of the 2022 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC), Chiang Mai, Thailand, 7–10 November 2022; pp. 1798–1804. [CrossRef]
40. Raman, A.; Kumar, S.; Arora, A. An Enhanced Intrusion Detection System Using Combinational Feature Ranking and Machine Learning Algorithms. In Proceedings of the 2022 2nd International Conference on Intelligent Technologies (CONIT), Hubli, India, 24–26 June 2022; pp. 1–8. [CrossRef]
41. Prastyo, P.H.; Ardiyanto, I.; Hidayat, R. A Review of Feature Selection Techniques in Sentiment Analysis Using Filter, Wrapper, or Hybrid Methods. In Proceedings of the 2020 6th International Conference on Science and Technology (ICST), Yogyakarta, Indonesia, 7–8 September 2020; pp. 1–6. [CrossRef]
42. Bubolz, T.; Grellert, M.; Zatt, B.; Correa, G. Coding Tree Early Termination for Fast HEVC Transrating Based on Random Forests. In Proceedings of the ICASSP 2019—2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Brighton, UK, 12–17 May 2019; pp. 1802–1806. [CrossRef]
43. Bakro, M.; Kumar, R.R.; Alabrah, A.; Ashraf, Z.; Ahmed, N.; Shameem, M.; Abdelsalam, A. An Improved Design for a Cloud Intrusion Detection System Using Hybrid Features Selection Approach with ML Classifier. *IEEE Access* **2023**, *11*, 64228–64247. [CrossRef]