Computer Systems Science and Engineering

Tech Science Press

DOI: 10.32604/csse.2024.056257

ARTICLE

Performance Analysis of Machine Learning-Based Intrusion Detection with Hybrid Feature Selection

Mohammad Al-Omari[1] **and Qasem Abu Al-Haija**[2,*]

[1]Department of Business Information Technology, Princess Sumaya University for Technology, Amman, 11941, Jordan

[2]Department of Cybersecurity, Faculty of Computer & Information Technology, Jordan University of Science and Technology, Irbid, 22110, Jordan

*Corresponding Author: Qasem Abu Al-Haija. Email: qsabuhaija@just.edu.jo

Received: 18 July 2024 Accepted: 14 September 2024 Published: 22 November 2024

ABSTRACT

More businesses are deploying powerful Intrusion Detection Systems (IDS) to secure their data and physical assets. Improved cyber-attack detection and prevention in these systems requires machine learning (ML) approaches. This paper examines a cyber-attack prediction system combining feature selection (FS) and ML. Our technique's foundation was based on Correlation Analysis (CA), Mutual Information (MI), and recursive feature reduction with cross-validation. To optimize the IDS performance, the security features must be carefully selected from multiple-dimensional datasets, and our hybrid FS technique must be extended to validate our methodology using the improved UNSW-NB 15 and TON_IoT datasets. Our technique identified 22 key characteristics in UNSW-NB-15 and 8 in TON_IoT. We evaluated prediction using seven ML methods: Decision Tree (DT), Random Forest (RF), Logistic Regression (LR), Naive Bayes (NB), K-Nearest Neighbors (KNN), Support Vector Machines (SVM), and Multilayer Perceptron (MLP) classifiers. The DT, RF, NB, and MLP classifiers helped our model surpass the competition on both datasets. Therefore, the investigational outcomes of our hybrid model may help IDSs defend business assets from various cyberattack vectors.

KEYWORDS

Machine learning; cybersecurity; cyberattacks; feature selection; classification; intrusion detection system

1 Introduction

At this point, information technologies constitute a vital element of the operations of most enterprises. Robust Intrusion Detection Systems (IDS) can identify and stop threatening network activities, dismissing cybersecurity risks [1,2]. They are an essential part of a comprehensive security strategy; they provide businesses with the necessary tools to detect, prevent, and respond to threats that could compromise both their digital and physical assets. As a result of the incremental number of zero-day attacks and the increasing amount of network traffic produced by cloud services and the Internet of Things (IoT) [3–5], it is becoming progressively challenging to distinguish between genuine and malicious behaviors. The learning capabilities of IDS are upgraded with the application of Machine Learning (ML) systems, which facilitates its use of past data to predict and reduce cyber threats. In this research, we utilize supervised ML models for binary classification to accurately identify network activities. We propose a flexible design and architecture that can adapt to new emergent cyberattacks by improving the IDS performance through feature selection (FS).

Supervised machine learning models make use of tools that were designed for binary classification. The model accurately distinguishes events as “normal” or “attack” using a binary classification [6]. Our models were trained and evaluated with the help of a large dataset, which we utilized. Using FS in situations like these is essential. Additionally, we improved the IDS model’s efficiency and accuracy by employing the most important components. An IDS architecture incorporating ML and FS methods is more resilient and flexible. This architecture can cope with the emergence of cyber-attacks.

When working with data holding irrelevant or redundant features, FS (choosing a subcategory of appropriate features) is vital for improving the predictive models’ efficiency and reducing the impact of irrelevant features; it can negatively influence predictive accuracy and require more computational resources [7]. The proposed hybrid FS method [8] solves these issues by integrating Correlation Analysis (CA), Mutual Information (MI), and Recursive Feature Elimination with Cross-Validation (RFECV). Furthermore, for a comprehensive analysis of the binary classification performance, we utilized various ML models such as Decision Tree (DT), Random Forest (RF), Logistic Regression (LR), Naive Bayes (NB), K-Nearest Neighbors (KNN), Support Vector Machines (SVM), and Multilayer Perceptron (MLP) on the UNSW-NB 15 [9] and TON_IoT datasets [10], which include contemporary attacks.

The primary purpose of this research is to expand our prior findings and provide a comprehensive performance analysis to anticipate cyber-attacks; we combine multiple ML algorithms with our hybrid FS strategy to assess its effectiveness in a broader context. The structure of this paper is as follows: Section 2 recaps relevant work on IDS models. Section 3 explains the methodology employed in this research, including the adapted IDS architecture. Section 4 presents the implementation of experiments along with the results. In addition, we analyze these results and compare them with similar works. The paper concludes in the final part, which summarizes our significant findings and suggests future directions.

2 Related Work

The UNSW-NB 15 and TON_IoT datasets are well-known in network intrusion research and have seen widespread use in recent years. These datasets are valuable because they include new attack patterns, making them perfect for evaluating the efficacy of the proposed IDSs. This section explores advances in ML-based IDSs, emphasizing FS methods. The evaluation of IDSs involves several factors, with particular attention given to the feature selection method used to differentiate between malicious and legitimate activities [11]. Sangaiah et al. [11] proposed a Hybrid Ant–Bee Colony Optimization (HABCO) approach, which converts the feature selection problem into an optimization task. They evaluated the performance of HABCO alongside other methods. The results revealed that HABCO surpassed the other methods, achieving significantly higher accuracy. Kasongo et al. [12] used XGBoost to choose ensemble features for IDSs. They used ML algorithms to do a performance analysis on the UNSWNB15 dataset. They found and picked the top 19 features out of 42 by sorting the feature relevance with XGBoost. Their findings revealed that employing XGBoost for FS in binary classification with DTs improved the accuracy by 1.9% compared to the baseline model’s use of all features.

Prasad et al. [13] used the UNSW-NB15 dataset to construct a multi-level correlation-based FS approach for IDSs. This approach involved a two-level FS process. Firstly, a Pearson correlation was applied to assess the correlation between features and between features and labels. The more correlated, redundant feature was eliminated if the correlation between any two features exceeded 0.9. Additionally, correlations between features and labels were used to filter out less important features. Their experiment ultimately chose 15 features for use in a DT model, which achieved a multi-classification accuracy of 95.2%. Unlike previous studies that used a pre-prepared 10% of the dataset for training and testing, Prasad et al. [13] utilized the entire dataset in their work.

Also, Awad et al. [14] introduced an improved FS technique, RFECV, utilizing a DT estimator (RFECV-DT). They highlighted the limitations of existing methods in this area. This new FS method was then used to train various advanced ML (models such as NB, LR, AdaBoost, RF, and MLP for IDSs). They employed the UNSW-NB15 dataset for their experiments. Their FS technique utilizing the RF classifier revealed an accuracy of 95.30% and a weighted F1-score of 95.29%. On the other hand, Subramani et al. [15] introduced a feature selection algorithm based on rules and Multi-Objective Particle Swarm Optimization (PSO). Furthermore, the researchers proposed an intelligent rule-based method that enhances the performance of Multi-Class Support Vector Machines for more accurate intruder identification. Experimental tests were conducted using the KDDCup99 and CIDD datasets. The results indicated that the proposed IDS significantly improves intruder detection accuracy while lowering the false-positive rates.

In their study, Alazzam et al. [16] suggested an FS method based on the behavioral patterns of pigeon groups called the Pigeon Inspired Optimizer (PIO). They developed Cosine PIO, an improved version of the PIO method that uses cosine similarity and compared its performance to that of the Sigmoid PIO variant. They employed the NSL-KDD, KDDCup99, and UNSW-NB15 datasets in their experiments. Across all three datasets, Cosine PIO beat Sigmoid PIO in binary classification tasks. Specifically, it selected five features from NSL-KDD, seven from KDDCup99, and five from UNSW-NB15, achieving accuracies of 88.3% in NSL-KDD, 96% in KDDCup99, and 91.7% in UNSW-NB15.

Yin et al. [17] used the UNSW-NB15 dataset in a study to develop a two-step FS approach for an anomaly-based network IDS. They used the RF and Information Gain algorithms to remove extraneous features. They then used an MLP classifier with RFECV to reduce the feature set to 23. Their proposed multi-classification model obtained an F1-score of 82.85% and an accuracy rate of 84.24%.

Gu et al. [18] consolidated the SVM with the NB techniques to effectively classify attacks and normal occurrences in IDSs. Initially, the NB technique was used to renovate the original features into a new dataset. This transformed data was successively used to train the SVM classifier model. They evaluated their model on four datasets: UNSW-NB 15, CICIDS2017, NSL-KDD, and Kyoto 2006+. Their empirical results were notable, recording accuracy rates of 98.92% on the CICIDS2017 dataset, 99.35% on the NSL-KDD dataset, 93.75% on the UNSW-NB 15 dataset, and 98.58% on the Kyoto 2006+ dataset.

Moustafa [10] advocated an inventive dataset called Network TON_IoT and employed the based wrapper FS method to realize the essential features. After that, they evaluated the performance of four ML models, such as Gradient Boosting Machine (GBM), RF, NB, and Deep Neural Networks (DNN). Their performance assessment revealed a comparable performance trajectory, with accuracy scores of 91.28%, 93.83%, 99.98%, and 99.92% for the NB, GBM, RF, and DNN models.

Elsayed et al. [19] introduced the Secured Automatic Two-level IDS (SATIDS), which was constructed utilizing an enhanced Long Short-Term Memory (LSTM) network. The purpose of SATIDS is to identify and classify attacks from benign traffic. Their model endured training and testing using two contemporary and comprehensive datasets: ToN-IoT and InSDN. Regarding performance, SATIDS displayed enhanced results, particularly for the ToN-IoT dataset, achieving an accuracy of 96.35%.

Guo et al. [20] developed an ML-based IDS framework for IoT networks using ten supervised machine-learning methods. They evaluated their IDS models using the TON_IoT dataset, a modern and inclusive dataset for IoT networks. According to their evaluation, the stacking-ensemble-based IDS model recorded the best performance figures, surpassing all of the IDS models investigated, achieving a 99.87% F1-score for binary classifications.

Disha et al. [1] employed the UNSW-NB 15 and Network TON_IoT datasets to assess the efficacy of several IDS algorithms. Their study investigated the efficacy of several binary classification models, including DT, gradient boosting trees, MLP, AdaBoost, long-and LSTM, and Gated Recurrent Unit (GRU). Also, the Gini Impurity-based Weighted Random Forest (GIWRF) algorithm was employed to select the relevant set of features using the Gini impurity metric to determine the tree partition policy. The weights were updated, considering the imbalanced distribution of classes. Accordingly, only 20 and 10 features were selected for the UNSW-NB-15 and TON_IoT datasets. The experimental assessment for DT models employing the selected features only confirmed an improved performance on both datasets. The model attained (93.01%, 99.90%), (94.76%, 99.87%), and (93.72%, 99.85%) for the accuracy, recall, and F1-score for the (UNSW-NB 15 dataset and TON_IoT dataset), respectively.

It has been reported several times that IDSs are vital in protecting computer networks, which has raised the issue of the need for efficient and automated IDSs due to the accelerating emergence of cyber-attacks and the increasing complexity of network and communication formations. The development of effective IDSs should ensure the precise recognition of unauthorized access attempts, dismiss occurrences of false alarms, and manage the substantial data sizes conceived by such systems. The accuracy of the diverse existing IDSs is significantly frustrated by their high False Positive Rates (FPR). While several state-of-the-art studies have demonstrated remarkable precision, this has often resulted in a plethora of computational complexity and data loss, leading to a drop in the efficiency of such systems.

Additionally, several IDSs rely on an independent dataset, limiting their capability to recognize diverse intrusions and hindering their capability to detect new attacks. In this research, we devoted several ML methods and a hybrid FS strategy to investigating two commonly recognized IoT-based cyberattack datasets. This study evaluated the efficiency of these strategies when applying the hybrid FS across different datasets and contrasted our findings with those of previous research. Table 1 summarizes the former studies on ML-based IDS reviewed in this research.

Table 1: Summary of contributions to ML-based IDS

Reference	Dataset	FS method	No. of features	Classifier	Accuracy
Kasongo et al. [12]	UNSW-NB 15	XGBoost	19	SVM, LR, KNN, DT, ANN	60.89% 77.64% 84.46% 90.85% 84.39%

(Continued)

Table 1 (continued)

Reference	Dataset	FS method	No. of features	Classifier	Accuracy
Prasad et al. [13]	UNSW-NB15	CA	15	DT	95.20%
Awad et al. [14]	UNSW-NB15	RFECV-DT	15	NB, LR, AdaBoost, RF, MLP	84.71% 84.79% 92.20% 95.30% 89.75%
Alazzam et al. [16]	UNSW-NB15 NSL-KDD KDDCup99	Sigmoid PIO Cosine PIO	5 5 7	DT	91.70% 88.30% 96.00%
Yin et al. [17]	UNSW-NB15	RFE & Information gain	23	MLP	84.24%
Gu et al. [18]	UNSW-NB15 CICIDS2017 NSL-KDD Kyoto 2006+	Transformation using Naïve Bayes	–	SVM	93.75% 98.2% 99.35% 98.58%
Moustafa [10]	TON_IoT	Wrapper FS technique using RF	–	RF GBM DNN	99.98% 93.83% 99.92%
Elsayed et al. [19]	TON_IoT	All features	–	LSTM	96.35%
Guo et al. [20]	TON_IoT	All features	–	Stacking-ensemble model	99.87%
Disha et al. [1]	UNSW-NB15 TON_IoT	GIWRF	20 10	DT	93.01% 99.90%

3 Methodology

This part provides an adapted architecture for our proposed ML-based IDS. This design is based on our hybrid FS method, described in detail in [8]. We tested the performance of this hybrid technique on two datasets using several ML algorithms. The revised IDS architecture is depicted in Fig. 1.

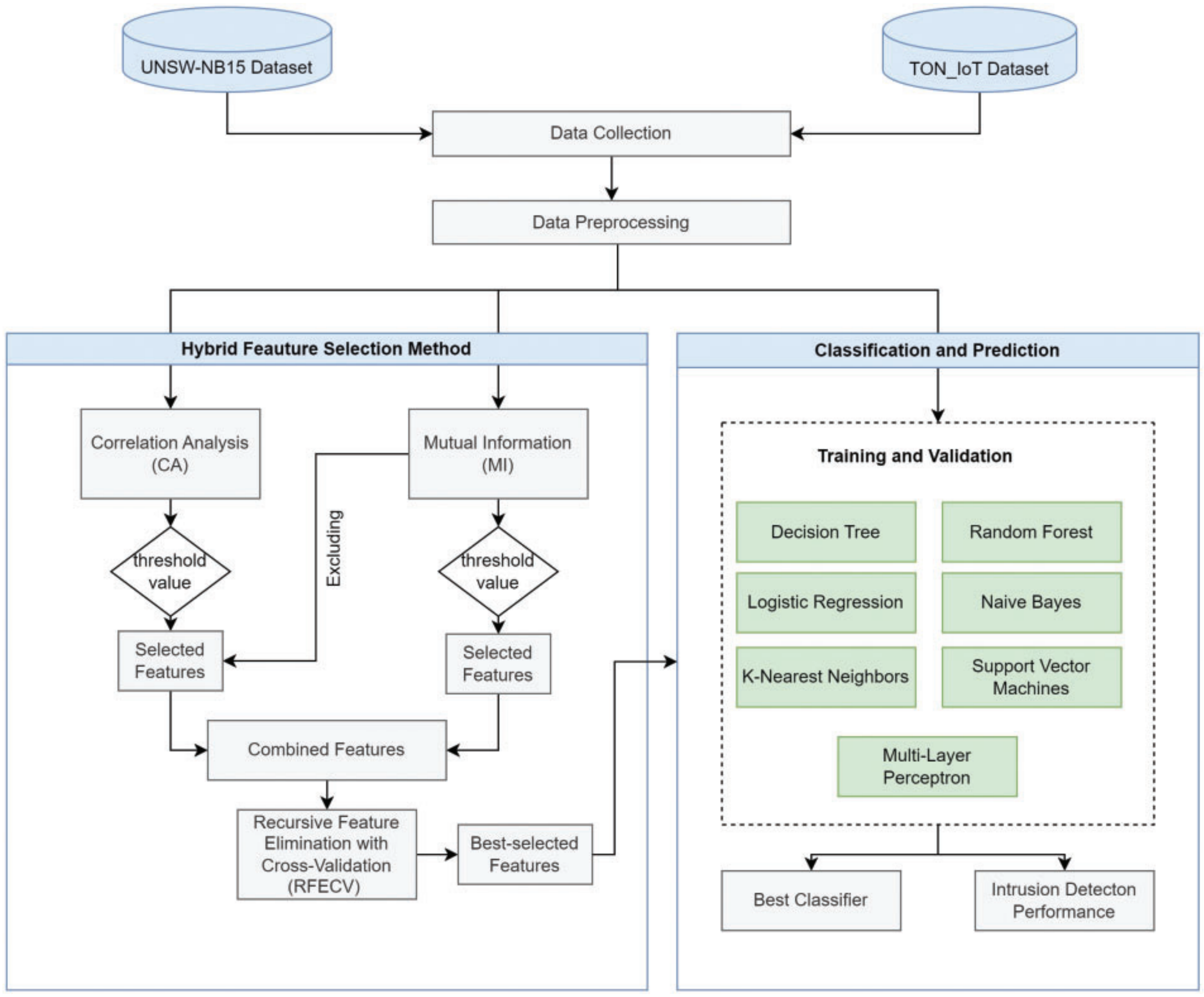

Figure 1: The IDS architecture, adapted with permission from [8]

3.1 The Datasets

In this paper, we use the UNSW-NB 15 and TON_IoT datasets to do a performance analysis of various ML algorithms using the hybrid FS strategy. These datasets are commonly utilized in security intrusion detection and have piqued the interest of ML-based IDS researchers [21,22]. The characteristics of these datasets are examined in detail in the following subsections.

3.1.1 The UNSW-NB15 Dataset

To address the shortcomings of the KDDCup99 dataset, Moustafa et al. [23] created the UNSW-NB15 dataset. Responding to concerns about the KDDCup99 dataset's outmoded attack types, imbalanced training and test sets, and lack of important network protocols, the UNSW-NB15 dataset includes more complicated and modern intrusion events and contemporary security mechanisms. The UNSW-NB15 dataset is divided into a training set with 175,341 records and a test set with 82,332 records, representing 10% of the entire dataset. It comprises 42 features, excluding the 'label' class that indicates normal activity or an attack. In this research, the feature indicating the type of attack was omitted, as it was not within the research scope.

Additionally, based on prior studies' recommendation to avoid classification biases [24,25], TTL-based features 'sttl,' 'dttl,' and 'ct_state_ttl' were excluded from the UNSW-NB15 dataset. Consequently, the number of security features considered was 39. 80% of the data was used for model training, while the remaining 20% was assigned for testing. Table 2 details the security features of the UNSW-NB15 dataset along with their data types. The description of each feature in this dataset can be retrieved from (https://research.unsw.edu.au/projects/unsw-nb15-dataset) (accessed on 13 September 2024).

Table 2: Security features of the UNSW-NB15 dataset

Feature	Type	Count
dur, rate, sload, dload, sinpkt, dinpkt, sjit, djit, tcprtt, synack, ackdat, spkts, dpkts, sbytes, dbytes, sttl, dttl, sloss, dloss, swin, stcpb, dtcpb, dwin, smean, dmean, trans_depth, response_body_len, ct_srv_src, ct_state_ttl, ct_dst_ltm, ct_src_dport_ltm, ct_dst_sport_ltm, ct_dst_src_ltm, ct_ftp_cmd, ct_flw_http_mthd, ct_src_ltm, ct_srv_dst.	Number	37
proto, service, state.	String	3
is_ftp_login, is_sm_ips_ports.	Boolean	2

3.1.2 The TON_IoT Dataset

The TON_IoT network dataset, created by Moustafa [10] in 2021, is a recent addition to the field. This dataset, created by the Intelligent Security Group of the Cyber Range and IoT Labs at UNSW, includes traffic from nine modern attack types. It comprises 22,339,021 instances, out of which a subset of 461,043 instances is labeled as the 'Train_Test_Network_dataset'. This subset was specifically designed to evaluate new Artificial Intelligence-based cybersecurity solutions. The dataset includes 43 features, excluding the 'label' and 'type' as class variables. Following Moustafa's recommendation [10], four features, source IP, destination IP, source ports, and destination ports, were excluded from the dataset. Furthermore, the 'type' feature, which focuses on binary classifications rather than multi-class classifications, was excluded in this study. Thus, the number of security features considered in this work was 39, excluding the 'label' class. For training and testing purposes, 80% of the data was used for training, while the remaining 20% was allocated for testing. Table 3 details the security features of the TON_IoT dataset along with their data types. The description of each feature in this dataset can be retrieved from (https://research.unsw.edu.au/projects/toniot-datasets) (accessed on 13 September 2024).

Table 3: Security features of the TON_IoT dataset

Feature	Type	Count
ts	Time	1

(Continued)

Table 3 (continued)

Feature	Type	Count
dns_AA, dns_RD, dns_RA, dns_rejected, ssl_resumed, ssl_established, weird_notice.	Boolean	7
proto, service, conn_state, dns_query, ssl_version, ssl_cipher, ssl_subject, ssl_issuer, http_method, http_uri, http_version, http_orig_mime_types, http_resp_mime_types, weird_name, weird_addl.	String	15
duration, src_bytes, dst_bytes, missed_bytes, src_pkts, src_ip_bytes, dst_pkts, dst_ip_bytes, dns_qclass, dns_qtype, dns_rcode, http_trans_depth, http_request_body_len, http_status_code, http_response_body_len, http_user_agent.	Number	16

3.1.3 Data Preprocessing

This step is pivotal for providing predictive models with precise and reliable data, which is vital for maintaining the overall accuracy of the models. Given that the datasets were free from any null or duplicate values, the data preprocessing phase in our study was confined to feature encoding and scaling. In this study, the method of Label Encoding was utilized for all categorical variables in both datasets. Each categorization value has a unique integer identifier [26]. This method enables the classifier to read and learn from numerical labels efficiently. The UNSW-NB15 dataset contains three categorical features, but the TON_IoT dataset contains fifteen. Certain ML models that compute the distance between data points require feature scaling. In ML, two extensively used approaches to feature scaling are normalization and standardization [27]. Normalization, specifically the Min-Max scaling approach, was used in this work to adapt the data values for both datasets to fall within a range of 0 to 1. The Min-Max scaling approach is represented in Eq. (1).

$$Feature_{Value\,Scaled} = \frac{(Feature\_value) - (Feature\_min\_value)}{(Feature\_max\_value) - (Feature\_min\_value)} \tag{1}$$

3.2 Hybrid Feature Selection (FS) Method

As discussed earlier in this paper, we used our proposed hybrid FS strategy [8]. CA, MI, and RFECV are three distinct FS approaches that are combined in this hybrid method. This method combines the benefits of the CA and MI filter methods, followed by the use of the RFECV wrapper method, which aids in identifying the most effective features. The CA and MI methods are employed as initial steps to reduce data dimensionality and speed up the RFECV process. By working with a smaller set of pre-filtered features, RFECV can select the most relevant features more efficiently. As the first filter method, the process begins by analyzing the linear associations between each characteristic of the dataset and the label class using the correlation coefficient. The following filtering process computes the MI scores for the remaining characteristics, omitting those chosen in the previous phase. This exclusion improves the performance of the MI computations. Accordingly, the features selected from the preceding methods are improved using the RFECV wrapper approach.

We refer the readers to [8] for further details and a discussion of our suggested hybrid technique. Besides, the thresholds for the CA and MI filter methods have undergone several trials to discover the most relevant features for attaining the best outcomes. Finally, they were set at (0.1, 0.1) and (0.3, 0.2) for (the UNSW-NB15 dataset and the TON_IoT dataset), respectively. Using the proposed hybrid FS model yielded 22 and 8 relevant features (for the UNSW-NB15 dataset and the TON_IoT dataset, respectively) that were appropriate for further processing by the ML models. It is worth mentioning that the optimum features from both datasets were selected using the wrapper technique, with the RF classifier subjected to 10-fold cross-validation to ensure its reliability. Furthermore, the wrapper approach selected the features with a rank of 1. Figs. 2 and 3 confirm the outcomes of using the hybrid strategy to select the most relevant features from the UNSW-NB15 and TON_IoT datasets.

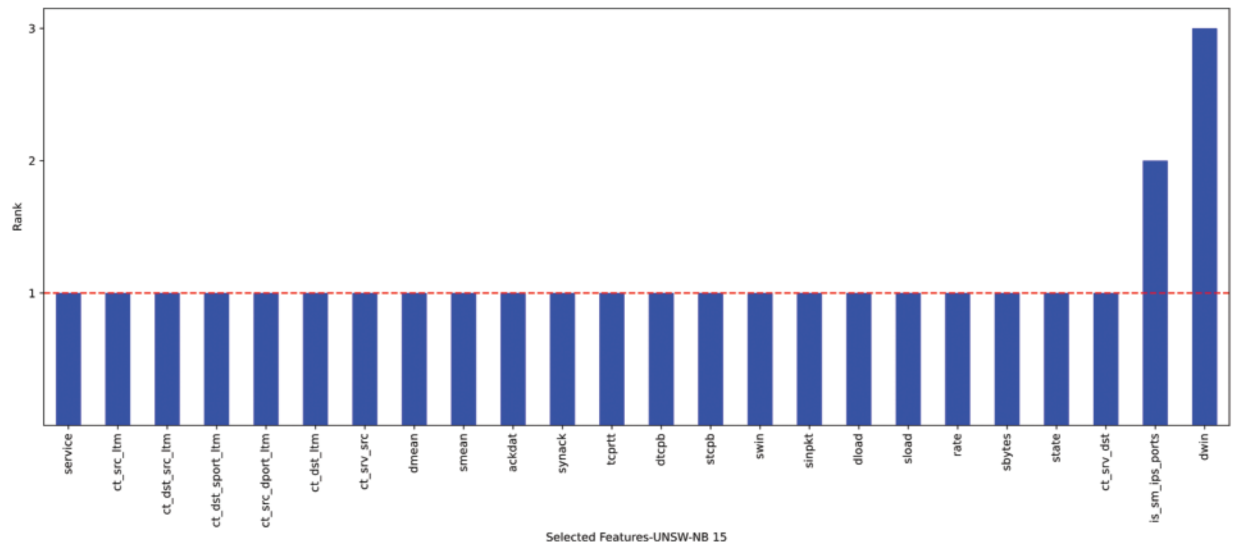

Figure 2: Optimal features from the UNSW-NB15 dataset

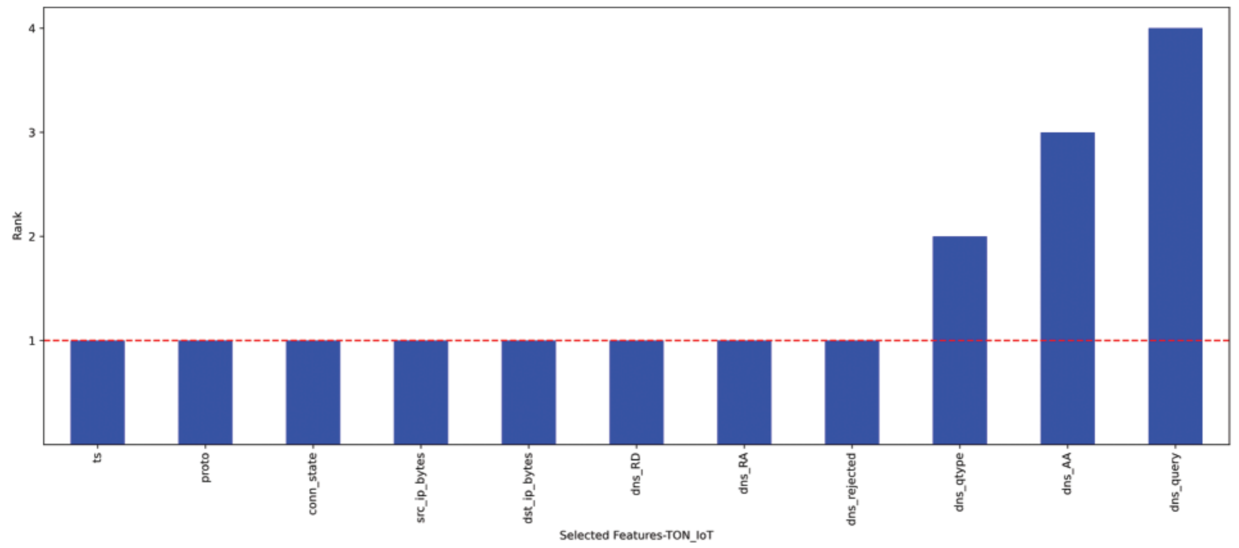

Figure 3: Optimal features from the TON_IoT dataset

3.3 ML Models

On the UNSW-NB15 and TON_IoT datasets, a range of supervised ML models was tested, both with and without, using our hybrid FS strategy. The goal was to examine the performance of these models when combined with the hybrid technique, emphasizing accuracy and other assessment metrics (which will be explained in the following section). These models were chosen due to their widespread use in the security arena, where they have demonstrated substantial efficacy in the construction of IDSs and have proven efficient in various other areas, as indicated in [28,29]. DT, RF, LR, NB, KNN, SVM, and MLP were the ML models used in this work, with their default parameters employed in Python. Each model is briefly described in the points that follow.

Decision Tree (DT): A popular machine-learning technique that uses a tree-like representation of decisions and their potential outcomes. It separates the dataset into branches according to a certain criterion at each level, leading to a decision node that predicts the output [30]. Because of its interpretability and simple structure, this method is particularly good for classification and regression applications. The decision tree is structured recursively by selecting the best feature to split the data at each step. The "best" feature and threshold are decided by criteria such as Gini impurity. The Gini impurity for a node t is calculated using Eq. (2).

$$Gini\,(t) = 1 - \sum_{i=1}^{c} p\,(i|t)^2 \tag{2}$$

where c is the number of classes and $p\,(i|t)$ is the proportion of samples of class i at node t.

Random Forest (RF): An ensemble learning method that creates many DTs during training and outputs the mean prediction (regression) or mode of the classes (classification) of the individual trees [30]. By averaging or merging the outputs of several trees, RF reduces the risk of overfitting, a typical shortcoming in DTs, and hence improves the predicted accuracy and robustness of the model. The "random" in RF originates from the point that each tree is trained on a random subset of the data. After that, the outcome is calculated as shown in Eq. (3).

$$Final\ Prediction = \begin{cases} M_o = argmax_c\,(count\,(c)) : \textit{If the task is "Classification"} \\ \\ M_e = \dfrac{1}{n}\sum_{i=1}^{n} p_i \qquad : \textit{If the task is "Regression"} \end{cases} \tag{3}$$

where:

- M_o is the mode (the predicted class that occurs most frequently), c represents each unique class in the set of predictions, and $argmax_c$ denotes the class c that maximizes the count.
- M_e is the average of the predictions; n is the total number of predictions; and p_i represents each prediction in the set.

Logistic Regression (LR): A popular statistical model for binary categorization. It employs a logistic function to estimate the likelihood of a binary response based on one or more predictor factors [31]. In circumstances when the outcome to be predicted is dichotomous, LR is popular due to its simplicity and efficiency. The cost function for LR calculates the difference between the predicted and the actual class labels as shown in Eq. (4).

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m} \left[y^{(i)} \log\left(h_0\left(x^{(i)}\right)\right) + \left(1 - y^{(i)}\right) \log\left(1 - h_0\left(x^{(i)}\right)\right)\right] \tag{4}$$

where:

- $J(\theta)$ is the cost function.
- m is the number of training examples.
- $h_0\left(x^{(i)}\right)$ is the predicted probability for the i-th example.
- $y^{(i)}$ is the actual class label for the i-th example.

Naïve Bayes (NB): NBs are ML-based probabilistic classifiers that are based on Bayes' theorem with strong (naive) independence assumptions between features [32]. They are commonly known for their efficiency and effectiveness in huge datasets. Despite their simplicity, NB classifiers are effective in many difficult real-world situations. For a classification problem with features $X_1, X_2, \ldots, X_{\mathrm{n},}$ and a class label C, the NB classifier involves calculating the probabilities shown in Eq. (5).

$$P(C \mid X_1,\ X_2, \ldots, X_n) \propto P(C) \cdot \prod_{i=1}^{n} P(X_i \mid C) \tag{5}$$

where:

- $P(C \mid X_1,\ X_2, \ldots, X_n)$ is the posterior probability of class C given the features.
- $P(C)$ is the prior probability of class C.
- $P(X_i \mid C)$ is the likelihood of feature X_i given class C.

K-Nearest Neighbors (KNN): A non-parametric, instance-based learning method mostly used for classification and regression. It predicts the output based on the k-nearest data points in the training dataset to a new data point [31]. KNN is well known for its simplicity and effectiveness, particularly in cases with irregular decision boundaries. Given a new instance X_{new} to be classified, then, the distance (e.g., Euclidean distance) between X_{new} and all instances (X_i) in the training set can be computed using Eq. (6) as follows:

$$Distance(X_{new},\ X_i) = \sqrt{\sum_{j=1}^{n} (X_{new,\,j} - X_{i,\,j})^2} \tag{6}$$

The k-nearest neighbors of X_{new} can be found by selecting the k instances from the training set with the smallest distances to X_{new} as shown in Eq. (7), and the predicted class c is calculated using Eq. (8).

$$kNN(X_{new}) = Min_{k\ out\ of\ n}\left\{Distance(X_{new}, X_i) = \sqrt{\sum_{j=1}^{n} (X_{new,\,j} - X_{i,\,j})^2}\right\} \tag{7}$$

$$Predicted\ Class = argmax_c\,(count\,(c)) \tag{8}$$

Support Vector Machines (SVM): SVM is a supervised learning approach that has proven its robustness and efficiency in high-dimensional domains. SVMs are specifically worthwhile for classification and regression tasks. SVM functions by revealing the optimal hyperplane that maximizes the margin between distinctive classes in the dataset, delivering efficient classification [30]. Given a dataset with features X and class labels y ($where\ y \in \{-1,\ 1\}$), then the decision function $f(X)$ and the optimization function $SVM(w,\ b)$ for linear SVM can be computed as shown in Eqs. (9) and (10).

$$f(X) = sign(w \cdot X + b) \tag{9}$$

$$SVM\,(w,\ b) = minimize\left\{\frac{1}{2}\,\|W\|^2 + C\sum_{i=1}^{m} \max(0,\ 1 - y^{(i)}(W\,.\,X^{(i)} + b))\right\} \tag{10}$$

where:

- $f(X)$ is the decision function, W is the weight vector, X is the feature vector, and b is the bias term.
- $\|W\|^2$ is the squared norm of the weight vector, C is the regularization parameter, and $y^{(i)}$ is the class label of the i-th instance.

Multilayer Perceptron (MLP): MLP is a feedforward Artificial Neural Network (ANN) that has at least three layers of artificial neurons: an input layer, a hidden (processing) layer, and an output layer [33]. MLP can be trained using a backpropagation algorithm, which enhances its ability to handle obscured pattern recognition and classification tasks (in case the simpler linear models are inadequate). Also, to configure the neural network parameters, we exploited the Adam Optimizer Algorithm predefined in the Python/sklearn.neural_network; regularization was applied with the default alpha value of 0.0001, the initial learning rate was set to 0.001, and the maximum number of iterations was 200. The use of MLP encompasses three stages of computation: forward propagation, backpropagation, and update weights and biases, which are calculated as shown in Eqs. (11)–(13).

Forward Propagation:

$$Z^{(l)} = g^{(l)}\left(W^{(l)} \cdot Z^{(l-1)} + B^{(l)}\right) \tag{11}$$

Backpropagation:

$$\begin{aligned} \delta^{(l)} &= g'^{(l)}\left(A^{(l)}\right) \odot \left(W^{(l+1)T} \cdot \delta^{(l+1)}\right) \\ \frac{\partial J}{\partial W^{(l)}} &= \delta^{(l)} \cdot \left(Z^{(l-1)}\right)^T \\ \frac{\partial J}{\partial B^{(l)}} &= \delta^{(l)} \end{aligned} \tag{12}$$

Update Weights and Biases:

$$\begin{aligned} W^{(l)} &= W^{(l)} - \alpha \frac{\partial J}{\partial W^{(l)}} \\ B^{(l)} &= B^{(l)} - \alpha \frac{\partial J}{\partial B^{(l)}} \end{aligned} \tag{13}$$

where $W^{(l)}$ is the weight matrix, $B^{(l)}$ is the bias vector, $Z^{(l-1)}$ is the output of the previous layer, and $g^{(l)}$ is the activation function.

3.4 Evaluation Metrics for ML Models

In this section, we report on the performance assessment phase of the proposed system using five standard evaluation criteria: confusion matrix (true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN)), accuracy, precision, recall, and F1-score [34]. In addition, the 'Fit time' metric was measured for all models to represent the average training duration the classifier used to train on the dataset. The 'fit()' function in the Python/sci-kit-learn library measures the time.

- **Accuracy** is the proportion of true results (TP and TN) among the cases examined. It provides a general measure of a model's effectiveness. The accuracy is calculated as shown in Eq. (14).

$$Accuracy = \frac{TP + TN}{Total\ Instances} \tag{14}$$

- **Precision** measures the accuracy of positive predictions, that is, the fraction of TP among all positive predictions made by the classifier. It is calculated as per Eq. (15).

$$Precision = \frac{TP}{TP + FP} \tag{15}$$

- **Recall**, also known as Sensitivity, quantifies the fraction of actual positives correctly identified, essentially the TP rate within the Attack class, as outlined in Eq. (16).

$$Recall = \frac{TP}{TP + FN} \tag{16}$$

- **F1-score** balances Precision and Recall, considering both FP and FN. The harmonic mean of the two metrics indicates the balance between them, as shown in Eq. (17).

$$F1 - score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{17}$$

4 Experiments and Results

This research utilized Python version 3.9.7 in a Jupyter Notebook setting for all experiments. Moreover, the experiments were executed on a laptop with Windows 11 Enterprise 64-bit OS, equipped with 16 GB of RAM and an Intel(R) Core(TM) i7-1065G7 CPU with a base clock speed of 1.5 GHz. Our experiments were designed to train and test seven machine-learning models on the UNSW-NB15 and TON_IoT datasets. Specifically, these ML models were first trained and tested using all features of each dataset. Subsequently, they underwent training and testing on the optimal features of each dataset, identified using our hybrid FS approach. A performance analysis was also conducted to compare these models on each dataset.

4.1 The Experiment on the UNSW-NB15 Dataset

In this experiment, the ML models selected for this research were trained and tested on the UNSW-NB15 dataset. The initial phase involved training and testing these models on the dataset comprising 39 features, excluding the 'Label' as detailed in Section 3. In the second phase, the models were trained and tested using only the optimal features obtained from the hybrid FS method, which are 22 optimal features. The results of this experiment are shown in Table 4.

Table 4: Results of the experiment on UNSW-NB15

Classifier	Phase 1: All features (39)					Phase 2: Optimal features (22)				
	Accuracy	Precision	Recall	F1-score	Fit time (s)	Accuracy	Precision	Recall	F1-score	Fit time (s)
DT	96.42%	96.42%	96.42%	96.42%	0.94	96.54%	96.54%	96.54%	96.54%	0.5
RF	97.76%	97.76%	97.76%	97.76%	9.9	97.81%	97.81%	97.81%	97.81%	7
LR	85.32%	85.36%	85.32%	85.33%	1.89	81.45%	81.53%	81.45%	81.47%	0.61
NB	73.95%	74.55%	73.95%	74.02%	0.07	74.54%	75.14	74.54%	74.83	0.04
KNN	92.60%	92.80%	92.60%	92.61%	8.35	92.24%	92.42%	92.24%	92.26%	0.51
SVM	92.85%	92.85%	92.85%	92.84%	127.6	92.73%	92.73%	92.73%	92.73%	89.6
MLP	95.77%	95.80%	95.77%	95.78%	167.7	96.37%	96.40%	96.37%	96.37%	99.6

Upon analyzing the experimental results, as detailed in Table 4, we observed the performance of seven ML classifiers on the UNSW-NB15 dataset across two distinct phases. The first phase utilized all 39 features, whereas the second phase employed a refined set of 22 optimal features. In Phase 1, the RF classifier demonstrated the highest effectiveness, with accuracy, precision, recall, and an F1-score of 97.76%. The SVM and MLP also showed strong performance metrics; however, they were notably more time-intensive, with Fit times of 127.6 and 167.7 s, respectively. In contrast, the NB classifier performed the least well with lower accuracy and F1-score metrics of around 74%, but it was the fastest model, with a Fit time of only 0.07 s. The second phase, which involved the optimal features, presented a general trend of improved classifier efficiency, with reduced Fit times across all models. The DT classifier showed a slight increase in all metrics, notably achieving an Accuracy and F1-score of 96.54%. It is important to note that the LR classifier's metrics decreased in this phase, suggesting that our hybrid FS method may have omitted significant variables for this model's performance. It is also worth mentioning that the MLP classifier improved the accuracy to 96.37% and saw a reduction in Fit time.

4.2 The Experiment on the TON_IoT Dataset

In this experiment, the TON_IoT dataset was utilized to train and test our selected ML models, following the same steps as our earlier experiment. In the first phase, the models were trained and tested on the original dataset comprising 39 features, excluding the 'Label' as detailed in Section 3. Then, in the second phase, the models were trained and tested using the 8 optimal features obtained from the hybrid FS method. The results of this experiment are shown in Table 5.

Table 5: Results of the experiment on TON_IoT

Classifier	Phase 1: All features (39)					Phase 2: Optimal features (22)				
	Accuracy	Precision	Recall	F1-score	Fit time (s)	Accuracy	Precision	Recall	F1-score	Fit time (s)
DT	99.91%	99.91%	99.91%	99.91%	1.3	99.98%	99.98%	99.98%	99.98%	0.64
RF	99.96%	99.96%	99.96%	99.96%	28.5	99.98%	99.98%	99.98%	99.98%	19
LR	87.73%	89.57%	87.73%	87.96%	13	87.69%	89.55%	87.69%	87.93%	3.2
NB	45.78%	77.77%	45.78%	38.30%	0.23	78.94%	84.21%	78.94%	79.41%	0.1
KNN	99.88%	99.88%	99.88%	99.88%	545	99.91%	99.91%	99.91%	99.91%	120
SVM	90.82%	91.05%	90.82%	90.89%	15089	89.98%	91.90%	89.98%	90.18%	3600
MLP	97.99%	98.05%	97.99%	98%	636.8	98.35%	98.40%	98.35%	98.36%	567

Table 5 presents the results of applying various ML classifiers to the TON_IoT dataset, which were structured into two phases. Phase 1 encompassed training and testing with all 39 features, while Phase 2 used a reduced set of 8 optimal features. In Phase 1, the DT and RF classifiers showed exceptional performance, achieving near-perfect scores across all metrics (99.91% and 99.96%, respectively, in Accuracy, Precision, Recall, and F1-score). The KNN classifier performed remarkably well, with 99.88% across the same metrics. However, the NB classifier exhibited significantly lower performance, with an accuracy of 45.78% and an F1-score of 38.30%. The SVM and MLP classifiers showed strong accuracy (90.82% and 97.99%, respectively), but their fit times were notably longer. Phase 2, where only optimal features were used, revealed an overall increase in efficiency, with reduced fit times for all classifiers.

Notably, the DT and RF classifiers maintained high performance, with slight improvements in all metrics, achieving accuracy and an F1-score of 99.98%. The LR classifier showed a minor decrease in performance but improved efficiency, reducing its fit time from 13 to 3.2 s. The NB classifier's performance improved in Phase 2, demonstrating the benefit of our hybrid FS method. The KNN classifier retained the high performance with a more efficient fit time. While their performance remained robust for the SVM and MLP classifiers, the fit times decreased substantially, particularly for SVM, which dropped from 15,089 to 3600 s, still indicating a considerable computational requirement.

To that aim, our findings highlight the influence of our hybrid FS technique on the efficiency and effectiveness of ML classifiers, specifically in the context of cyber threat detection on the UNSW-NB15 and TON_IoT datasets. The results indicate that classifiers can achieve comparable or improved performance using the hybrid FS strategy, increasing computing efficiency. This improvement is critical in practical applications where computational resources and response time are of the essence.

4.3 Comparison with Similar Work

In the field, numerous studies have proposed ML-based IDSs employing a variety of FS methods and ML techniques. While some of these studies focused on a single dataset, others extended their research to multiple datasets. However, this study compares our findings with those from similar research that utilized the datasets examined in this paper.

Regarding the UNSW-NB15 dataset, our research revealed that when used with our hybrid FS method and 22 optimal features, our RF classifier achieved an accuracy of 97.81%. This performance surpasses that of Awad et al. [14], who reported an accuracy of 95.30% using the RF classifier with the RFECV-DT FS method and 15 features. Furthermore, the MLP classifier in our study demonstrated superior performance with an accuracy of 96.37%, compared to the 89.75% obtained in their research.

Comparing our results with the work of Prasad et al. [13], our DT classifier using the hybrid method outperformed their results, delivering an accuracy of 96.54% *vs.* 95.20%. Their study employed a CA for FS, yielding 15 optimal features. In the research by Kasongo et al. [12], which used the XGBoost method for FS and obtained 19 features; our classifiers, SVM, LR, KNN, and DT, performed better than their reported results. The highest accuracy in their study was with the DT classifier at 90.85%, while our DT classifier reached an accuracy of 96.54%. Additionally, the outcomes achieved using our DT and MLP classifiers were better than those reported by Alazzam et al. [16] and Yin et al. [17]. It is worth mentioning here that Alazzam et al. [16] utilized only 5 features, resulting in a better fit time with excellent accuracy of 91.70%. Table 6 summarizes the comparison results based on the UNSW-NB15 dataset, with the highest accuracies highlighted in bold within the Accuracy column.

For the TON_IoT dataset, our findings indicate that the RF and DT classifiers, utilizing our hybrid FS method with 8 optimal features, attained an accuracy of 99.98%. These results are comparable to those achieved by Moustafa [10], who used a wrapper method with RF. However, our DT classifier surpassed the performance achieved by Disha et al. [1], who reported an accuracy of 99.90% using the GIWRF FS method with 10 features, in contrast to our hybrid approach with 8 features. Elsayed et al. [19] achieved a commendable accuracy of 96.35% using the LSTM classifier on the original dataset. Additionally, Guo et al. [20] implemented a stacking-ensemble model on the full dataset, achieving an excellent accuracy of 99.87%. Table 7 summarizes the results of this comparison on the TON_IoT dataset.

Table 6: Performance comparison of the UNSW-NB15 dataset

Work	Classifier	FS method	No. of features	Accuracy
Awad et al. [10]	RF	DT-RFECV	15	**95.30%**
	MLP			89.75%
Prasad et al. [9]	DT	CA	15	95.20%
Kasongo et al. [8]	DT	XGBoost	19	**90.85%**
	KNN			84.46%
	LR			77.64%
	SVM			60.89%
Alazzam et al. [11]	DT	Sigmoid PIO	5	**91.70%**
Yin et al. [12]	MLP	RFE and information gain	23	**84.24%**
Our work	RF	Hybrid method (CA, MI, and RF-RFECV)	22	**97.81%**
	DT			96.54%
	MLP			96.37%
	SVM			92.73%
	KNN			92.24%
	LR			81.45%
	NB			74.54%

Table 7: Performance comparison of the TON_IoT dataset

Work	Classifier	FS method	No. of features	Accuracy
Awad et al. [14]	RF	DT-RFECV	15	**95.30%**
	MLP			89.75%
Prasad et al. [13]	DT	CA	15	95.20%
Kasongo et al. [12]	DT	XGBoost	19	**90.85%**
	KNN			84.46%
	LR			77.64%
	SVM			60.89%
Alazzam et al. [16]	DT	Sigmoid PIO	5	**91.70%**
Yin et al. [17]	MLP	RFE and information gain	23	**84.24%**
Our work	RF	Hybrid method (CA, MI, and RF-RFECV)	22	**97.81%**
	DT			96.54%
	MLP			96.37%
	SVM			92.73%
	KNN			92.24%
	LR			81.45%
	NB			74.54%

To this end, it is important to note that while our proposed IDS has demonstrated promising results, its performance has not yet been extensively tested across a broader range of datasets. Furthermore, incorporating more advanced machine learning and deep learning algorithms could offer valuable insights into the effectiveness of the proposed IDS with our hybrid FS strategy in various contexts.

5 Conclusion and Future Work

In this paper, we introduce an innovative hybrid FS approach that effectively boosts the effectiveness of IDS. The proposed approach incorporates the RFECV, CA, and MI techniques to determine the most advantageous features. This method reduces the dimensionality of datasets incontrovertibly while retaining crucial discriminatory information. Accordingly, 22 and 8 optimal features were identified by applying this method to the UNSW-NB15 and Network TON IoT datasets. After utilizing the reduced datasets, we trained and evaluated seven machine-learning models: MLP, DT, RF, LR, and NB. Our results demonstrate the efficacy of our hybrid FS approach. The RF classifier outperformed all alternative models with an impressive accuracy of 97.81% when applied to the UNSW-NB15 dataset. In the same way, the DT and RF classifiers demonstrated their efficacy in managing distinct network traffic attributes by attaining an outstanding accuracy rate of 99.98% on the TON_IoT dataset. In addition, the training period for every classifier was substantially diminished by our method. The accuracy of the DT, RF, NB, and MLP classifiers on both datasets was enhanced by our hybrid method compared to prior research. The efficacy of our methodology in improving binary classification to predict cyber-attacks on a wide range of datasets is underscored by these results. In the upcoming phase, we aim to evaluate the performance of our hybrid approach in identifying a broader range of cyber-attacks and applying it to multi-class classification tasks.

Acknowledgement: We express our gratitude to our families and colleagues for their moral support.

Funding Statement: The authors received no specific funding for this study.

Author Contributions: The authors confirm the contributions to the paper as follows: study conception and design: Mohammad Al-Omari; data collection: Mohammad Al-Omari and Qasem Abu Al-Haija; analysis and interpretation of results: Mohammad Al-Omari; draft manuscript preparation: Mohammad Al-Omari and Qasem Abu Al-Haija. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: The UNSW-NB 15 dataset is publicly available at: (https://research.unsw.edu.au/projects/unsw-nb15-dataset, accessed on 16 May 2024). The Network TON_IoT dataset is publicly available at: (https://research.unsw.edu.au/projects/toniot-datasets, accessed on 16 May 2024).

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] R. A. Disha and S. Waheed, "Performance analysis of machine learning models for intrusion detection system using Gini Impurity-based Weighted Random Forest (GIWRF) feature selection technique," *Cybersecurity*, vol. 5, no. 1, pp. 1–22, Dec. 2022. doi: 10.1186/s42400-021-00103-8.

[2] S. A. Alghamdi, "Novel trust-aware intrusion detection and prevention system for 5G MANET-cloud," *Int. J. Inf. Secur.*, vol. 21, no. 3, pp. 469–488, Jun. 2022. doi: 10.1007/s10207-020-00531-6.

[3] R. M. Yadav, "Effective analysis of malware detection in cloud computing," *Comput. Secur.*, vol. 83, pp. 14–21, Jun. 2019. doi: 10.1016/j.cose.2018.12.005.

[4] A. Javadpour, P. Pinto, F. Ja'fari, and W. Zhang, "DMAIDPS: A distributed multi-agent intrusion detection and prevention system for cloud IoT environments," *Cluster Comput.*, vol. 26, no. 1, pp. 367–384, Feb. 2023. doi: 10.1007/s10586-022-03621-3.

[5] L. Qi, Y. Liu, Y. Zhang, X. Xu, M. Bilal and H. Song, "Privacy-aware point-of-interest category recommendation in Internet of Things," *IEEE Internet Things J.*, vol. 9, no. 21, pp. 21398–21408, Nov. 2022. doi: 10.1109/JIOT.2022.3181136.

[6] S. Y. Diaba, M. Shafie-Khah, and M. Elmusrati, "Cyber security in power systems using meta-heuristic and deep learning algorithms," *IEEE Access*, vol. 11, pp. 18660–18672, 2023. doi: 10.1109/ACCESS.2023.3247193.

[7] Y. Lyu, Y. Feng, and K. Sakurai, "A survey on feature selection techniques based on filtering methods for cyber attack detection," *Information*, vol. 14, no. 3, 2023, Art. no. 191. doi: 10.3390/info14030191.

[8] M. Al-Omari and Q. A. Al-Haija, "Towards robust IDSs: An integrated approach of hybrid feature selection and machine learning," *J. Internet Serv. Inf. Secur.*, vol. 14, no. 2, pp. 47–67, 2024. doi: 10.58346/JISIS.2024.I2.004.

[9] N. Moustafa, B. Turnbull, and K. K. R. Choo, "An ensemble intrusion detection technique based on proposed statistical flow features for protecting network traffic of internet of things," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4815–4830, Jun. 2019. doi: 10.1109/JIOT.2018.2871719.

[10] N. Moustafa, "A new distributed architecture for evaluating AI-based security systems at the edge: Network TON_IoT datasets," *Sustain. Cities Soc*, vol. 72, Sep. 2021, Art. no. 102994. doi: 10.1016/j.scs.2021.102994.

[11] A. K. Sangaiah, A. Javadpour, F. Ja'fari, P. Pinto, W. Zhang and S. Balasubramanian, "A hybrid heuristics artificial intelligence feature selection for intrusion detection classifiers in cloud of things," *Cluster Comput.*, vol. 26, no. 1, pp. 599–612, Feb. 2023. doi: 10.1007/s10586-022-03629-9.

[12] S. M. Kasongo and Y. Sun, "Performance analysis of intrusion detection systems using a feature selection method on the UNSW-NB15 dataset," *J. Big Data*, vol. 7, no. 1, pp. 1–20, Dec. 2020. doi: 10.1186/s40537-020-00379-6.

[13] M. Prasad, R. K. Gupta, and S. Tripathi, "A multi-level correlation-based feature selection for intrusion detection," *Arab J. Sci. Eng.*, vol. 47, no. 8, pp. 10719–10729, Aug. 2022. doi: 10.1007/s13369-022-06760-2.

[14] M. Awad and S. Fraihat, "Recursive feature elimination with cross-validation with decision tree: Feature selection method for machine learning-based intrusion detection systems," *J. Sens. Actuator Netw.*, vol. 12, no. 5, Sep. 2023, Art. no. 67. doi: 10.3390/jsan12050067.

[15] S. Subramani and M. Selvi, "Multi-objective PSO based feature selection for intrusion detection in IoT based wireless sensor networks," *Optik*, vol. 273, Feb. 2023, Art. no. 170419. doi: 10.1016/j.ijleo.2022.170419.

[16] H. Alazzam, A. Sharieh, and K. E. Sabri, "A feature selection algorithm for intrusion detection system based on pigeon inspired optimizer," *Expert Syst. Appl.*, vol. 148, Jun. 2020, Art. no. 113249. doi: 10.1016/J.ESWA.2020.113249.

[17] Y. Yin *et al.*, "IGRF-RFE: A hybrid feature selection method for MLP-based network intrusion detection on UNSW-NB15 dataset," *J. Big Data*, vol. 10, no. 1, pp. 1–26, Dec. 2023. doi: 10.1186/s40537-023-00694-8.

[18] J. Gu and S. Lu, "An effective intrusion detection approach using SVM with naïve Bayes feature embedding," *Comput. Secur.*, vol. 103, Apr. 2021, Art. no. 102158. doi: 10.1016/j.cose.2020.102158.

[19] R. A. Elsayed, R. A. Hamada, M. I. Abdalla, and S. A. Elsaid, "Securing IoT and SDN systems using deep-learning based automatic intrusion detection," *Ain Shams Eng. J.*, vol. 14, no. 10, Oct. 2023, Art. no. 102211. doi: 10.1016/J.ASEJ.2023.102211.

[20] G. Guo, X. Pan, H. Liu, F. Li, L. Pei and K. Hu, "An IoT intrusion detection system based on TON IoT network dataset," in *Proc. 2023 IEEE 13th Annu. Comput. Commun. Workshop Conf., CCWC 2023*, 2023, pp. 333–338. doi: 10.1109/CCWC57344.2023.10099144.

[21] R. Anushiya and V. S. Lavanya, "A new deep-learning with swarm based feature selection for intelligent intrusion detection for the Internet of Things," *Meas. Sens.*, vol. 26, Apr. 2023, Art. no. 100700. doi: 10.1016/j.measen.2023.100700.

[22] R. Elsayed, R. Hamada, M. Hammoudeh, M. Abdalla, and S. A. Elsaid, "A hierarchical deep learning-based intrusion detection architecture for clustered Internet of Things," *J. Sens. Actuator Netw.*, vol. 12, no. 1, Dec. 2022, Art. no. 3. doi: 10.3390/jsan12010003.

[23] N. Moustafa and J. Slay, "UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," in *Proc. 2015 Mil. Commun. Inform. Syst. Conf., MilCIS 2015*, Canberra, ACT, Australia, Dec. 2015, pp. 1–6. doi: 10.1109/MILCIS.2015.7348942.

[24] M. Sarhan, S. Layeghy, and M. Portmann, "Feature analysis for machine learning-based IoT intrusion detection," Aug. 2021. doi: 10.21203/rs.3.rs-2035633/v1.

[25] S. Fraihat, S. Makhadmeh, M. Awad, M. A. Al-Betar, and A. Al-Redhaei, "Intrusion detection system for large-scale IoT NetFlow networks using machine learning with modified arithmetic optimization algorithm," *Internet of Things*, vol. 22, Jul. 2023, Art. no. 100819. doi: 10.1016/j.iot.2023.100819.

[26] E. Jackson and R. Agrawal, "Performance evaluation of different feature encoding schemes on cyber-security logs," in *Proc. IEEE SOUTHEASTCON*, Huntsville, AL, USA, Apr. 2019, pp. 1–9. doi: 10.1109/SOUTHEASTCON42311.2019.9020560.

[27] V. N. G. Raju, K. P. Lakshmi, V. M. Jain, A. Kalidindi, and V. Padma, "Study the influence of normalization/transformation process on the accuracy of supervised classification," in *Proc. 3rd Int. Conf. Smart Syst. Inven. Technol. ICSSIT 2020*, Aug. 2020, pp. 729–735. doi: 10.1109/ICSSIT48917.2020.9214160.

[28] A. L. Buczak and E. Guven, "A survey of data mining and machine learning methods for cyber security intrusion detection," *IEEE Commun. Surv. Tutorials.*, vol. 18, no. 2, pp. 1153–1176, Apr. 2016. doi: 10.1109/COMST.2015.2494502.

[29] A. Alsaedi, N. Moustafa, Z. Tari, A. Mahmood, and A. N. Anwar, "TON-IoT telemetry dataset: A new generation dataset of IoT and IIoT for data-driven intrusion detection systems," *IEEE Access*, vol. 8, pp. 165130–165150, 2020. doi: 10.1109/ACCESS.2020.3022862.

[30] J. Han, M. Kamber, and J. Pei, "Classification: Advanced methods," in *Data Mining: Concepts and Techniques*, 3rd ed., San Mateo, CA, USA: Morgan Kaufmann, 2012, pp. 393–442.

[31] F. Itoo, Meenakshi, and S. Singh, "Comparison and analysis of logistic regression, Naïve Bayes and KNN machine learning algorithms for credit card fraud detection," *Int. J. Inf. Technol.*, vol. 13, no. 4, pp. 1503–1511, Aug. 2021. doi: 10.1007/S41870-020-00430-Y/METRICS.

[32] D. Berrar, "Bayes' Theorem and Naive Bayes classifier," *Encycl. Bioinforma. Comput. Biol. ABC Bioinforma.*, vol. 403, 2018, Art. no. 412.

[33] S. Abirami and P. Chitra, "Energy-efficient edge based real-time healthcare support system," *Adv. Comput.*, vol. 117, no. 1, pp. 339–368, Jan. 2020. doi: 10.1016/bs.adcom.2019.09.007.

[34] T. Thomas, A. P. Vijayaraghavan, and S. Emmanuel, "Introduction to machine learning," in *Machine Learning Approaches in Cyber Security Analytics*, 1st ed., Cham, Switzerland: Springer, 2020, pp. 17–36.