Research Article

Academic Platform Journal of Engineering and Smart Systems (APJESS) 12(1), 28-36, 2024

Received: 17-Jan-2023 Accepted: 27-Jan-2024

homepage: https://dergipark.org.tr/tr/pub/apjess

https://doi.org/10.21541/apjess.1236912

IoT-based Smart Home Security System with Machine Learning Models

*[1]Selman HIZAL, [2]Ünal ÇAVUŞOĞLU, [3]Devrim AKGÜN

[1] Department of Computer Engineering, Sakarya University of Applied Sciences, Sakarya, Türkiye, selmanhizal@subu.edu.tr

[2] Department of Software Engineering, Sakarya University, Sakarya, Türkiye, unalc@sakarya.edu.tr

[3] Department of Software Engineering, Sakarya University, Sakarya, Türkiye, dakgun@sakarya.edu.tr

Abstract

The Internet of Things (IoT) has various applications in practice, such as smart homes and buildings, traffic management, industrial management, and smart farming. On the other hand, security issues are raised by the growing use of IoT applications. Researchers develop machine learning models that focus on better classification accuracy and decreasing model response time to solve this security problem. In this study, we made a comparative evaluation of machine learning algorithms for intrusion detection systems on IoT networks using the DS2oS dataset. The dataset was first processed for feature extraction using the info-gain feature selection approach. The original dataset (12 attributes), the dataset (6 attributes) produced using the info gain approach, and the dataset (11 attributes) obtained by eliminating the timestamp attribute were then formed. These datasets were subjected to performance testing using several machine learning methods and test choices (10-crossfold, percentage split). The test performance results are presented, and an evaluation is performed, such as accuracy, precision, recall, and F1 score. According to the test results, it has been observed that 99.42% accuracy detection rates are achieved with Random Forest for IoT devices with limited processing power.

Keywords: information security; IoT; intrusion detection system; machine learning

1. INTRODUCTION

Due to the rapid development of information and communication in all sectors, numerous sensors, hardware components, and software programs exist. Today, IoT is widely used in many fields, such as industry, military, health, energy distribution, education, entertainment, agriculture, and transportation. IoT also has many specialized application areas in supply chain management, smart homes, smart cities, connected cars, and so on. With the decrease in the cost of IoT devices and the increase in their usage, they are also actively performed, especially in smart home systems. These systems make our homes smart and can be controlled with mobile applications. In addition to offering many conveniences to people, it also reveals some personal security concerns. Malicious attacks on IoT communication infrastructure have been increasing daily and bringing severe security problems in recent years. Especially since IoT devices need less computational capacity and energy consumption, security systems developed for IoT must comply with these requirements. But cybercriminals are increasingly focusing on these systems. For this reason, there is a need to develop security systems specific to these networks that will ensure the security of IoT networks. Intrusion Detection Systems (IDS) have been developed in this area with many different methods.

Machine learning (ML) algorithms are widely used in security systems designed to secure IoT networks. Many studies in the literature use machine learning methods to achieve IoT system security. Some studies presented the recently developed methods and architectures to ensure IoT security. Hasan et al. [1] suggested an IDS using different ML algorithms in IoT sensor networks. Many methods, such as Random Forest, Artificial Neural Network, Support Vector Machine, Decision Tree, and Logistic Regression, are used to develop the system. Latif et al. [2] introduced an IDS for IoT-based industrial networks. It is possible to identify several threats to industrial networks, including denial of service (DoS), espionage data probing, scan, and malicious operation and control. A novel lightweight random neural network-based prediction model for IDS is suggested and compared to previous research. Kumar et al. [3] presented a new IDS based on a distributed ensemble design using fog computing for IoT networks. A double-layer structure is recommended in the proposed system, with K-Nearest Neighbors (KNN), eXtreme Gradient Boosting (XGBoost), and Naive Bayes used in the first layer and Random Forest techniques chosen in the second layer. Training and testing

processes were carried out in the UNSW cyber security lab in 2015 (UNSW-NB15), and the distributed smart space orchestration system (DS2oS) data sets and performance test results are presented. Reddy et al. [4] suggested an IDS to use in smart city applications. In the article, attacks were classified, and performance tests were carried out on the DS2oS data set. It has been reported that the proposed deep learning-based system provides a serious improvement for most attack types. Cheng et al. [5] proposed an IDS for IoT systems using a kind of convolutional neural network. For the training of the proposed system, two separate data sets were derived from the DS2oS data set, and optimal parameters were determined for labeled and unlabeled data. The proposed model is compared with many different methods, computation complexity analyses, and performance results are presented. It has been stated that it provides a serious improvement, especially on unlabeled data. Rashid et al. [6] developed a deep learning-based adversarial IDS for their IoT smart city applications. DS2oS data set is used, and different attack models are tested. The proposed model has been shown to achieve successful results in both binary and multi-class classification. Weinger et al. [7] worked with the publicly available Telemetry datasets of IoT (TON_IoT) and DS2oS datasets. They tested five different data augmentation methods on these datasets and showed that class imbalances have a negative impact on the detection rate. Chen et al. [8] have shown that their proposed DAGAN architecture can produce better results by preventing a marginal sample from being mispriced in industrial control systems. They have demonstrated this advantage in their experimental studies on DS2oS and Secure Water Treatment (SWaT) datasets. Mukherjee et al. [9] proposed an ML-based system for detecting attacks on the IoT device, which is now also referred to as smart. They tried classification models for two different cases on the DS2oS dataset. In the first case, Naïve Bayes had the lowest success rate, while in the second case, they achieved the highest prediction rates using Decision Tree and Random Forest. Amroui and Zouari [10] have proposed an architecture called Duenna to detect user behaviors that exhibit different behaviors, taking into account the use of devices within smart home systems by regular users. In this way, they have helped increase security against malicious individuals who threaten smart-home users and want to hijack the systems. Lysenko et al. [11] developed an ML-based IDS by analyzing the information in the network infrastructure packets that IoT devices use to communicate. They tested their flow-based models with the low computational cost for IoT devices using five different ML classification algorithms. For their study, they used traffic data from six different datasets. It was found that Random Forest (RF) performed the best, while Support Vector Machine (SVM) performed the worst. Hassan et al. [12] proposed a real-time method for detecting and mitigating Distributed Denial-of-Service (DDoS) attacks using the DS2oS and UNSW-NB15 datasets. They utilized fog computing and a machine learning approach based on KNN. Mendonça et al. [13] focused on a lightweight implementation of the IDS system using a model based on a sparse connected multi-layer perceptron structure. They gave the training and test time performance results to show the sparse model in addition to attack detection evaluations. Wahab [14] developed a deep learning model that dynamically determines the depths of hidden layers and considers concept drift and data drift conditions in an IoT environment. Le et al. [15] proposed a model based on ensemble tree models, decision trees, and random forests. They used an online fine-tuning method for their deep learning model and drift detection methods. Also, they used the Shapley Additive Explanations (SHAP) to interpret the decision of the ensemble tree approach. Shobana et al. [16] proposed a new method for IoT smart city applications using a privacy-preserving model based on blockchain. They employed an optimization algorithm to optimize the hyperparameters of the hybrid deep neural network for IDS.

According to recently reviewed studies, many strategies were employed in IDS designs for IoT systems. Different IoT datasets were preferred for the training and testing of the developed systems. In this work, we focused on using the DS2oS data set. Common evaluation criteria were used in the examination of system performance. The results show that performance is strictly related to the data set and ML method. In addition, it is understood that the preprocessing operations on the data set also have an effect on the performance. It has been determined that high performances that do not reflect the truth are obtained in the performance results where the datasets containing repetitive recordings are used without preprocessing the datasets. In general, it can be concluded that tree-based system designs have higher performance.

The contribution of this article is as follows.

- This study evaluated the efficiency of different machine-learning algorithms for IoT networks in terms of IDS using the DS2oS dataset.
- The IDS's performance was evaluated with the reduced number of features using feature selection procedures.
- The IDS performance of different machine-learning algorithms in IoT networks in this work.
- We determined methods that provide high performance and low energy consumption by obtaining datasets with fewer features.

We organized the article as follows: First, we presented an evaluation by a literature review. Then, we explained the general structure of IDS in IoT home security, DS2oS dataset features, feature selection technique, and performance metrics. In the third part, we carried out performance tests. In the last part, we made evaluations and suggestions for future work.

2. BACKGROUND

In this section, we first describe IDS for IoT networks. Secondly, we introduce the DS2oS dataset, which is widely used in IoT security analysis, and its characteristics. Then, we presented the ML models that are used for detecting IDS in IoT. Also, we covered the methods for extracting features to be used on the dataset. Finally, we give the performance metrics to evaluate the model.

2.1. Intrusion Detection Systems for IoT

The variety of IoT devices and their richness of services make them indispensable parts of our daily lives. These devices can communicate with each other over a certain protocol, facilitating the necessary work and even making some decisions for us. With the rapid growth of IoT networks in the coming years, attacks will likely diversify. At the initial stage, cryptographic security mechanisms like authentication and encryption are insufficient due to the resource constraints of the devices, and there are several security vulnerabilities against attacks. Therefore, it is necessary to provide more advanced security by effectively detecting infiltration against attacks. Moreover, it is important to develop systems with less computational overhead for IoT devices compared to traditional IDS, as IoT devices are generally lightweight in terms of resources. Figure 1 shows IoT devices in a smart home system including a variety of ordinary appliances and devices that have sensors, connections, and the potential to communicate with other devices or the internet. These electronic devices have the purpose of making the home more intelligent, safer, and more valuable.

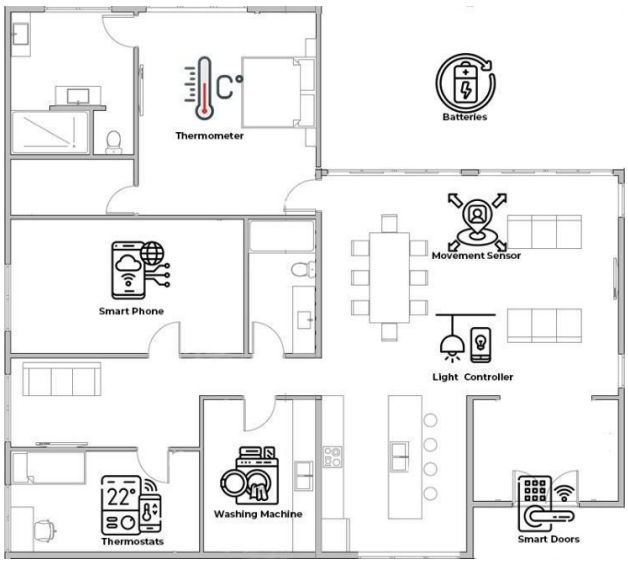

Figure 1. A Basic Sample of IoT Smart Home Plan

Today, many traditional security measures automatically detect threats with Artificial Intelligence and ML tools. Additionally, systems that can make necessary decisions to prevent attacks are available. For example, ML anomaly detection can be performed by automatically detecting upcoming threats. ML algorithms constantly work on IoT device traffic data to detect abnormal behavior. The notifications are generated when an anomaly is detected, and apps can be programmed to react automatically to specific irregularities.

2.2. Overview of DS2oS Dataset

The DS2oS dataset, was developed in 2008 by Oliver Pahl and extended in 2018 by François-Xavier Aubert with the module for anomaly detection as part of his bachelor thesis in computer science [17] [18]. DS2oS was created to ensure the privacy and security of IoT users. The DS2oS dataset is accessible to the public through Kaggle. During the dataset development process, the system is trained to detect abnormal activity while taking into account normal user behavior. Sensors in a home and the actions of IoT devices at the application layer were utilized to create the dataset. Using a knowledge agent and virtual state layer (VSL) in the dataset architecture, the acquired data from IoT services may be shared with other IoT devices. Using a web interface or mobile application, users can give instructions to all IoT devices through a central administration system. Their actions are automatically logged in the data repository. A developer can also access the recorded data and publish new services.

/kaName/serviceName/variableName is the specific address used to access each node in the system. These nodes' types, including SmartDoors, Batteries, LightController, etc., as well as their locations, including entrance, kitchen, and bathroom, are also known. Four properties—serviceID (service1), accessednodeaddress (kaName/service1), operation (read, write), and timestamp (847690962, 1513093731) are used to define each connection.

The DS2oS dataset was produced using light controls, motion sensors, thermostats, washing machines, solar batteries, door locks, and smartphones, as seen in Figure 1. Four different IoT places (the house, two-room apartment, three-room apartment, and office) were tracked for a whole day while this dataset was being created, and traffic was recorded. Within each place, there are variations in the structure and procedures. It is discovered that there are the most DDoS attacks (5,780) and the fewest wrongSetUp (122) when only 3% of the created dataset is analyzed, as shown in Figure 2.

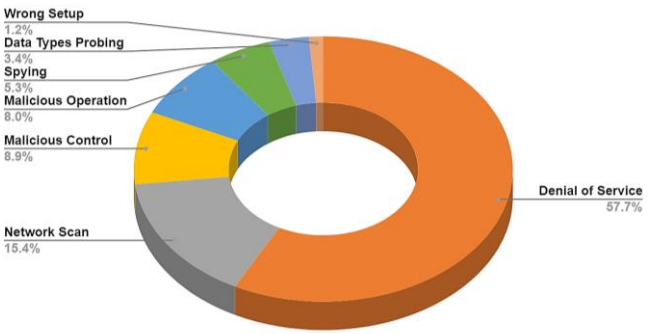

Figure 2. Anomalous Attack Types in the DS2oS Dataset

Table 1. The number of samples for source types

No	Number of Sources	Type	Number of Samples
1	22	lightControler	135,775
2	21	movementSensor	1,301
3	20	sensorService	85,196
4	6	batteryService	81,273
5	5	doorLockService	335
6	4	thermostat	5,980
7	3	washingService	47,986
8	3	smartPhone	106
Total	84		

The DS2oS dataset was generated from various devices. Table 1 shows the number of sources, types, and the number of samples.

The number of samples in the dataset is a total 357,952. The number of the normal trafficking dataset is 347,935 while 10,017 samples represent abnormal behavior. The class label and the number of the sample in the DS2oS dataset are given in Table 2.

Table 2. The number of samples for normal and anomaly attack types.

No	Label	Count
1	Normal	347,935
2	Scan	1,547
3	Malicious Operation	805
4	DoS	5,780
5	Spying	532
6	Data Probing	342
7	Wrong Setup	122
8	Malicious Control	889
	Total	357,952

Normal: Regular user activity behavior.

Denial of Service (DoS): The attacker transmits large amounts of traffic in order to disrupt services on IoT devices.

Network Scan: This attack first scans IoT devices to learn more about their networks before launching complex attacks to compromise security. IP address, port, and version scanning are often used scanning methods to gather device network information.

Malicious Control: This attack allows unauthorized access to IoT devices, allowing vital information to be accessed. These attacks are frequent cyberattacks in which the victim's system is compromised by malware, typically malicious software. Ransomware, malware, command and control, and other specialized attacks are all included in malicious software, sometimes known as viruses. For instance, hackers utilize the factory default login details of unprotected devices to infect thousands of IoT devices.

Malicious Operation: This type of attack occurs when IoT devices undertake operations that are not anticipated by them.

Spying: This is a sort of attack in which a hostile IoT device gains access to the sensitive information of others by exploiting system weaknesses.

Data Types Probing: This is a form of attack in which the attacker looks for weaknesses or vulnerabilities in an IoT device.

Wrong Setup: In this attack, a hacker might gain access to sensitive and important information about clients or the sector by taking advantage of an incorrect system setup.

2.3. Machine Learning Models

The ML models for classifying the DS2oS dataset is generally explained in this section. We used NaiveBayes, Random Tree, J48, and SVM. Here are the detailed working principles of the algorithms as follows.

Naive Bayes: As a probabilistic classifier, Naive Bayes applies Bayes' theorem for the decision rule and uses training data to determine the proper parameters to classify data. There are various algorithms to train classifiers, but the common point is that the features in the training set are assumed to be independent. Therefore, the form of the covariance matrices is diagonal. Table 3 shows the details of the features in the DS2oS dataset.

Table 3. The data type of the features in the DS2oS dataset.

No	Features	Type
1	sourceID	nominal
2	sourceAddress	nominal
3	sourceType	nominal
4	sourceLocation	nominal
5	destinationServiceAddress	nominal
6	destinationServiceType	nominal
7	destinationLocation	nominal
8	accessedNodeAddress	nominal
9	accessedNodeType	nominal
10	operation	nominal
11	value	continuous
12	timestamp	discrete
13	normality	nominal

C4.5 (J48) Algorithm: One of the decision tree algorithms is the C4.5 algorithm which is represented with J48 in Weka and is derived from the ID3 (Iterative Dichotomiser 3) algorithm. C4.5 uses an if-then set of rules converted from trained trees. It selects splitting attributes based on the information gain ratio. The algorithm's performance remains the same regardless of the amount of data to be trained.

Random Forest: Random Forest is a prominent ensemble learning algorithm employed in machine learning for enhancing predictive accuracy and reducing overfitting. This algorithm builds a collection of decision trees, each constructed from a different subset of the training data and employing random feature selection, and combines their outputs to make a final prediction. The aggregation process is mathematically represented as follows:

$$H(x) = argmax\left(1/T \sum_{t=1}^{T} \Pi\left(c_t(x)\right)\right) \quad (1)$$

where H(x) is the ensemble's final prediction, T represents the number of decision trees, $\left(\Pi\left(c_t(x)\right)\right)$ indicates the prediction of the t-th tree for input x, and the majority vote is used for classification tasks. By introducing randomness during the tree construction process, Random Forest mitigates overfitting and provides improved generalization,

making it a widely used tool in various ML applications. This algorithm's robustness and effectiveness have solidified its place as a fundamental component of ensemble methods in the field of data science and pattern recognition.

Bagging: Bagging, short for Bootstrap Aggregating, is a popular ensemble learning method widely used in machine learning and data mining. The primary objective of Bagging is to enhance the predictive performance and reduce the variance of base classifiers by generating multiple bootstrap samples from the training dataset and training a set of base classifiers on these samples. The final prediction is typically achieved through a majority vote (for classification) or averaging (for regression) of the individual base classifiers. The aggregation process is mathematically represented as

$$H(x) = argmax\left(\sum_{i=1}^{T} w_i \cdot \Pi\big(c_i(x)\big)\right) \quad (2)$$

where H(x) is the final ensemble prediction, $\Pi\big(c_i(x)\big)$ is an indicator function evaluating the prediction of the i-th base classifier for input x, and w, i represents the weight assigned to each base classifier. The Bagging algorithm provides a powerful framework for improving the robustness and generalization of machine learning models, effectively reducing over fitting and enhancing classification accuracy. This method has been successfully applied in a variety of domains, making it a cornerstone of ensemble learning techniques in the field of data science and pattern recognition.

K-Star: The KStar algorithm is a well-established instance-based machine learning approach employed for feature selection in the field of data mining. It is particularly useful for classification tasks and is based on the k-nearest neighbor (k-NN) principle. The central idea behind KStar is to assess the relevance of each feature in a dataset by comparing the class distribution for the k-nearest neighbors of each instance with the class distribution for the entire dataset. The algorithm assigns a weight to each feature based on this comparison, enabling the selection of the most informative features. Mathematically, the weight (w_i) assigned to each feature (F_i) is computed as

$$w_i = 1/k\left(\sum_{j=1}^{k} N_{ij}/N_j\right) \quad (3)$$

where N_{ij} represents the number of instances in the k-nearest neighbors of instance i belonging to class j, and N_j is the total number of instances belonging to class j in the dataset. The features with higher weights are considered more relevant for classification. KStar offers a computationally efficient approach to feature selection and is commonly employed for improving the efficiency and accuracy of classification models in various research and practical applications.

2.4. Feature Selection

Feature selection was performed on the DS2oS dataset. As a result of this process, the number of features was reduced from 12 in the original data set to 6. Applying the feature extraction process is aimed at obtaining equivalent or higher performance values with a lower number of features. The effect of feature selection on the performance was examined with the tests made with the data set with a reduced number of features. There are many different techniques in the literature as feature extraction methods. This study explains the Info Gain (IG) attribute selection method widely used in the literature.

Algorithm-independent relevant features are found using sorting in filtering-based feature extraction techniques. The algorithms have a lower computing load and provide results faster. The IG technique [19] minimizes dataset size and provides a small dataset with efficient and superior performance outcomes. Using this algorithm, 6 features were selected according to the IG method shown in Table 4. The IG method assigns ranks for each feature according to the importance determined by the algorithm. Feature reduction improves the algorithm's speed and ensures a straightforward tree search.

Table 4. The number of samples for normal and anomaly attack types

Ranked	No	Attribute
0.1543	2	sourceAddress
0.1521	8	accessedNodeAddress
0.1511	5	destinationServiceAddress
0.1478	1	sourceID
0.0902	11	value
0.0882	3	sourceType

Performance tests were made to evaluate the efficiency of the IoT-based traffic of the DS2oS dataset using various algorithms. Performance tests were conducted on the WEKA program using a variety of measures that are often used in the literature, such as Accuracy, Recall, Precision, and F-Measure [20], [21], [22]. The following provides an explanation of the values calculated in the complexity matrix, which is shown in Table 5:

TP (True-Positive): The amount of data in the dataset that is in the normal class and predicted in the normal class.

FN (False-Negative): The amount of data in the dataset in the normal class and predicted as an attack.

FP (False-Positive): The amount of data in the dataset that is in the attack class and normally estimated.

TN (True-Negative): The amount of data in the dataset in the attack class and estimated as an attack.

Accuracy: The percentage of correct predictions made by our model out of all the estimates is known as accuracy. This metric is calculated as the ratio of the number of samples properly identified by a data mining algorithm to the entire sample. The number of samples allocated from the data set for testing is utilized to calculate this value. The following formula is used to compute the value.

$$Accuracy = (TP + TN)/(TP + FN + FP + TN) \quad (4)$$

Recall: Recall measures the percentage of true positives that were accurately detected. The ratio of the number of items in the normal class and predicted as normal in the data set to all samples that are normal gives the sensitivity value. The value is calculated as follows.

$$Recall = TP/(TP + FN) \quad (5)$$

Precision: Precision indicates what percentage of positive predictions were correct. It is the ratio of the number of values classified as normal in the data set to the number of all samples predicted as normal. The value is calculated as follows.

$$Precision = TP/(TP + FP) \quad (6)$$

F-Measure: The F1 score, which represents the harmonic average of Precision and Recall, is a lesser-known performance metric.

This evaluation criterion generates a new value by combining precision and sensitivity. This value is calculated using the harmonic mean of the precision and sensitivity values obtained. The following formula is used to compute the value.

$$FMeasure = 2x\left(\frac{Recall\ x\ Precision}{Recall + Precision}\right) \quad (7)$$

Table 5. Confusion Matrix for Performance Calculation.

Class/Attack Type		Predicted Class	
		Normal	Attack
True Class	Normal	TP	FN
	Attack	FP	TN

3. EXPERIMENTAL EVALUATIONS

This section explains how the other datasets were derived from the DS2oS dataset. Then we carried out performance testing processes with different test options and ML algorithms. We made various performance comparisons to evaluate the model and the selected features.

3.1. Experimental Setting

We used the Weka tool for training ML models and performance evaluations. Table 6 summarizes the hardware and software used to evaluate the models. Performance measurements were done on the Windows operating system, which runs on hardware with a CPU model, i5-11400H @ 2.70GHz processor with 16 GB memory. The GPU model is NVIDIA GeForce® GTX1650 with 4 GB memory.

Table 6. Experimental hardware and software environment.

Hardware / Software	Features
Operating System	Windows 10, 64-bit
Weka	3.8.6
CPU	i5-11400H @ 2.70GHz
RAM	16 GB
Video Graphics Card	NVIDIA GeForce® GTX1650

We derived two different datasets from the original dataset. The first dataset contains 11 attributes by removing the timestamp attribute from the original dataset. The other dataset contains six features produced using the Infogain evaluation method. Removed feature numbers are 4, 6, 7, 9, 10, 12. We used these two datasets and the original dataset for training the ML models. Duplicate checks on the first derived dataset revealed many repetitive data depending on the Timestamp attribute in the DS2oS dataset. Therefore, training ML models on the original dataset where the only differentiating feature is the timestamp for many features produces unrealistically high-performance results.

Figure 3 shows the options used for the training and testing. Our training and testing datasets contain three alternatives based on the selected features. We used the Weka tool to train and test Naive Bayes and J48 ML algorithms. There are various options, such as test split and k-fold cross-validation. In the Percentage split test option, 80% of the DS2oS dataset was used for training, and testing was carried out on the remaining 20%. In the K-fold method, the k value is selected as 10, the dataset is divided into ten different parts, and tests are performed on another part in each iteration. As a machine learning method, operations were carried out on all datasets and test options with NaiveBayes and J48 classifier algorithms, which are widely used in the literature.

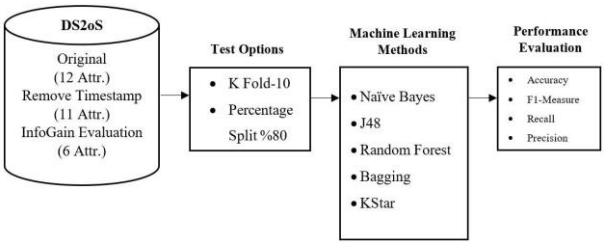

Figure 3. System Architecture and Performance Evaluation

3.2. Performance Comparisons

The results of all tests performed are presented in Table 7. When the results with the original dataset with 12 features are examined, there is no significant difference between the k-fold and percentage split test methods. Algorithms obtained with the J48 algorithm have higher performance than the NaiveBayes algorithm in both options. Since it is used as a decision attribute over the timestamp attribute in the J48 algorithm, the results have been obtained as weighted average performance values of 1.00. This shows that the system detects at a high rate by memorizing. Therefore, the timestamp property was deleted from the dataset, and a new 11-qualified dataset was obtained. In the tests made with this dataset, better results were obtained using the k-fold test option and the J48 algorithm.

The parameter values of the machine learning algorithms used in training and testing are given below. For Naive Bayes and K-Star algorithms, the batch size value was used as 100. For the J-48 algorithm, the confidence factor, batch size, number folds, and seed values were set as 0.25,100,3,1 respectively. In the random forest algorithm, the parameter values are batch size 100, max depth unlimited, num_iterations 100 and seed 1. In the bagging method, the REPTree algorithm was preferred as the classifier and the

parameter values of bag size percent, batch size, num_iterations and seed were set as 100, 100, 10 and 1 respectively.

Table 7. The Performance Evaluation Results on DS2oS Dataset.

Dataset	Method	Acc.	Prec.	Recall	F1	AUC
percentage 80% 12-features	NB	96.93	0.99	0.97	0.98	1.00
	J48	100.00	1.00	1.00	1.00	1.00
	RF	100.00	1.00	1.00	1.00	1.00
	Bagging	99.88	1.00	1.00	1.00	1.00
	KStar	99.48	0.99	1.00	1.00	0.95
k-fold 10 12-features	NB	97.07	0.99	0.97	0.98	1.00
	J48	100.00	1.00	1.00	1.00	1.00
	RF	100.00	1.00	1.00	1.00	1.00
	Bagging	99.89	1.00	1.00	1.00	1.00
	KStar	99.54	1.00	1.00	1.00	0.96
percentage 80% 11-features	NB	96.98	0.99	0.97	0.98	0.99
	J48	99.39	0.99	0.99	0.99	1.00
	RF	99.39	0.99	0.99	0.99	1.00
	Bagging	99.29	0.99	0.99	0.99	1.00
	KStar	99.32	0.99	0.99	0.99	1.00
k-fold 10 11-features	NB	97.12	0.99	0.97	0.98	0.99
	J48	99.42	0.99	0.99	0.99	1.00
	RF	**99.42**	0.99	0.99	0.99	1.00
	Bagging	99.33	0.99	0.99	0.99	1.00
	KStar	99.35	0.99	0.99	0.99	1.00
percentage 80% 6-features	NB	96.16	0.99	0.96	0.97	0.99
	J48	98.71	0.99	0.99	0.98	0.99
	RF	99.28	0.99	0.99	0.99	1.00
	Bagging	99.19	0.99	0.99	0.99	1.00
	KStar	99.27	0.99	0.99	0.99	1.00
k-fold 10 6-features	NB	96.32	0.99	0.96	0.97	0.99
	J48	98.70	0.99	0.99	0.98	0.99
	RF	**99.30**	0.99	0.99	0.99	1.00
	Bagging	99.20	0.99	0.99	0.99	1.00
	KStar	99.29	0.99	0.99	0.99	1.00

The 6-selected features percentage split and k-fold validation results created by applying the IG method were close to each other for J48 and NaiveBayes. The accuracy value obtained with 11 attributes decreased from 99.42% to 98.69%. In Figure 4 and Figure 5, confusion matrices were obtained using the k-fold test option, and the J48 algorithm for datasets with 12 and 6 attributes is seen in the confusion matrix. As shown in Figure 4, the J48 algorithm only made an error for normal classification. Figure 5 shows that the J48 algorithm misclassified especially DoS, malicious operation, and data probe attacks for the dataset with six features.

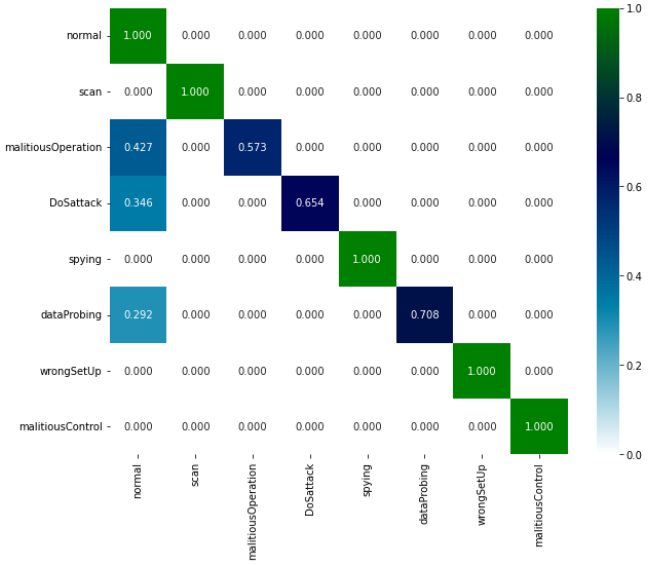

Figure 4. Confusion Matrix for Random Forest algorithm using selected 12 features

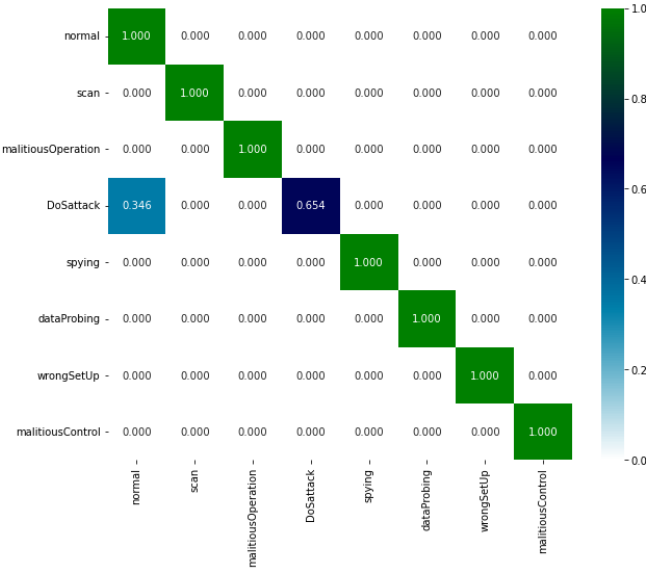

Figure 5. Confusion Matrix for Random Forest algorithm using selected 6 features

4. DISCUSSION

Comparisons of recent studies in the literature with the DS2oS dataset are presented in Table 8. When the results of the studies are analyzed, it is seen that high achievements around 99% are generally obtained. In the comparison with the proposed model, results with similar accuracy rates were obtained. In the performance tests performed without removing the timestamp attribute from the dataset, results close to 100% were obtained as shown in Table 6. Since timestamp is a unique value, it was observed that the tests performed without removing timestamp from the dataset resulted in unrealistic results. In addition, a dataset with 6 features was created with the info gain feature extraction method and a 99.30% success rate was obtained with the RF method. It is evaluated that achieving this performance over a 6-attribute data set can be used especially in IoT networks with low computational capacity.

Table 8. Comparison with recent studies on the DS2oS Dataset.

Authors	Model Tested	Best Model	Accuracy (%)	Precision (%)	Classification
Hasan et al. 2019 [1]	LR, SVM, DT, RF, ANN	RF	99.40	98.00	Multiclass
Latif et al. 2020 [2]	SVM, DT, ANN, RaNN	RaNN	99.20	99.08	Multiclass
Cheng et al. 2020 [5]	TCN, LSTM, SVM	HS-TCN	98.22	97.67	Multiclass
Reddy et al. 2020 [23]	Bayes Net, DT, NB, RF, DNN	DNN	98.28	97.00	Multiclass
Yadav et al. 2022 [24]	LR, RF, DT, ANN, KNN, AdaBoost	Adaboost	99.56	NA	Multiclass
Kushwah Et al. 2023 [25]	SVM, DT, LR, RF, ANN AdaBoost	CatBoost	99.45	98.73	Multiclass
Paul Et al. 2023 [26]	Ensemble-DNN, RNN, CVT, DBN, TANN, F-SVM, DMM, DNN	Hybrid ML Model	99.80	99.50	Multiclass
Our Model 2023	NB, J48, RF, Bagging, K-Star	Random Forest	99.42	99.0	Multiclass

The scatter plot, a powerful tool in data exploration and analysis, enabled us to investigate the distribution and correlation of variables. Figure 6 shows a scatter plot which is a crucial aspect of our research to display the relationships and patterns in the DS2oS dataset. Distributions were obtained for normal and other attack types. When the distribution values in the dataset are examined, it is seen that the number of normal traffic samples is higher than the attack types and is distributed homogeneously. In other attack types, it was determined that the distributions were concentrated in certain regions due to the low number of samples.

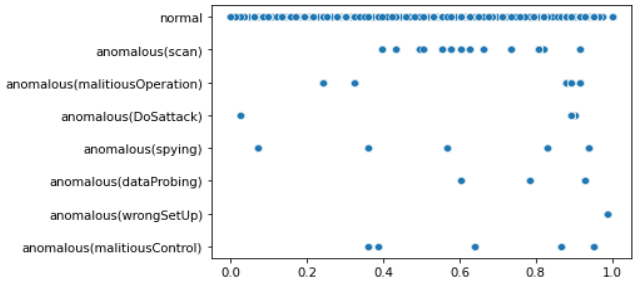

Figure 6. Distribution of attack types in the DS2oS

5. CONCLUSIONS

We performed a comparative assessment of machine learning techniques for intrusion detection systems on IoT networks using the DS2oS dataset. In the paper, DS2oS data set is introduced, the main principles of IDS are discussed, and details on the data set's attack types are provided. The feature selection approach, performance metrics utilized for evaluation, and comparison of ML algorithms are introduced. The dataset was processed using info-gain feature selection, resulting in 12 attributes, 6 attributes, and 11 attributes after eliminating the timestamp attribute. The datasets have been evaluated for performance using various machine learning algorithms and test configurations. Random Forest demonstrated its efficacy by achieving 99.42% accuracy detection rates for IoT devices with limited processing resources. The results show that using the timestamp value as a determining feature produced unhealthy results in terms of performance, so it was not used in the processed data sets. The test findings on the 6 attribute data sets acquired by the IG feature selection approaches have been proven effective, and good performance is achieved with fewer features. When we compare the obtained results, it is observed that the RF algorithm produces higher performance for all datasets. For future studies, the number of samples for specific classes, such as Wrong setup or Data Probing, is considerably low compared to the other classes. Advanced techniques such as GAN-based re-samplers can be trained to increase the number of samples.

5.1. Data Availability:

The DS2oS dataset is accessible at: https://www.kaggle.com/datasets/francoisxa/DS2oStraffictraces

Author contributions: All authors contributed equally to the creation of the idea and the design.
Conflict of Interest: No conflict of interest was declared by the authors.
Financial Disclosure: This study is supported by Sakarya University of Applied Sciences, Scientific Research Projects under the grant number 105-2022.

REFERENCES

[1] M. Hasan, M. M. Islam, M. I. I. Zarif, and M. Hashem, "Attack and anomaly detection in iot sensors in iot sites using machine learning approaches," Internet of Things, vol. 7, p. 100059, 2019.

[2] S. Latif, Z. Zou, Z. Idrees, and J. Ahmad, "A novel attack detection scheme for the industrial internet of things using a lightweight random neural network," IEEE Access, vol. 8, pp. 89 337–89 350, 2020.

[3] P. Kumar, G. P. Gupta, and R. Tripathi, "A distributed ensemble design based intrusion detection system using fog computing to protect the internet of things networks," Journal of Ambient Intelligence and Humanized Computing, vol. 12, no. 10, pp. 9555–9572, 2021.

[4] D. K. Reddy, H. S. Behera, J. Nayak, P. Vijayakumar, B. Naik,and P. K. Singh, "Deep neural network based anomaly detection in internet of things network traffic tracking for the applications of future smart cities," Transactions on Emerging Telecommunications Technologies, vol. 32, no. 7, p. e4121, 2021.

[5] Y. Cheng, Y. Xu, H. Zhong, and Y. Liu, "Leveraging semisupervised hierarchical stacking temporal convolutional network for anomaly detection in iot communication," IEEE Internet of Things Journal, vol. 8, no. 1, pp. 144–155, 2021.

[6] M. M. Rashid, J. Kamruzzaman, M. M. Hassan, T. Imam, S. Wibowo, S. Gordon, and G. Fortino, "Adversarial training for deep learning-based cyberattack detection in iot-based smart city applications," Computers & Security, p. 102783, 2022.

[7] B. Weinger, J. Kim, A. Sim, M. Nakashima, N. Moustafa, and K. J. Wu, "Enhancing iot anomaly detection performance for federated learning," Digital Communications and Networks, 2022.

[8] L. Chen, Y. Li, X. Deng, Z. Liu, M. Lv, and H. Zhang, "Dual auto-encoder gan-based anomaly detection for industrial control system," Applied Sciences, vol. 12, no. 10, p. 4986, 2022.

[9] I. Mukherjee, N. K. Sahu, and S. K. Sahana, "Simulation and modeling for anomaly detection in iot network using machine learning," International Journal of Wireless Information Networks, pp. 1–17, 2023.

[10] N. Amraoui and B. Zouari, "Anomalous behavior detection based approach for authenticating smart home system users," International Journal of Information Security, vol. 21, no. 3, pp. 611–636, 2022.

[11] S. Lysenko, K. Bobrovnikova, V. Kharchenko, and O. Savenko, "Iot multi-vector cyberattack detection based on machine learning algorithms: Traffic features analysis, experiments, and efficiency," Algorithms, vol. 15, no. 7, p. 239, 2022.

[12] K. F. Hassan and M. E. Manaa, "Detection and mitigation of ddos attacks in internet of things using a fog computing hybrid approach," Bulletin of Electrical Engineering and Informatics, vol. 11, no. 3, 2022.

[13] R. V. Mendonc¸a, J. C. Silva, R. L. Rosa, M. Saadi, D. Z. Rodriguez, and A. Farouk, "A lightweight intelligent intrusion detection system for industrial internet of things using deep learning algorithms," Expert Systems, vol. 39, no. 5, p. e12917, 2022.

[14] O. A. Wahab, "Intrusion detection in the iot under data and concept drifts: Online deep learning approach," IEEE Internet of Things Journal, 2022.

[15] T.-T.-H. Le, H. Kim, H. Kang, and H. Kim, "Classification and explanation for intrusion detection system based on ensemble trees and shap method," Sensors, vol. 22, no. 3, p. 1154, 2022.

[16] M. Shobana, C. Shanmuganathan, N. P. Challa, and S. Ramya, "An optimized hybrid deep neural network architecture for intrusion detection in real-time iot networks," Transactions on Emerging Telecommunications Technologies, p. e4609, 2022.

[17] M.-O. Pahl and F.-X. Aubet, "All eyes on you: Distributed multi-dimensional iot microservice anomaly detection," in 2018 14th International Conference on Network and Service Management (CNSM). IEEE, 2018, pp. 72–80.

[18] F. Aubet and M. Pahl, "Ds2os traffic traces," 2018. [Online]. Available: https://www.kaggle.com/datasets/francoisxa/ds2ostraf ostraffic

[19] S. Jadhav, H. He, and K. Jenkins, "Information gain directed genetic algorithm wrapper feature selection for credit rating," Applied Soft Computing, vol. 69, pp. 541–553, 2018.

[20] N. Japkowicz and M. Shah, Evaluating learning algorithms: a classification perspective. Cambridge University Press, 2011.

[21] T. R. Patil, "Mrs. ss sherekar," performance analysis of j48 and j48 classification algorithm for data classification"," International Journal of Computer Science And Applications, vol. 6, no. 2, 2013.

[22] X. Deng, Q. Liu, Y. Deng, and S. Mahadevan, "An improved method to construct basic probability assignment based on the confusion matrix for classification problem," Information Sciences, vol. 340, pp. 250–261, 2016.

[23] D. K. Reddy, H. S. Behera, J. Nayak, P. Vijayakumar, B. Naik et al., "Deep neural network based anomaly detection in internet of things network traffic tracking for the applications of future smart cities," Transactions on Emerging Telecommunications Technologies, pp. 1–26, 2020.

[24] P. K. Yadav and A. Kumar, "Analysis of Machine Learning Model for Anomaly and Attack Detection in IoT Devices," 2022 4th International Conference on Inventive Research in Computing Applications (ICIRCA), Coimbatore, India, 2022, pp. 387-392, doi: 10.1109/ICIRCA54612.2022.9985703.

[25] R. Kushwah and R. Garg, "Anomaly Detection in IOT Site Using CatBoost," 2023 3rd Asian Conference on Innovation in Technology (ASIANCON), Ravet IN, India, 2023, pp. 1-6, doi: 10.1109/ASIANCON58793.2023.10269881.

[26] J. T P, P. K, A. Paul, R. R. Chandran and P. P. Menon, "A Hybrid Machine Learning Approach to Anomaly Detection in Industrial IoT," 2023 3rd International Conference on Advances in Computing, Communication, Embedded and Secure Systems (ACCESS), Kalady, Ernakulam, India, 2023, pp. 32-36, doi: 10.1109/ACCESS57397.2023.10199711.