ISSN 2303-4521
Periodicals of Engineering and Natural Sciences *Original Research*
Vol. 9, No. 3, August 2021, pp.579-591

Enhancement performance of random forest algorithm via one hot encoding for IoT IDS

Adil Yousef Hussein [1], Paolo Falcarin [2] , and Ahmed T. Sadiq[3]
[1,3] Computer Science Department, University of Technology-Iraq, Baghdad, Iraq
[2] Software Systems Engineering (SSE) Research, University of East London, UK

ABSTRACT

The random forest algorithm is one of important supervised machine learning (ML) algorithms. In the present paper, the accuracy of the results of the random forest (RF) algorithm has been improved by the use of the One Hot Encoding method. The Intrusion Detection System (IDS) can be defined as a system that can predict security vulnerabilities within network traffic and is located out of range on a network infrastructure. It does not affect the efficiency of the built-in network because it analyzes a copy of the built-in traffic flow and reports results to the administrator by giving alerts. However, since IDS is a listening system only, it cannot take automatic action to prevent an attack or security vulnerability detected from infecting the system, it provides information about the source address to start the break-in, the address of the target and the type of suspected attack. The IoTID20 dataset is used to verify the improved algorithm, where this dataset is having three targets, the proposed system is compared with the state-of-art approaches and shows superiority over them.

Keywords: IoT, IDS, IoTID20 dataset, Random Forest, one hot encoding

Corresponding Author:

Adil Yousef Hussein
Computer Science Department, University of Technology-Iraq, Baghdad, Iraq
Baghdad-Iraq
Adil.alomran@gmail.com

1. Introduction

The Internet of Things, or IoT, can be defined as a network of several million computers that are linked together by the internet for the purpose of collecting, exchanging, and processing data. To put it another way, any smart computer connecting to a wireless network can be managed and communicated with. IoT applications can be used in a variety of settings, including homes, businesses, vehicles, and industries. The Internet of Things (IoT) is extremely elastic and has a significant economic effect on society. The IoT Market is expected to be consisting of over 84 billion connected devices that generate 186 zettabyte of data by the year of 2025[1, 2]. Having several gadgets attached to the internet is a major security issue. Not all data transmitted over the IoT system is secured, making it susceptible to malicious attacks. To build a stable and safe network, a lot of research is being carried to improve IoT security. The three fundamental rules of confidentiality, integrity, honesty, and authenticity, should be followed when developing solutions. Developing different types of security mechanisms for IoT networks, such as intrusion detection systems (IDS), has become very important, and the concept of applying machine learning to IDS is gaining a lot of interaction[3]. Intrusion detection systems are applications or services that can be used to track or diagnose unusual behavior in a network or device. A number of ML approaches are being utilized to forecast anomaly detection in IoT networks, with promising findings[4, 5]. Host-based IDSs and network-based IDSs are the two major categories of intrusion detection systems. Based on information from the computer, such as machine logs, host-based IDS is used to track and protect a single device or network. Network-based IDS is used to track an entire network by accessing and evaluating the flows that exist within it. Packet-based IDS and Flow-based IDS are 2 types of network-based IDS. Network packet information, such as

payload or header information, was used by packet-based IDS. Traditional IDS [6]is another name for them. Flow-based IDS, on the other hand, analyzes and monitors network anomalies using network flow characteristics such as data rate and byte information. Network Behavior Analysis [6] is another name for flow-based IDS. Several supervised and unsupervised machine learning models are used to classify malicious or unusual network behavior. Traditional protection methods are difficult to implement directly to protected IoT devices due to computational and fundamental resource constraints. Rule-based detection techniques, on the other hand, produced effective results[7, 8]. As a result, as IoT environments and technologies evolve, anomaly-based detection mechanisms may become increasingly important. Machine-learning algorithms can benefit greatly from big data generated by IoT devices so they can carry out the data analysis and produce meaningful predictions and interpretations of IoT devices. As a result, using machine learning to secure IoT systems is seen as the best way for protecting them from the intrusion attacks, particularly through the detection of any unusual behavior in the system. It's also worth noting that ML performs well in other fields[9, 10]. The following is the rest of the paper's structure. The related work to the proposed approach is discussed in the second section. The IoTID20 dataset is defined in the third section. In the fourth part, the proposed architecture for anomaly detection in IoT systems is presented. The proposal's results and analyses are discussed in Section 5. Finally, we wrap up this study with suggestions for future research in the final section.

2. Related work

In 2021, Qaddoura et al. [11] introduced a 3-stage solution that included clustering with the reduction, over-sampling, and classification with the use of a Single Hidden Layer Feed-Forward NN (SLFN). The paper's innovation lies in the approaches of data reduction and oversampling that have been utilized for the generation of the balanced and usable training data, as well as the hybrid consideration of unsupervised and controlled approaches for detecting activities of the intrusion. The tests were divided into four stages and tested in terms of the precision, recall, accuracy, and G-mean: measuring the impact of clustering on data reduction, the framework's performance against basic classifiers, the impact of oversampling method, and a contrast against basic classifiers On the chosen IoTID20 [12]dataset, the SLFN method of the classification and the option of Synthetic Minority Oversampling Technique (SVM-SMOTE) and Support Vector Machines (SVMs) with a ratio of 0.90 and a k value of 3 for k-means++ clustering method produce superior performance, with a score of 98.4%. Qaddoura et al.[13] in 2021 introduced a deep multilayer classification method that included the following components: an oversampling strategy utilizing SMOTE to address the problem of mismatch data-sets; and a deep multilayer method of classification that included the following components. To improve the classification outcomes, two methods are proposed. The SLFN technique's first level of classification forecasts interference and routine operations. DNN predicts the type of intrusion activity in the second stage of classification. The tests were carried out on IoTID20 [12], demonstrating that the suggested solution produced better performance. Ullah and Mahmoud in 2020 [12] used similarity of features, rating of features, and various ML approaches for classification for analyzing and comparing the IoTID20 dataset. For the normalization and interpretation of IoTID-20 data-set, they used ML algorithms and column normalization approaches. The identification capabilities of a machine learning algorithm are harmed by associated characteristics. For IoTID20, twelve associated attributes were removed from the dataset. The Shapira-Wilk algorithm was used to rate the features in the IoTID20 dataset, which tests the regularity of the feature-related distribution of occurrences. More than 70% of the features graded with a score greater than 0.50, indicating that they have a high rating. The binary, subclass, and subcategory mark datasets were all evaluated. Machine learning models are built using ensemble, Gaussian Naïve Bayes, Support Vector Machine, Logic Regression, Latent Dirichlet Allocation, RF, and Decision Tree classifiers. To evaluate the effectiveness of the different classifiers, we used numerous K-fold cross-validation measures, including 3, 5, and 10 fold cross-validation tests. The highest accuracies were achieved by Ensemble, Random Forest, and Decision Tree classifiers, while the lowest accuracies were achieved by SVM and Logic Regression. In 2020, Yang and Shami suggested the adaptive Light GBM model for the IoT data analytics [14, 15], which has high precision while using little time and memory. The proposed model will automatically respond to the ever-changing data streams of complex IoT systems owing to incorporation of their proposed novel drift-handling algorithm, Optimized Adaptive and Sliding Windowing (OASW), an ensemble ML algorithm (i.e. Light GBM), and a hyper-parameter tool, Particle Swarm Optimization (PSO). Experiments on two public IoT anomaly detection data-sets, IoTID-20 [12, 8] and NSLKDD [13], are utilized in order to test and discuss the suggested process.

On IoTID-20 and NSL-KDD datasets, the approach for effective outperforms many state-of-the-art drift adaptation methods in terms of detecting IoT attacks and adapting to idea drift, with accuracy values of 99.92% and 98.31%, respectively. Farah in 2020 [3] used two freely accessible simulation databases, IOTID20 [12] and Bot-IoT [16], for training and testing, which were developed to capture IoT networks for various attacks like the Denial of Service (DoS) and Scanning. Machine learning models that were applied to these datasets were tested within each dataset before being evaluated through datasets. There was a large variation in the analyses obtained using the two datasets. Supervised machine learning models were developed and tested for binary classification, which differentiated between standard and anomaly attack cases, as well as multiclass classification, which classified the type of attack on the IoT network. To ensure that the model works well, the flow identifiers of a network packet, like the source and destination IP addresses, port numbers, and timestamp, had to be removed. Since attackers will use various IP addresses and times to initiate attacks on the network, if the model is trained using these features, it may fail to generalize well when deployed. Furthermore, ten more functions were removed because they only had a single value and did not add much value to the machine learning models. The models were then trained using a total of 67 features. The models work admirably and are capable of detecting irregularities in the data. On both datasets, the decision tree, k-NN, and ensemble scores are both above 0.95. Assi and Sadiq in 2018 [17]proposed a feature selection strategy that has been based upon the Modified Artificial Immune System. The suggested algorithm makes use of the benefits of the Artificial Immune System to optimize the efficiency and the randomization of features. The NSL-KDD dataset [18] revealed that when compared to other feature selection algorithms, the NSL-KDD algorithm was more effective (best first search, correlation, and information gain). In 2017, Assi and Sadiq suggested five core classification methods to classify network attacks using the NSL-KDD dataset [13] and three feature selection strategies [19]. These methods include the J48 decision tree, SVM, Decision Table, Bayes Network, and Neural Network Back Propagation. Feature discovery approaches include correlation-based feature selection (CFS), information gain (IG), and decision tables. A number of the trials have been carried out to yield positive outcomes by using NSL-KDD preparation and research in the general attack (Anomaly and Normal).

Those have been performed using 4 different types of attacks: DOS, U2R, R2L, and probing. The J48 classification scheme with training data produces the highest performance (80.3 percent) by using the testing dataset and (93.9 percent) when using the consistency training dataset.

3. Random forest

Random forests can be defined as an ensemble learning system for the classification, regression, as well as other tasks, functioning through the construction of a wide range of the decision trees at training time and producing the class which is the individual trees classes (i.e. classification) or mean/average predictor (i.e. regression) mode. Random decision forests are correct for the decision trees' habit. RFs have been considered as one of the ways for averaging several deep decision trees that have been trained with the intentions to reduce the variance on different sections of one training set. Which comes at the cost of a slight rise in the bias and some interpretability lack, however, typically results in the substantial improvement of efficiency in the final model. Forests are like pulling decision tree algorithm attempts together. In this way, the teamwork of multiple trees increases the productivity of a single random tree [20] proposed hybrid feature selection for Random Forest depends on two measures Information Gain and Gini Index in different weight-based percentages. The key plan is to measure the Information Gain for all random selection features and then look for the best split point in the node that offers the best value for a Gini Index hybrid equation [22]. It proposed a random forest algorithm using an accuracy-based ranking that relies on the accuracy of a single tree from the previous Random Forest assessment. The proposed model consists of two primary stages, the first being the training process responsible for the development of the tree and the evaluation phase containing two test tiers (evaluation test and accuracy test) [23].

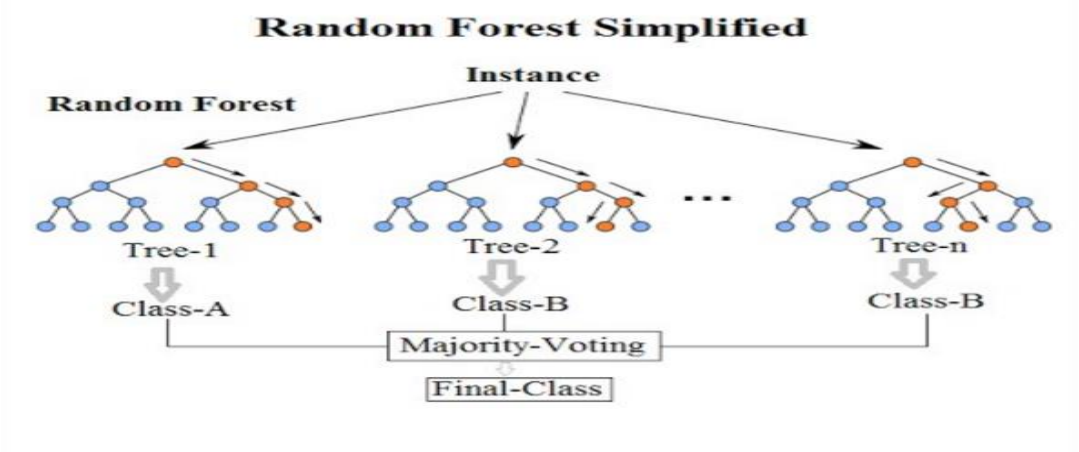

Figure 1. Random Forest Classifier [21]

4. Enhancement Performance of Random Forest Algorithm

One Hot Encoding is a process of exemplifying categorical values into binary numbers. Many learning algorithms either use distances between samples or learn a single weight per element. The former is true for linear models like logistic regression, which are simple to understand. It is a known fact that machine learning algorithms fail to work on categorical data and hence have to be converted to numbers where one hot encoding technique plays its significant role. As a natural reaction it is possible to opine on the use of integer coding directly but it has its limitations when used on relationships of the natural ordinal types. In one hot encoding technique, the categorical values in the data are directly assessed to integer values and each integer value is changed to a binary value. In this proposed system framework shown at Figure 3. The three classes are one hot encoded which resulted in two labels for binary, five labels for category and 9 labels for subcategory. The process of one-hot encoding is done through using one for the corresponding label and zero for the other labels, and the result vector is used as a multi classification problems where we have a vector of target classes. One-hot encoding makes the training process more effective and the built model would learn more efficient as it gives the network more expressive power to learn a probability-like number for each possible label value. This can help in both making the problem easier for the network to model. When a one hot encoding is used for the output variable, it offers a more nuanced set of predictions than a single label. Random forest classifier is then trained using the training set and the trained model is then tested used the testing set. The proposed system achieved higher accuracies which is 99.9% for binary label, 99.3% for category label and 95.8% for subcategory. To represent one-hot encoding process more formally, Consider $C = \{c^{(1)}, c^{(2)}, \dots, c^{(k)}\}$ represents corresponding labels for D shown in table 1, which is a given as input set. Also, let $D = \{ID_1, ID_2, \dots, ID_N\}$ represents N labeled network traffic data in industrial environments. And for each instance there are $F = \{F^{(1)}, F^{(2)}, \dots, F^{(m)}\}$ represents m features for the instance. If $C^{(1)} = Label_1$, while $C^{(2)} = Label_2$, and $C^{(3)} = Label_3$, and the other response values of C are either $Label_1$,or $Label_2$, or $Label_3$. One hot encoding process would assume a vector where the vector size equal the number of distinct labels, that is three in this example, and would refer to each label in the vector as 0 or 1 where 0 means that the data instance not correspond to that label and 1 means it have that label. So, $Label_1$ would be replace with vector [1 0 0], $Label_2$ would be replaced with vector [0 1 0] and $Label_3$ would be replaced with vector [0 0 1]. For generalization, $c^{(1)}$would be [1 0 0 …], $c^{(2)}$ would be [0 1 0 …] and $c^{(k)}$ would be [0 0 … 1]. This process is clarified in Table 1 and its encoding process. For more clarification, If we have a single categorical feature "Category", with values “Scan”, “Mirai”, “MITM”, “ARP” an “DOS”. Assume, without loss of generality, that these are encoded as 0, 1 and 2, 3, 4. The integer encoding is insufficient for categorical variables with no such ordinal relationship. In reality, encouraging the model to assume a normal ordering between categories and using this encoding which result in bad performance or unpredictable effects (predictions halfway between categories).

In this case, the integer representation can be encoded using a one-hot encoding. For each unique integer value, the integer encoded variable is discarded and a new binary variable is inserted. But with the one-hot encoding, the representation is [1, 0, 0, 0, 0], [0, 1, 0, 0, 0], [0, 0, 1, 0, 0], [0, 0, 0, 1, 0], [0, 0, 0, 0, 1].

Table 1. Input Data Set Example before and after One-hot Encoding

	Feature Vector						Target
ID	F1	F2	F3	F4	…	F^m	classes
01	F_1^1	F_1^2	F_1^3	F_1^4	…	F_1^m	C1
02	F_2^1	F_2^2	F_2^3	F_2^4	…	F_2^m	C2
03	F_3^1	F_3^2	F_3^3	F_3^4	…	F_3^m	C3
⋮	⋮	⋮	⋮	⋮	⋮ ⋮	⋮	⋮
N	F_N^1	F_N^2	F_N^3	F_N^4	…	F_N^m	Ck

Via One Hot Encoding

	Feature Vector						**Target**				
ID	**F1**	**F2**	**F3**	**F4**	**…**	**F^m**	**C1**	**C2**	**C3**	**….**	**Ck**
01	F_1^1	F_1^2	F_1^3	F_1^4	…	F_1^m	1	0	0	.....	0
02	F_2^1	F_2^2	F_2^3	F_2^4	…	F_2^m	0	1	0	.....	0
03	F_3^1	F_3^2	F_3^3	F_3^4	…	F_3^m	0	0	1	.....	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
N	F_N^1	F_N^2	F_N^3	F_N^4	…	F_N^m	0	0	0	…..	1

Algorithm 1 shows the algorithm proposed in the second framework where we one-hot encode the classes by creating a vector classes that have one for the corresponding classed and zero for the other classes and the resulting dataset is then input for the random forest algorithm. The second algorithm shows how we calculate the accuracy evaluation for the second approach where the true positives and true negatives are calculated by comparing the results of the dataset with results of the RF algorithm and if the position of the on in the result vector matches that of the dataset then the true are added by one. The true is then calculated

by looping through the testing dataset and then it's divided by the dataset size and the final percentage is then evaluated by multiplying to 100.

Randomized forests are a collection of regression trees and classifications which train on a training Dataset generated by random selection on the initial Data-set. When building a tree, the researchers use a collection of randomized test Data that contains no Record constraints from training data-set as a group for testing the trees within the forest. To calculate the consistency of every one of the trees, take the error rate in determining the form of input from the test community for the tree, and to determine the error rate of the forest, take the error rate for all the trees in the forest. The rule will split the data into two bits, allowing it to be checked with the same data and with the same precision as laboratory tests, according to the out of bag scale. These values are added to all of the trees in the random forest in order to define a new entry. Each tree predicts the class of this entry Record, and the forest then makes a decision based on the majority vote of the trees [24].

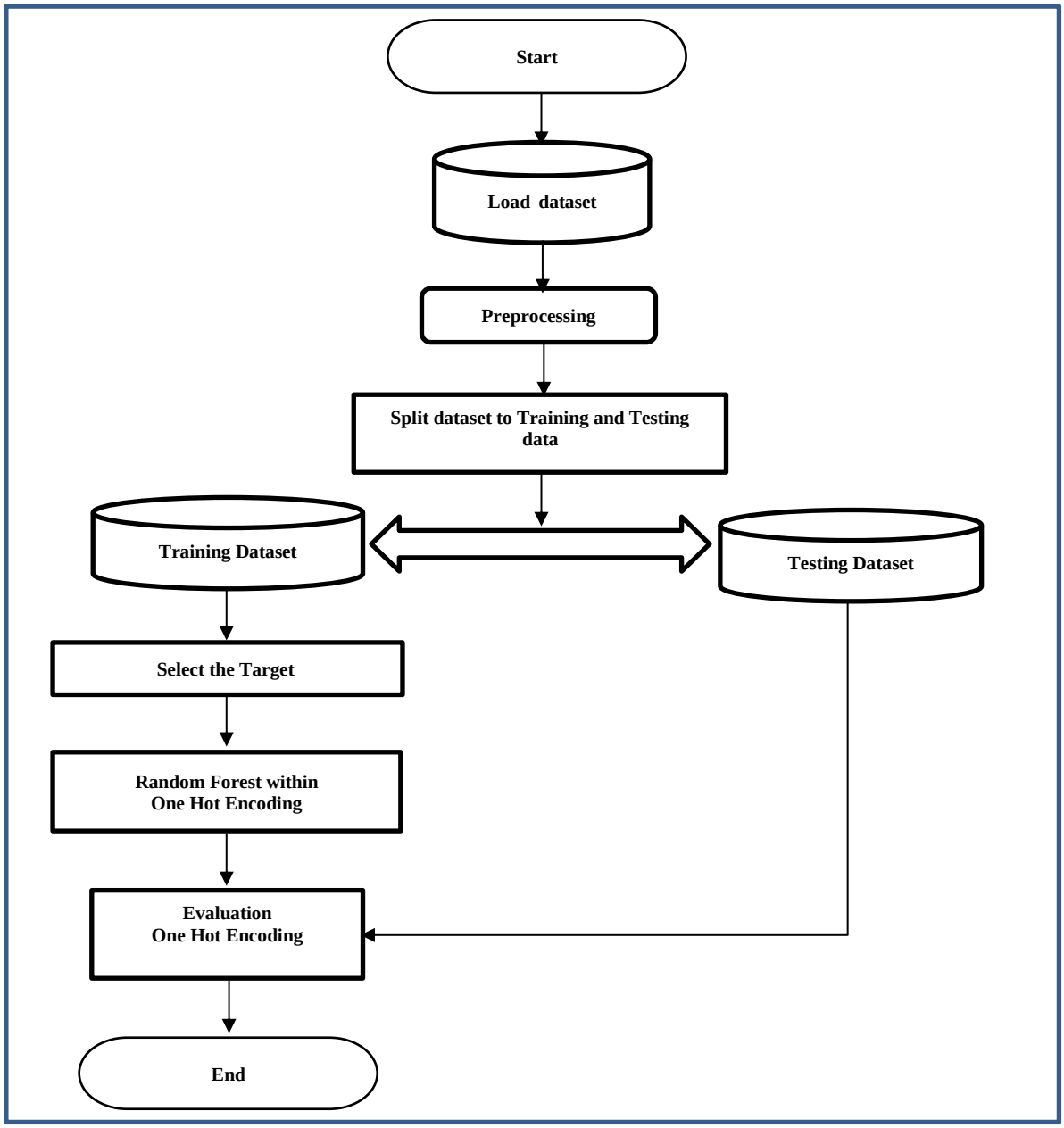

Figure 2. Enhancement Performance of Random Forest Algorithm Framework

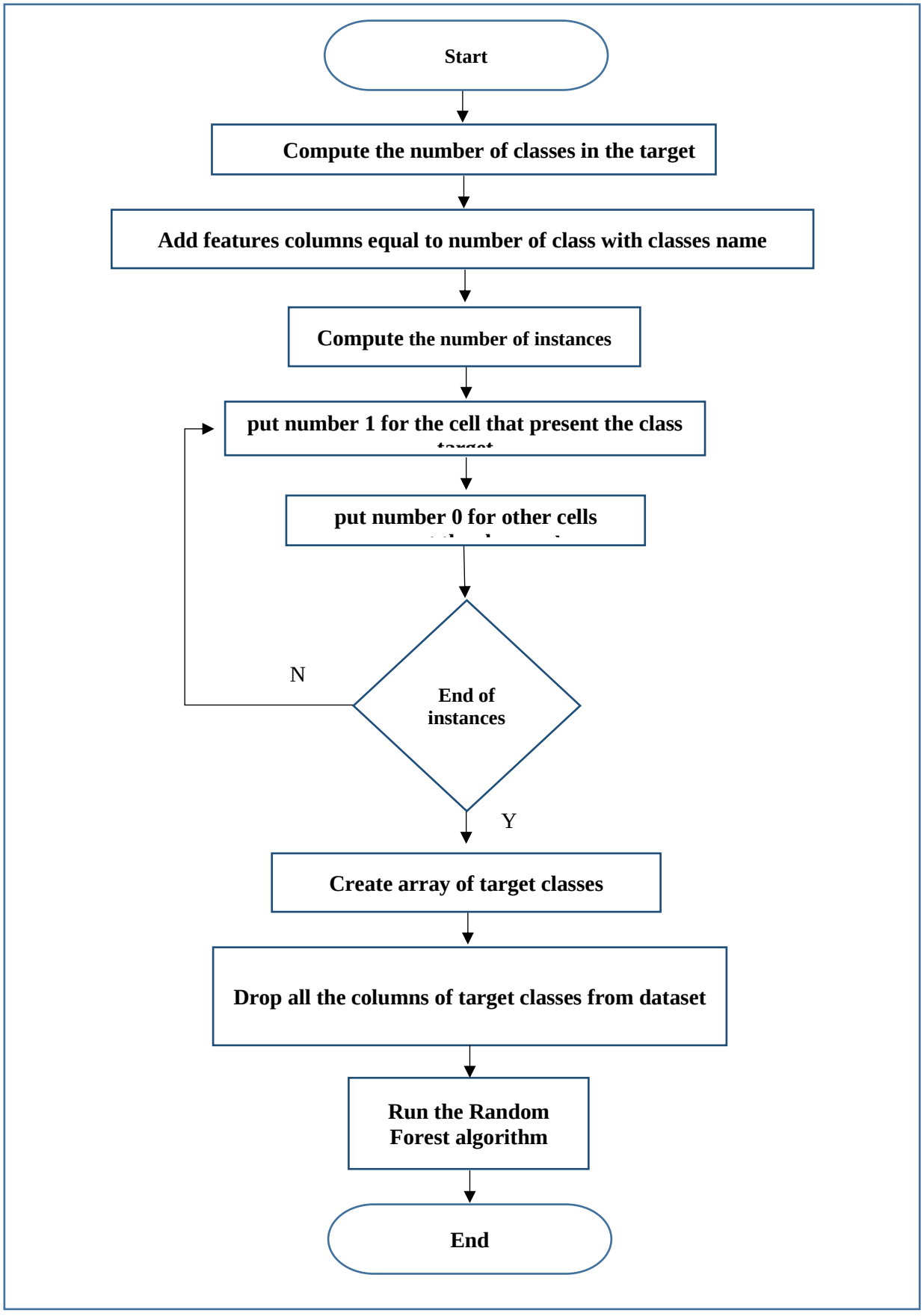

Figure 3. Enhancement Performance of Random Forest Algorithm Framework

Algorithm 1. Random Forest within One-hot Encoding

Random Forest within One Hot Encoding
Input: **Training Dataset, Number of Features, select target**
Output: **Tree of random forest**
Begin: Step 1: Count number of classes for target in the training dataset Step 2: Add features columns equal to number of classes with classes name in the training data-set Step 3: K= the number of instances in the training data-set Step 4: Loop i from 1 to K Begin: a. put number 1 for the cell that present the class b. put number 0 for other cells End Step 5: TG = Create array of target classes Step 6: TD = Drop all the columns for target classes from the training dataset Step 7: Call Random Forest algorithm (TD, TG) End

Algorithm 2. One-hot Encoding Evaluation

Evaluation One Hot Encoding
Input: **Tasting Dataset, T=Tree of Random Forest, Select Target**
Output: **Final accuracy for Rrandom Forest**
Begin: Step 1: Count number of classes for target in the testing data-set Step 2: Add features columns equal to number of classes with classes name in the testing data-set Step 3: K= the number of instances in the testing dataset Step 4: Loop i from 1 to K Begin: a. put number 1 for the cell that present the class b. put number 0 for other cells End Step 5: TG = Create array of target classes Step 6: TD = Drop all the columns for target classes from the tasting dataset Step 7: Predictions = Random Forest algorithm (T,TD, TG) Step 8: Right = 0 Step 9: Loop i from 1 to K Begin: if Predictions = TG; Then: Right = Right +1 End Step 10: Accuracy = (100*right/(K)) End

5. Experimental results

5.1. Dataset description

IoTID-20 data-set [12] includes intrusion and regular activities recorded by notebooks, tablets, and smartphones in a smart home IoT network with Wi-Fi router, SKT NGU computer, and EZVIZ camera. There are 83 attributes and 625,783 instances in the dataset. with the nominal attributes omitted, resulting in a dataset with 79 characteristics. The intrusion detection mark, the category label, and the sub category label are all present in the dataset. The IoTID20 dataset's binary, category, and sub-category labels are listed in Table 2.

Table 2. IoTID20 dataset

Binary	Category	SubCategory
Anomaly Normally	Anomaly-Scan Anomaly-Mirai Anomaly-MITM ARP Anomaly-DOS Normal	Hot Port Port OS ACK Flooding Host BruteForceg HTTP Flooding UDP Flooding MITM ARP Spoofing Synflooding Normal

Table 3 lists the precise distribution of dataset records between standard and intrusion operations. By adding more damaging risks, IoT systems have increased the attack surface. Denial of service (DoS), Man-in-the-Middle (MITM), Distributed DoS (DDoS), and active scanning were the most malicious activities injected and tracked to produce the dataset. The type of DoS attack that has been considered is one that floods synchronized (SYN) packets into TCP-based connections (TCP-based connections). SYN packets are typically utilized to create TCP connections between the communicating parties through reserving resources on both sides, primarily ports and buffers. It may be used to target the server's and/or victim machines' availability. Furthermore, flooding acknowledgment, Hyper Text Transfer Protocol (HTTP), and User Datagram Protocol (UDP) packets were used to introduce DDoS attacks in the form of IoT Mirai. Furthermore, a brute force attack has been used to decrypt data and reveal the confidentiality. MITM has been also used for the poisoning of the Address Resolution Protocol (ARP) table and map the attacker's Media Access Control (MAC) address to the router's Internet Protocol (IP) address. As a result, the intruder will impersonate the network router and disrupt interactions between network entities. The key goal of this attack is to sniff or manipulate data that is being transmitted [11]. The IoTID20 dataset is split into two parts: 75 percent preparation and 25% research.

Table 3. IoTID20 Distribution

		Binary		**Category**		**Sub_Categor**		**75%**		**25%**	
Dataset IoTID20	625.783	Anomaly	585.710	Anomaly-Scan	75,265	Hot Port	22,192	For Training	16,644	For Test	5,548
						Port OS	53,073	For Training	39,805	For Test	13,268
				Anomaly-Mirai	415,676	ACK Flooding	55,124	For Training	41,343	For Test	13,781
						Host BruteForceg	121,181	For Training	90,886	For Test	30,295
						HTTP Flooding	55,818	For Training	41,864	For Test	13,955
						UDP Flooding	183,553	For Training	137,665	For Test	45,888
				Anomaly-MITM ARP	35,377	MITM ARP Spoofing	35,377	For Training	26,533	For Test	8,844
				Anomaly-DOS	59,392	Synflooding	59,391	For Training	44,544	For Test	14,848
		Normally	40.073	Normal	40,073	Normal	40,073	For Training	30,055	For Test	10,018
								Total Sample	469,337	Total Sample	156,446

	for Training	for Testing

5.2. Results

Figure 4 shows a comparison between the proposed system for predicting the binary (normal/ anomaly) target and other state-of-the-art studies in terms of accuracy. The accuracy is specified by the total number of correct predictions divided by the total number of predictions. The accuracy of the proposed system and systems proposed by Farah [3] and Qaddoura et al. [13] discussed before in the related work section approaches 100%. The overall evaluation shows the superiority of our proposed algorithm.

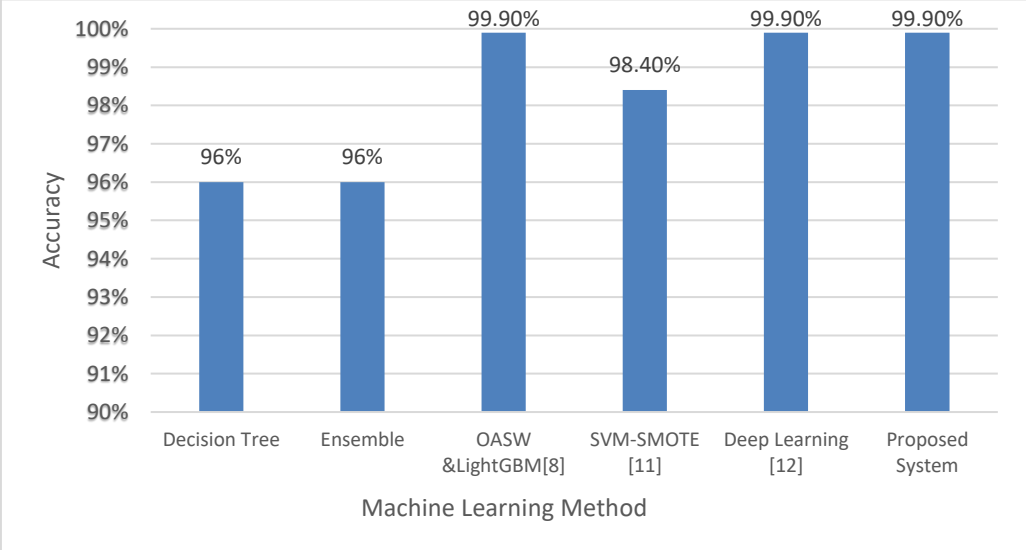

Figure 4. Comparison between the proposed system and other algorithms predicting the binary (normal/ anomaly) target in terms of accuracy and time.

Figure 5 shows a comparison between the proposed system for predicting the category (normal/ Scan, Mirai, MITM ARP, DOS) target and other machine learning algorithms in terms of accuracy. The accuracy of the proposed system and Decision Trees are the highest. The overall evaluation shows the superiority of our proposed algorithm.

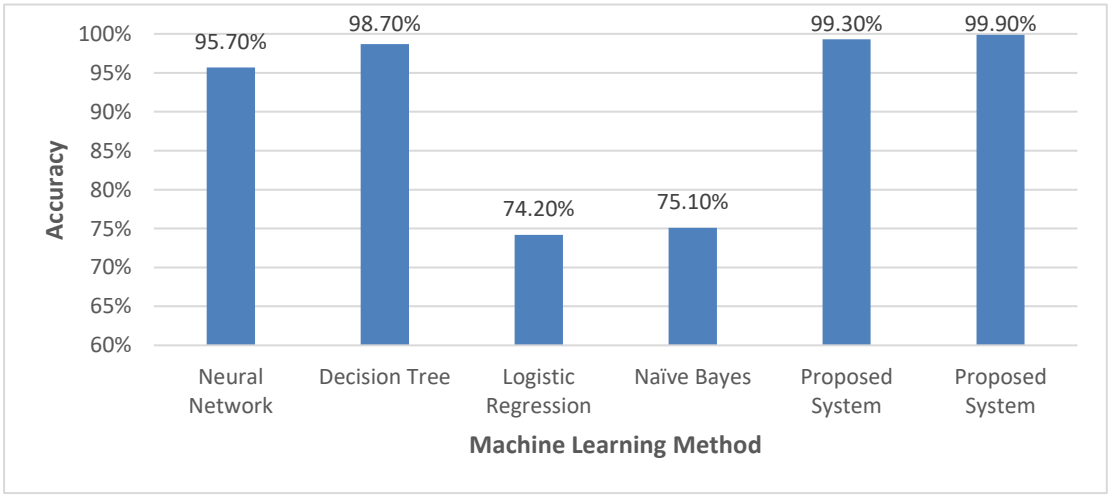

Figure 5. Comparison between the proposed system and other algorithms predicting the category (normal/ Scan, Mirai, MITM ARP, DOS) target in terms of accuracy and time.

Figure 6 shows the accuracy of algorithms proposed by Ullah and Mahmoud [12]for predicting sub-category (Normal/ Host Port/ Port OS/ ACK Flooding/ Host BruteForce/ HTTP Flooding/ UDP Flooding/ MITM ARP

Spoofing/ Synflooding) on the IoTID20 dataset, they achieved high accuracy using their decision tree, random forest, and ensemble algorithms, but their proposed system uses all the data in the training step which makes the algorithms suffer from overfitting and make it not dependable.
On the other hand, highly small microstrip filters and antennas [25-26], can be utilized to improve portability of IoT system with an efficient wireless communication.

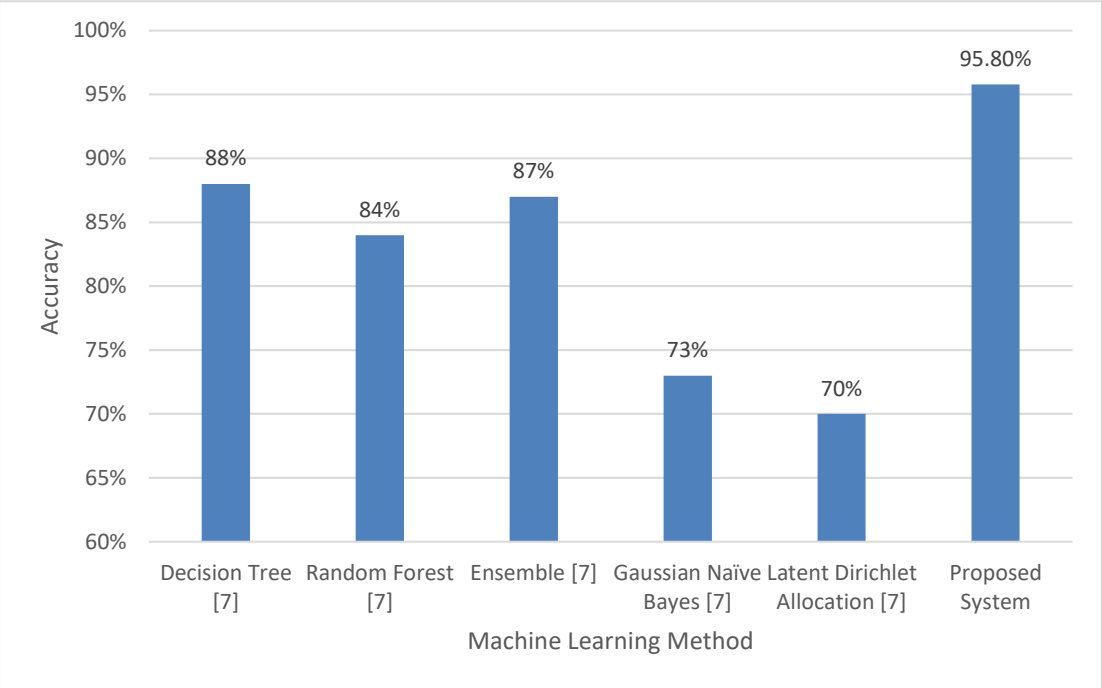

Figure 6. Accuracy of algorithms proposed by Ullah and Mahmoud [12] predicting sub-category (Normal/ Host Port/ Port OS/ ACK Flooding/ Host BruteForce/ HTTP Flooding/ UDP Flooding/ MITM ARP Spoofing/ Synflooding).

6. Conclusion

Intruders will initiate more disruptive cyber-attacks due to the rapid development of IoT gadgets. With disruptive operations, the attacker hoped to deplete the target IoT network's resources. Researchers and company owners are concerned about the reliability of IoT networks, which has a significant effect on availability of services that are provided by IoT devices as well as the safety of users connecting to the network. An intruder prevention mechanism protects the network by detecting malicious activity. The nominal attributes are omitted from the IoTID20 dataset, resulting in a dataset with 79 characteristics. Label, category, and subcategory are the three groups in the dataset. The data-set is then divided into two parts: training and testing. The three groups are hot encoded, yielding two binary labels, five category labels, and nine subcategory labels. The training set is then used for training the random forest classifier, and the trained model is then evaluated with the testing set. The suggested scheme reached higher accuracies, with a binary label accuracy of 99.9%, a division label accuracy of 99.3%, and a subcategory label accuracy of 95.8%. The methodology outperforms currently available state-of-the-art approaches.

References

[1] A.A.H. Mohamad, Y. S. Mezaal, S. F. Abdulkareem, "Computerized power transformer monitoring based on internet of things," International Journal of Engineering & Technology 7, no. 4, pp.2773-2778, 2018.

[2] H. T. Salim, N. A. Jasim, "Design and Implementation of Smart City Applications Based on the Internet of Things," *International Journal of Interactive Mobile Technologies (iJIM)*, vol. 15, no. 13, pp. 4-15, 2021.

[3] A. Farah, "Cross Dataset Evaluation for IoT Network Intrusion Detection," The University of Wisconsin-Milwaukee, 2020.
[4] H. Liu and B. Lang, "Machine learning and deep learning methods for intrusion detection systems: A survey," *applied sciences*, vol. 9, no. 20, p. 4396, 2019.
[5] B. K. Mohammed, M. B. Mortatha, A. S. Abdalrada, and H. ALRikabi, "A comprehensive system for detection of flammable and toxic gases using IoT," *Periodicals of Engineering Natural Sciences*, vol. 9, no. 2, pp. 702-711, 2021.
[6] H. Alaidaros, M. Mahmuddin, and A. Al Mazari, "An overview of flow-based and packet-based intrusion detection performance in high speed networks," in *Proceedings of the International Arab Conference on Information Technology*, 2011, pp. 1-9.
[7] J. Krimmling and S. Peter, "Integration and evaluation of intrusion detection for CoAP in smart city applications," in *2014 IEEE Conference on Communications and Network Security*, 2014, pp. 73-78: IEEE.
[8] A. S. Hussein, R. S. Khairy, S. M. Najeeb, and H. Salim,"Credit Card Fraud Detection Using Fuzzy Rough Nearest Neighbor and Sequential Minimal Optimization with Logistic Regression," International Journal of Interactive Mobile Technologies, vol. 15, no. 5, 2021.
[9] J. S. Abbasi, F. Bashir, K. N. Qureshi, M. N. ul Islam, and G. Jeon, "Deep learning-based feature extraction and optimizing pattern matching for intrusion detection using finite state machine," *Computers Electrical Engineering*, vol. 92, p. 107094, 2021.
[10] O. H. Yahya, H. Th, R. M. Al_airaji, and M. Faezipour, "Using Internet of Things Application for Disposing of Solid Waste," *International Journal of Interactive Mobile Technologies*, vol. 14, no. 13, pp. 4-18, 2020.
[11] R. Qaddoura, A. M. Al-Zoubi, I. Almomani, and H. Faris, "A Multi-Stage Classification Approach for IoT Intrusion Detection Based on Clustering with Oversampling," *Applied Sciences*, vol. 11, no. 7, p. 3022, 2021.
[12] I. Ullah and Q. H. Mahmoud, "A Scheme for Generating a Dataset for Anomalous Activity Detection in IoT Networks," in *Canadian Conference on AI*, 2020, pp. 508-520.
[13] R. Qaddoura, M. Al-Zoubi, H. Faris, and I. Almomani, "A Multi-Layer Classification Approach for Intrusion Detection in IoT Networks Based on Deep Learning," *Sensors*, vol. 21, no. 9, p. 2987, 2021.
[14] L. Yang and A. J. I. I. o. T. M. Shami, "A Lightweight Concept Drift Detection and Adaptation Framework for IoT Data Streams," 2021.
[15] N. A.Hussien, H. Salim, and F.Abed, "Monitoring the Consumption of Electrical Energy Based on the Internet of Things Applications," *International Journal of Interactive Mobile Technologies (iJIM)*, vol. 15, no. 7, 2021.
[16] N. Koroniotis, N. Moustafa, E. Sitnikova, and B. Turnbull, "Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-iot dataset," *Future Generation Computer Systems*, vol. 100, pp. 779-796, 2019.
[17] J. H. Assi and A. T. Sadiq, "Modified Artificial immune system as Feature Selection," *Iraqi Journal of Science*, vol. 59, no. 2A, pp. 733-738, 2018.
[18] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in *2009 IEEE symposium on computational intelligence for security and defense applications*, 2009, pp. 1-6: IEEE.
[19] J. H. Assi and A. T. Sadiq, "NSL-KDD dataset classification using five classification methods and three feature selection strategies," *Journal of Advanced Computer Science Technology Research*, vol. 7, no. 1, pp. 15-28, 2017.
[20] J. Friedman, T. Hastie, and R. Tibshirani, *The elements of statistical learning* (no. 10). Springer series in statistics New York, 2001.
[21] N. Kumar, "Random forest algorithm, an interactive discussion," ed.
[22] A. T. Sadiqâ and K. S. Musawi, "Modify Random Forest Algorithm Using Hybrid Feature Selection Method," *International Journal on Perceptive Cognitive Computing*, vol. 4, no. 2, pp. 1-6, 2018.
[23] K. S. Mohsen and A. T. Sadiq, "Random Forest Algorithm Using Accuracy-Based Ranking," *Journal of Computational Theoretical Nanoscience*, vol. 16, no. 3, pp. 1039-1045, 2019.
[24] R. Kaluri, D. S. Rajput, Q. Xin, K. Lakshmanna, S. Bhattacharya, T. R. Gadekallu, and P. K. R. Maddikunta, "Roughsets-based Approach for Predicting Battery Life in IoT," *arXiv preprint arXiv:.06026*, 2021.

[25] S. Shandal, Y. S. Mezaal, M. Kadim, and M. Mosleh, “New compact wideband microstrip antenna for wireless applications,” Adv. electromagn., vol. 7, no. 4, pp. 85–92, 2018.
[26] Y. S. Mezaal, and H. T. Eyyuboglu. "Investigation of new microstrip bandpass filter based on patch resonator with geometrical fractal slot, " *PloS one,* vol.11, no. 4, e0152615, 2016.