International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 7s
DOI: https://doi.org/10.17762/ijritcc.v11i7s.6976
***Article Received**: 28 March 2023 **Revised**: 14 May 2023 **Accepted**: 30 May 2023*

IOT Security Against Network Anomalies through Ensemble of Classifiers Approach

Saurav Verma[1], Dr. Chetana Prakash[2]
[1]Assistant Professor, MPSTME, NMIMS University,
Mumbai (M.H.) India
sauravtheleo@gmail.com
[2]Professor, BIET, VTU University
Davangere (K.A), India
chetana.p.m@gmail.com

Abstract— The use of IoT networks to monitor critical environments of all types where the volume of data transferred has greatly expanded in recent years due to a large rise in all forms of data. Since so many devices are connected to the Internet of Things (IoT), network and device security is of paramount importance. Network dynamics and complexity are still the biggest challenges to detecting IOT attacks. The dynamic nature of the network makes it challenging to categorise them using a single classifier. To identify the abnormalities, we therefore suggested an ensemble classifier in this study. The proposed ensemble classifier combines the independent classifiers ELM, Nave Byes (NB), and the k-nearest neighbour (KNN) in bagging and boosting configurations. The proposed technique is evaluated and compared using the MQTTset, a dataset focused on the MQTT protocol, which is frequently utilised in IoT networks. The analysis demonstrates that the proposed classifier outperforms the baseline classifiers in terms of classification accuracy, precision, recall, and F-score.

Keywords- Machine Learning, IoT, Intrusion Detection System, Extreme Learning Machine (ELM), Naïve Bayes (NB), KNN, Anomaly Detection.

I. INTRODUCTION

Over the last several years, there has been a considerable advancement in the design of both network infrastructures and intelligent devices. As a direct consequence of this, the Internet of Things (IoT) has emerged as a widely used and fashionable kind of technology that is now being used in a broad range of contexts to improve the quality of life of people [1]. Examples of smart applications include smartphone-controlled home appliances, "smart" cities and factories, "smart" health care, and large-scale infrastructures such as electricity grids and industrial systems [2], [3].

A major source of concern is the security of the Internet of Things, which has grown in importance as a result of its rapid development and subsequent emergence of numerous new attacks against networks and objects. An attack on or use of IoT could have serious consequences in social contexts in which it operates, making IoT-related services a critical infrastructure. Patients' safety may be compromised if cyberattacks target medical imaging equipment like computed tomography machines [4].

In order to reduce the occurrence of traffic irregularities, the engineers need to refine new detection methods for Internet of Things devices that have been compromised. Detection of network anomalies via the use of machine learning (ML) Anomaly-based approaches have just lately been shown to be useful. To put it another way, ML-based approaches have the ability to gather and evaluate both regular and aberrant IoT data. A baseline profile of normal and abnormal behavior in the observed environment is used for the detection of anomalies using supervised machine learning. This profile is based on the monitored environment. The comparison of system actions and the making of network event-related decisions are both based on this baseline. New or zero-day attacks can also be predicted using ML-based methods. There are numerous ways to secure IoT devices and networks using ML-based algorithms [5]. IoT anomalies can be detected using a variety of methods. The existing approaches to anomaly detection employ a variety of supervised machine learning algorithms. However, a single classifier will have a hard time spotting anomalies in such a dynamic and complex network. Consequently, we proposed an ensemble classifier technique that not only learns from data sets but also preserves the margin between training samples and classifier boundaries to deal with the network's dynamics.

The remaining portion of the paper is structured as follows: The related research on IoT security is compiled in Section 2. Section 4 describes the MQTTset dataset after Section 3 gives a summary of the mathematical underpinnings. Section 5 outlines the suggested approach. The experimental design,

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 7s
DOI: https://doi.org/10.17762/ijritcc.v11i7s.6976
***Article Received**: 28 March 2023 **Revised**: 14 May 2023 **Accepted**: 30 May 2023*

evaluation metrics, and outcomes are covered in Section 6. Finally, Section 7 brings the paper to a conclusion.

II. LITERATURE REVIEW

A lot of ideas have been generated as a result of analyzing traffic patterns in order to search for unusual occurrences. One of the oldest and most used ways [3, 4], that recognizes anomalies as connected rapid changes in the underlying distribution is the construction of traffic behavior models based on statistical techniques. These models are one of the most prevalent methods. Estimating dimensional distributions is challenging in any situation since there are so many different alternatives to consider. The data points associated with routine traffic all belong to the same cluster, whereas the data points associated with anomalous traffic belong to several clusters; hence, using clustering methods [5] might assist identify all of these data points.

As a consequence of this, clustering algorithms are not as successful as they once were in recognizing anomalous activity, which is the main objective of the majority of detection approaches. Rule-based techniques [6] are another prominent way that are used to develop classifiers. These techniques include the use of a set of rules. The disadvantage is that it requires a significant amount of a priori information in addition to calculations. Models for categorizing traffic have been developed using a wide variety of machine learning techniques [7, 8]. These strategies have been applied to both unsupervised and supervised machine learning models.

The effectiveness of neural networks is severely restricted due to a number of disadvantages, the most notable of which are the high computational cost and the high chance of overfitting. Support vector machine (SVM) approaches, which utilize nonlinear classification functions, allow the discovery of anomalous patterns in datasets and the categorization of data points based on their values. This is made possible via the use of nonlinear classification functions. SVMs are used to build classification models, and one of its goals is to maximize the difference that exists between the data points that belong to each class.

As a consequence of the quadratic optimization issue that has to be fixed, a number of different SVM detection variations are presented and assessed [9]. Last but not least, many real-time classification systems make use of Bayesian networks (BN) approaches because of the relative simplicity of these networks. These models are developed on the premise that different attributes are conditionally independent of one another. The chance that this assumption is true is used to categorize any two data points that are being compared.

A wide range of BNs have been introduced as part of various intrusion detection techniques. The majority of the models given in this article are not suited for sequence classification or the detection of anomalies. Only input instances are processed independently, and in doing so, the sequential presence of traffic data in these situations is ignored. Nevertheless, traffic data is a multivariate time series, and the anomalous patterns are constantly shifting throughout the course of time.

III. MATHEMATICAL BACKGROUND

In this section, the details of Extreme Learning Machine (ELM), Naïve Byes (NB), and k-nearest neighbor (KNN) are presented.

A. K-Nearest Neighbor (KNN):

In regard to statistical pattern recognition techniques, the Nearest Neighbor Algorithm consistently outperforms them all, regardless of the distributions from which the learning examples are derived. It involves a collection of instances that are both positive and negative. A new sample is listed in the measurement of the distance to the nearest training event, and the sample class is defined by that indicator. This definition is expanded by the k-NN classifier by considering the k closest points and allocating the majority symbol. The process of the KNN algorithm is as follows:

Suppose that there are M training classes which are represented as $C = \{c_1, c_2, \dots, c_M\}$, and N be the total number of used training samples. The combination of entries in training samples is represented by $S = \{s_1, s_2, \dots, s_N\}$. Let any entry in S is represented by a $d-$ dimensional feature vector $s_i = \{a_1, a_2, \dots, a_d\}$, here a_i represents the i^{th} attribute.

Let, we have to classify a testing samples T, amongst the M classes, to do this, the following steps are needed to be performed:

1. Compute the similarity between sample T and each training sample using the formula below:

$$SIM(i) = \frac{\sum_{j=1}^{d} s_i(j) \cdot T(j)}{\sqrt{\left(\sum_{j=1}^{d} T(j)\right)^2} \cdot \sqrt{\left(\sum_{j=1}^{d} T(j)\right)^2}} \qquad (1)$$

   Where $SIM(i)$ represents the similarity between training sample $s_i \in S$, and test sample T.

2. Select the top K elements of SIM in SIM_{TopK} and arrange them in descending order $SIM_{TopK} = \{s_{T1}, s_{T2}, \dots, s_{TK}\}, \forall i (s_{Ti} > s_{Tj} \wedge i > j)$ and respective classes of the samples in SIM_{TopK} in $C_{TopK} = \{c_{T1}, c_{T2}, \dots, c_{TK}\}$.
3. Count the appearance of each class of C, in C_{TopK} and classify the T to the class having the highest count.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 7s
DOI: https://doi.org/10.17762/ijritcc.v11i7s.6976
***Article Received**: 28 March 2023 **Revised**: 14 May 2023 **Accepted**: 30 May 2023*

B. Naive Bayes (NB) Classifier:

The term "basic Bayesian algorithm" refers to the Naive Bayes (Lewis 1992) classification algorithm. Naive Bayes' categorization has proven to be quite effective. Concerning the anomaly detection problem, the instance $s_i \in S$ corresponds to the class C of the training dataset. Instance s_i can be displayed in terms of d −dimensional feature vector. The feature vector is defined as $\{a_1, a_2, \dots, a_d\}$ here a_i represents the i^{th} attribute. A class label (c_i) is affixed to each instance (s_i) where $c_i \in C = \{c_1, c_2, \dots, c_M\}$ refers to a class label set. Estimates from the Naive Bayes classifier $P(c_j|s_i)$, which represents the likelihood of an event s_i and that is belongs to a class c_j. Through the Bayes rule, the predicted label of a given test sample s_T can be calculated as:

$$C_{predicted} = \arg\max_{c_k \in C} \frac{P(C = c_k) \prod_{i=1}^{d} P(s_T(i)|C = c_k)}{\sum_{j=1}^{M} \left[P(C = c_j) \prod_{i=1}^{d} P(s_T(i)|C = c_j)\right]} \quad (2)$$

Since the denominator does not depend on c_k the Eq. (2) can further be simplified to the following:

$$C_{predicted} = \arg\max_{c_k \in C} P(C = c_k) \prod_{i=1}^{d} P(s_T(i)|C = c_k) \quad (3)$$

C. Extreme Learning Machine

"The Extreme Learning Machine (ELM) is a new approach to machine learning, which involve a single hidden layer feedforward neural network (SLFN)" [15]. It is rapid to train and performs with accuracy comparable to that of support vector machines (SVMs) [16]. ELM is a conventional three-layer feedforward architecture that was first put into use in the year 2006 [17]. The input layer is its first layer, while the hidden layer, also known as the second layer, projects the data from the input layer into a higher dimensional space using a staggering number of nonlinear sigmoid neurons. The topmost layer is the output, and the neurons that make up this layer have linear input and output connections. The structure of the ELM is shown in Figure 1.

$$y(p) = \sum_{j=1}^{m} \beta_j g\left(\sum_{i=1}^{n} w_{i,j} x_i + b_j\right) \quad (4)$$

Where β_i stands for the weights between the input layer and the hidden layer, and β_j for the weights between the output layer and the hidden layer. The threshold value of the neurons in the hidden layer is given as b_j, and activation function is given as $g(.)$ Random assignments are made to both the weights of the same input layer $(w_{i,j})$ and the bias (b_j). The activation function, denoted by the symbol $g(.)$, is assigned at the start of the input layer neuron number (n) and the hidden layer neuron number (m). Now, using this knowledge, by merging and rearranging the parameters that are known to be in equilibrium, the output layer has become as it is shown in equation (6) [18].

$$H(w_{i,j}, b_j, x_i) = \begin{bmatrix} g(w_{1,1}x_1 + b_1) & \cdots & g(w_{1,m}x_m + b_m) \\ \vdots & \ddots & \vdots \\ g(w_{n,1}x_n + b_1) & \cdots & g(w_{n,m}x_m + b_m) \end{bmatrix} \quad (5)$$

$$y = H\beta \quad (6)$$

The goal of every single model of the training algorithm is to achieve the lowest feasible error rate. The error function of the output y_p, which was calculated from the actual output value y_o in ELM is $\sum_k^s (y_o - y_p)$ (with "s": training data number) and $\left\| \sum_k^s (y_o - y_p)^2 \right\|$ can be minimized.

For each of these functions, the output y_p that is generated from the actual output value y_o has to be equivalent to the output y_p. When this occurs, the unknown parameter in equation (8) known as the H matrix might be an extremely low probability matrix. This shows that the number of features included in each data point cannot be equal to the quantity of data points present in the training set. As a result, calculating the inverse of H and coming up with weights (b) will be a difficult task. The pseudo-inverse of the matrix H can be computed using the Moore-Penrose Inverse algorithm, which will allow us to escape from this predicament. Therefore, the output weights may be calculated by using the equation $\beta = y \times MPI\ (H)$.

IV. Experimental Datasets

The most popular protocols for machine-to-machine communication on the Internet of Things (IoT) is the Message Queuing Telemetry Transport (MQTT) protocol[6].

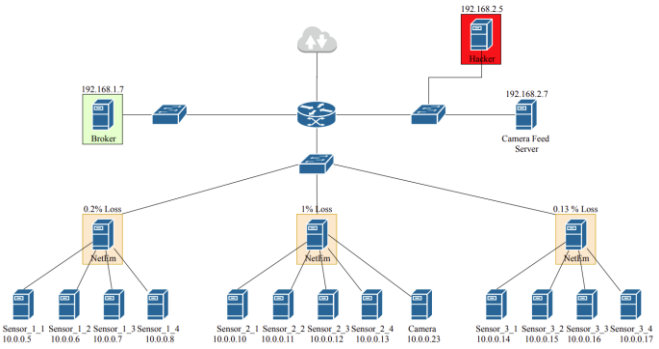

Figure 1: MQTT Network Architecture.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 7s
DOI: https://doi.org/10.17762/ijritcc.v11i7s.6976
***Article Received**: 28 March 2023 **Revised**: 14 May 2023 **Accepted**: 30 May 2023*

TABLE 1: DETAIL OF DATA INSTANCES DISTRIBUTION.

Name of the file		**Normal**	scan A (aggressive)	Scan_sU (UDP)	sparta
File Size (Pcap)		192.5 MB	16.2 MB	41.3 MB	3.4 GB
Packet Count	**Normal**	1056220	7058	210809	947167
	Attack	0	40620	22426	19728933
Count of Uni-flow	**Normal**	171826	11550	34407	154165
	Attack	0	39787	22426	28222
Count of Bi-flow	**Normal**	86008	5776	17220	77192
	Attack	0	19905	22424	14106

The MQTT-IoT-IDS2020 dataset is the first of its kind to simulate a network built on top of MQTT. The dataset is generated via the usage of a simulated MQTT network architecture (Fig. 2). The network is comprised of a total of thirteen nodes, which include a broker, a simulated camera, an adversary, and twelve sensors. The following are the five such situations that have been recorded:

1. Normal operation
2. Aggressive scan (Scan_A)
3. UDP scan (Scan_sU)
4. Sparta SSH brute-force (Sparta)
5. MQTT brute-force attack (MQTT_BF)

The features are extracted once the raw pcap files have been saved. Three abstraction layers of feature extraction from the raw pcap data are:

1. packet features
2. unidirectional flow features
3. bidirectional flow features

Applications that use Machine Learning (ML) will find the csv feature files of the dataset to be quite useful. Additionally, the raw pcap files are suitable for a more in-depth examination of MQTT Internet of Things network traffic as well as accompanying threats [7].

Table 2 provides a summary of the most important features that were extracted from the packets. For both unidirectional and bidirectional systems, the feature set can be found in Table 2, columns 5 and 6, respectively.

In the case of flows in both directions, some characteristics, denoted by the asterisk (*), possess two distinct values: one for the forward flow and one for the backward flow. It is essential that you keep this in mind. Prefixes such as "fwd" (means "forward") and "bwd" (means "backward") are used to denote the direction in which the feature is traveling [12]. In addition, the distribution of occurrences is shown in Table 1.

In Table 2, the following features have been omitted in order to minimize the impact of specific features. These features include IP addresses, protocols, and MQTT flags. 25% of the data is utilised for testing, while 75% is used for training.

TABLE 2: DESCRIPTION OF FEATURES [22]

Feature	**Definition**	**Type**	**Packet**	**Uni-flow**	**Bi-flow**
ip_src	Source IP Address	Text	√	√	√
ip_dest	Destination IP Address	Text	√	√	√
protocol	Last layer protocol	Text	√		
ttl	Time to live	Integer	√		
ip_len	Packet Length	Integer	√		
ip_flag_df	Don't fragment IP flag	Binary	√		
ip_flag_mf	More fragments IP flag	Binary	√		
ip_flag_rb	Reserved IP flag	Binary	√		
prt_src	Source Port	Integer	√	√	√
prt_dst	Destination Port	Integer	√	√	√
proto	Transport Layer protocol (TCP/UDP)	Integer		√	√
tcp_flag_res	Reserved TCP flag	Binary	√		
tcp_flag_ns	Nonce sum TCP flag	Binary	√		
tcp_flag_cwr	Congestion Window Reduced TCP flag	Binary	√		
tcp_flag_ecn	ECN Echo TCP flag	Binary	√		
tcp_flag_urg	Urgent TCP flag	Binary	√		
tcp_flag_ack	Acknowledgement TCP flag	Binary	√		
tcp_flag_push	Push TCP flag	Binary	√		
tcp_flag_syn	Reset TCP flag	Binary	√		
tcp_flag_fin	Synchronization TCP flag	Binary	√		
num_pkts	Finish TCP flag	Integer		√	*
mean_iat	Number of Packets in the flow	Decimal		√	*
std_iat	Average inter arrival time	Decimal		√	*
min_iat	Standard deviation of inter arrival time	Decimal		√	*
max_iat	Minimum inter arrival time	Decimal		√	*
num_bytes	Maximum inter arrival time	Integer		√	*
num_psh_flags	Number of bytes	Integer		√	*
num_rst_flags	Number of push flag	Integer		√	*
num_urg_flags	Number of reset flag	Integer		√	*
mean_pkt_len	Number of urgent flag	Decimal		√	*
std_pkt_len	Average packet length	Decimal		√	*
min_pkt_len	Standard deviation packet length	Decimal		√	*
max_pkt_len	Minimum packet length	Decimal		√	*
mqtt_message type	Maximum packet length	Integer	√		
mqtt_message lenght	MQTT message type	Binary	√		
mqtt_flag_uname	MQTT message length	Binary	√		
mqtt_flag_passwd	User Name MQTT Flag	Binary	√		
mqtt_flag_retain	Password MQTT flag	Binary	√		
mqtt_flag_qos	Will retain MQTT flag	Integer	√		
mqtt_flag_willflag	Will QoS MQTT flag	Binary	√		
mqtt_flag_clean	Will flag MQTT flag	Binary	√		
mqtt_flag_reserved	Clean MQTT flag	Binary	√		
is_attack	Reserved MQTT flag	Binary			

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 7s
DOI: https://doi.org/10.17762/ijritcc.v11i7s.6976
***Article Received**: 28 March 2023 **Revised**: 14 May 2023 **Accepted**: 30 May 2023*

V. PROPOSED METHODOLOGY

Figure 3 presents the anomaly detection models that have been proposed. The objective of the categorization process is to assign the test labels to one of many predetermined categories.

The suggested model's classification procedure is broken down into its individual phases and shown in Figure below. First, the raw data that was gathered is processed in order to extract the features and labels, and then the dataset is split into two sets, a training set and a testing set, as shown in figure 3. During the training phase, the ensemble classifier is taught using a training set that contains a predetermined feature set and its associated labels. This set is known as the training set.

On the test dataset, the trained ensemble classifier is used as an anomaly detector. During the testing phase, the trained classifiers are put to the test by applying the testing dataset to their analysis in order to provide classification labels. In the last stage of the procedure, an evaluation of the proposed model's performance is carried out. This is accomplished by contrasting the newly created set of labels from the testing phase with the testing set labels that were used initially.

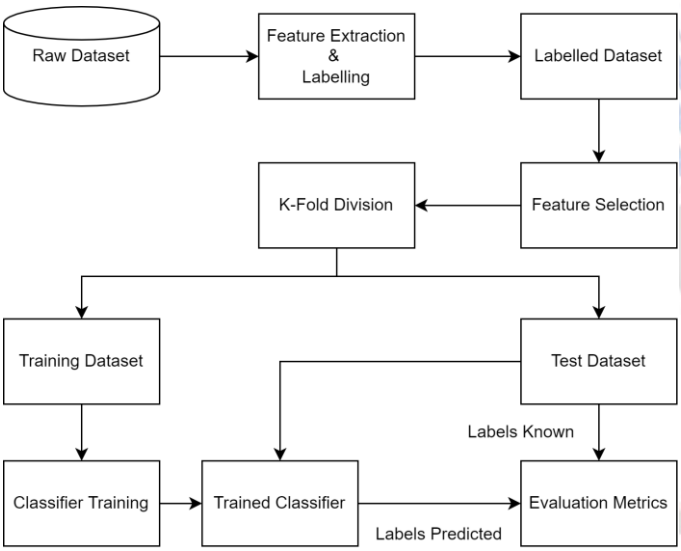

Figure 2: Proposed Classification Process.

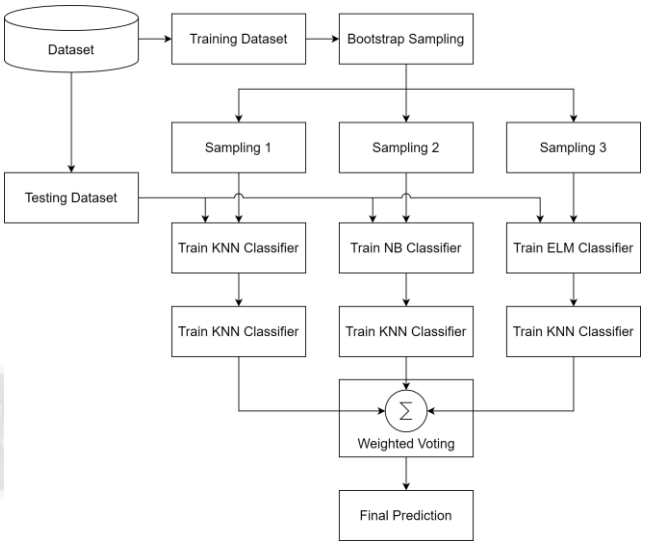

Figure 3: Ensemble classifiers.

1) Ensemble Classifier:

We perceive ensemble methods as a collection of machine learning algorithms that pool judgements to enhance system performance. Multiple classifiers, multi-strategy, the fusion of classifiers, combination, aggregated, integrative, etc. are additional names given to concepts with similar connotations in the literature. In this article, we refer to all classification techniques collectively. Voting, or weighted voting, is the simplest approach to combine several learning strategies. The primary concept of ensemble learning holds that no single strategy or method can consistently outperform any other approach, hence mixing different strategies will improve the performance of the final classifier. As a result, an ensemble classifier outperforms a single base classifier in general. The efficiency of ensemble approaches is significantly impacted by the limitations of each individual basic classifier.

The precision and variety traits of the fundamental learners have a significant impact on how well ensemble techniques function. Many studies have demonstrated that decision-making systems have a tendency to produce diverse classifiers in response to slight training modifications, making them suitable candidates for the ensemble system's fundamental learning [33, 34, 35]. The simplest method is to create various base classifiers by manipulating the training data. The two most popular ensemble approaches, bagging and boosting, are used in this study to detect IOT anomalies.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 7s
DOI: https://doi.org/10.17762/ijritcc.v11i7s.6976
Article Received: 28 March 2023 Revised: 14 May 2023 Accepted: 30 May 2023

Input: Integer N (total iterations), Training Dataset $< x, y >$, Classification Algorithm CL

$For\ i\ =\ 1 \ldots N$

$The\ training\ dataset\ D\ is\ sampled$

$into\ a\ subset\ S\ of\ size\ N.$

$End\ for$

Creates a classifier $C(x)$ from S.

By combining the j classifiers, create the final classifier $C^*(x)$.

Using the supplied data x, forecast a class label as follows.:

$$C^*(x) = arg \max_y \sum_{i=1}^{N} C_i(x) = y$$

Output: $C^*(x)$.

Figure 4: Bagging algorithm Pseudo-codes

Input: Integer N (total iterations), Training Dataset $< x, y >$, Classification Algorithm CL.

Create input initial weights w_i for all $x_i \in D$ equal to $1/N$.

$For\ i\ =\ 1 \ldots N$

$$weighted\ error\ estimate\ err_i = \frac{\sum w_i\ for\ incorrectly\ classified\ x_i}{\sum_{i=1}^{N} w_i}$$

Compute classifier weight

$$\alpha_i = \frac{1}{2}\log\left(\frac{1 - err_m}{err_m}\right)$$

When instances are accurately classified, the weight is

$$w_i = w_i e^{-\alpha_i}$$

When cases are misclassified, the weight is

$$w_i = w_i e^{\alpha_i}$$

w_i was normalised to demonstrate that so that

$$1 = \sum w_i$$

$End\ for$

Construct final classifier $C^*(x)$ using a weighted average of the individual $C_i(x)$ votes based on the accuracy of the class it correctly identified.

Output: $C^*(x)$.

Figure 5: AdaBoost algorithm's pseudo-codes.

Bagging: By randomly swapping N elements for the original D-training dataset, bagging tried to reorder the training details. For bootstraps, substitution sets are referred to as replicates since certain instances might appear more than once while others would not. The final attribute $C^*(x)$ classification consisting by aggregating $C(x)$ and each voting with an equal weight of $C_i(x)$. The bagging algorithm's pseudo-codes are displayed in Figure 4.

Boosting: The method first gave each instance of the training data an initial value of 'x'. The weight remains constant. The learning method returns an $C_i(x)$ classification after each iteration while attempting to reduce the overall weighted error. The training instance weights have been updated using the weighted $C_i(x)$ error calculation. x_i. $x_i's$ weight increases in accordance with how the classifier performs, which gives a larger weight to incorrectly classified data and a smaller weight to accurate data.

The final classifier $C^*(x)$ is created using the weighted vote of individually attribute of $C(x)$. Figure 5 displays the AdaBoost algorithm's pseudo-codes.

VI. PERFORMANCE EVALUATION METRICS

Four common measures are used to evaluate the effectiveness of classifiers: accuracy, precision, recall and F1 are define in equation number 8,9,10,7 respectively. The capacity of a classifier to make accurate predictions about normal and abnormal conditions is measured by its accuracy. The projected correctness of the label is what is meant by "accuracy." The level of recall is used to evaluate how comprehensive the category is.

$$F = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \quad (7)$$

$$Accuracy = \frac{|TP + TN|}{|TP + TN + FP + FN\ |} \quad (8)$$

$$Precision = \frac{|TP|}{|TP + FP\ |} \quad (9)$$

$$Recall = \frac{|TP|}{|TP + FN|} \quad (10)$$

An effective way to balance accuracy and memory is to employ the measurement known as the F-Score, which is the harmonic mean of precision and recall.

The term "true positive", "true negative", "false positive", and "false negative", respectively, are denoted by the letters TP, TN, FP, and FN. They are defined as follows:

- TP: The number of occasions in which the classifier correctly detected an anomaly as being present in the data.
- TN: The number of occurrences of normality that the classifier was able to accurately identify as normal.
- FP: the number of typical examples that the classifier erroneously categorized as anomalous
- FN: The number of out-of-the-ordinary occurrences that the classifier mistakenly categorized as typical.

These can also be defined by a confusion matrix.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 7s
DOI: https://doi.org/10.17762/ijritcc.v11i7s.6976
***Article Received**: 28 March 2023 **Revised**: 14 May 2023 **Accepted**: 30 May 2023*

TABLE 3: CONFUSION MATRIX

		Known Labels	
		True (Anomaly)	False (Normal)
Classifier's identification result	Positive (Anomaly)	TP	FP
	Negative (Normal)	FN	TN

TABLE 2: PERFORMANCE COMPARISON FOR UNIDIRECTIONAL FEATURES

	Accuracy (%)				Precision			
Attack	**SVM**	**KNN**	**NB**	**Proposed**	**SVM**	**KNN**	**NB**	**Proposed**
Normal	98.34	99.76	99.76	100.00	1.00	1.00	0.99	1.00
Scan_A	98.52	99.40	98.39	99.91	0.72	0.95	0.99	0.99
Scan_sU	99.83	99.04	99.45	99.97	1.00	0.99	0.98	1.00
Sparta	100.0	100.0	100.0	100.0	1.00	1.00	1.00	1.00
MQTT_BF	99.45	99.68	99.09	99.94	0.98	0.97	0.98	1.00
Average	99.228	99.576	99.538	99.964	0.94	0.982	0.988	0.998
	Recall				**F-Score**			
Attack	**SVM**	**KNN**	**NB**	**Proposed**	**SVM**	**KNN**	**NB**	**Proposed**
Normal	0.93	0.99	1.00	1.00	0.97	0.99	1.00	1.00
Scan_A	1.00	0.93	0.81	0.99	0.84	0.94	0.89	0.99
Scan_sU	1.00	0.99	0.99	0.99	1.00	0.99	0.98	0.99
Sparta	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
MQTT_BF	1.00	0.97	1.00	1.00	0.99	0.98	0.99	1.00
Average	0.986	0.976	0.96	0.996	0.96	0.980	0.972	0.996

TABLE 1: PERFORMANCE COMPARISON FOR BIIDIRECTIONAL FEATURES

	Accuracy (%)				Precision			
Attack	**SVM**	**KNN**	**NB**	**Proposed**	**SVM**	**KNN**	**NB**	**Proposed**
Normal	97.16	99.70	99.57	99.88	1.00	1.00	0.93	1.00
Scan_A	97.33	99.58	97.36	99.94	0.45	1.00	0.98	1.00
Scan_sU	99.59	99.68	99.84	99.93	1.00	0.94	0.99	0.99
Sparta	100.0	99.36	99.82	99.74	1.00	0.99	0.99	0.99
MQTT_BF	99.40	99.95	98.74	99.64	0.98	0.98	0.97	0.99
Average	98.696	99.654	99.066	99.826	0.886	0.982	0.972	0.994
	Recall				**F-Score**			
Attack	**SVM**	**KNN**	**NB**	**Proposed**	**SVM**	**KNN**	**NB**	**Proposed**
Normal	0.88	0.99	1.00	0.99	0.94	0.99	0.96	1.00
Scan_A	0.99	0.92	0.65	0.99	0.62	0.96	0.78	0.99
Scan_sU	0.93	1.00	0.98	1.00	0.96	0.97	0.99	0.99
Sparta	1.00	0.99	1.00	1.00	1.00	0.99	1.00	1.00
MQTT_BF	1.00	0.99	0.99	0.99	0.99	0.98	0.98	0.99
Average	0.96	0.978	0.924	0.994	0.902	0.978	0.942	0.994

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 7s
DOI: https://doi.org/10.17762/ijritcc.v11i7s.6976
***Article Received**: 28 March 2023 **Revised**: 14 May 2023 **Accepted**: 30 May 2023*

VII. RESULTS ANALYSIS

The proposed approach has been tested against three cutting-edge machine learning algorithms. The most advanced machine learning methods include Support Vector Machine (SVM), Naive Bayes (NB), and k-Nearest Neighbours (k-NN). Table 4 and 5 details the total evaluation outcomes for each ML approach with packet, unidirectional, and bidirectional features. The proposed method clearly improves performance for both unidirectional and bidirectional datasets.

The results demonstrate the classifiers perform significantly better with unidirectional features, according to more in-depth analyses of precision, recall, and F1-score.

Comparing the classifiers performances for the unidirectional dataset the proposed classifier performs best for the average accuracy. However, for Sparta attack all classifiers achieves 100% accuracy whereas for Scan_A and Scan_sU attacks the KNN achieves the highest accuracy of 99.94%. the lowest accuracy of 98.34% is achieved by the SVM for Non-Attack condition. Like accuracy the proposed classifier also performs the best for average precision. It achieves the precision of 1.0 for all conditions except the Scan_A where it scores 0.99. while the SVM achieves the worst precision of 0.72 for Scan_A attack.

The recall scores shows that the SVM achieves the 1.0 for all conditions except for Normal where it performs the worst and scores only 0.93. Finally, for the F-Score the proposed classfier outperforms all others by achieving aa average score of 0.996.

For the bidirectional dataset the proposed classifier performs best for the average accuracy. However, only SVM achieves the 100% accuracy for Sparta attack whereas for MQTT_BF attack the KNN achieves the highest accuracy of 99.95%. The lowest accuracy of 96.16% is achieved by the SVM for Non-Attack condition. Like accuracy the proposed classifier also performs the best for average precision. It achieves the precision of 1.0 for two conditions and 0.99 for rest of three conditions. while the SVM achieves the worst precision of 0.45 for Scan_A attack. The recall scores shows that the NB performs worst and achieves only 0.65 for Scan_A conditions. Finally, for the F-Score the proposed classifier outperforms all others by achieving aa average score of 0.994.

VIII. CONCLUSION

Using our research, we presented an ensemble classifier-based technique in this paper for identifying abnormalities in IoT networks. The ensemble classifier is developed through the combination of the KNN classifier, the NB classifier, and the ELM classifier. In the last step of this process, the performance of the proposed method is assessed using the MQTT IoT intrusion detection dataset. This evaluation takes place for two distinct feature sets that are respectively labelled unidirectional and bidirectional. The results of the experiments show that the suggested algorithm performs much better than the SVM, HMM, and NB algorithms. Furthermore, the proposed method offers a performance that is significantly more consistent across all the datasets and features. The results of the experiments reveal that the suggested approach delivers considerably enhanced performance when compared to the techniques that are already in use, regardless of whether the datasets have balanced or non-balanced class distributions.

REFERENCES

[1] A. B. Pawar and S. Ghumbre, “A survey on IoT applications, security challenges and counter measures,” in 2016 International Conference on Computing, Analytics and Security Trends (CAST), Dec. 2016, pp. 294–299. doi: 10.1109/CAST.2016.7914983.

[2] I. Vaccari, S. Narteni, M. Aiello, M. Mongelli, and E. Cambiaso, “Exploiting Internet of Things Protocols for Malicious Data Exfiltration Activities,” IEEE Access, vol. 9, pp. 104261–104280, 2021, doi: 10.1109/ACCESS.2021.3099642.

[3] Mondal, D., & Patil, S. S. (2022). EEG Signal Classification with Machine Learning model using PCA feature selection with Modified Hilbert transformation for Brain-Computer Interface Application. Machine Learning Applications in Engineering Education and Management, 2(1), 11–19. Retrieved from http://yashikajournals.com/index.php/mlaeem/article/view/20

[4] M. J. Kaur and P. Maheshwari, “Building smart cities applications using IoT and cloud-based architectures,” in 2016 International Conference on Industrial Informatics and Computer Systems (CIICS), Mar. 2016, pp. 1–5. doi: 10.1109/ICCSII.2016.7462433.

[5] T. Mahler et al., “Know Your Enemy: Characteristics of Cyber-Attacks on Medical Imaging Devices,” arXiv:1801.05583 [cs], Feb. 2018, Accessed: Apr. 24, 2022. [Online]. Available: http://arxiv.org/abs/1801.05583

[6] J. Asharf, N. Moustafa, H. Khurshid, E. Debie, W. Haider, and A. Wahab, “A Review of Intrusion Detection Systems Using Machine and Deep Learning in Internet of Things: Challenges, Solutions and Future Directions,” Electronics, vol. 9, no. 7, Art. no. 7, Jul. 2020, doi: 10.3390/electronics9071177.

[7] Mr. Anish Dhabliya. (2013). Ultra Wide Band Pulse Generation Using Advanced Design System Software . International Journal of New Practices in Management and Engineering, 2(02), 01 - 07. Retrieved from http://ijnpme.org/index.php/IJNPME/article/view/14

[8] V. Chandola, A. Banerjee, and V. Kumar, “Anomaly detection: A survey,” ACM Comput. Surv., vol. 41, no. 3, p. 15:1-15:58, Jul. 2009, doi: 10.1145/1541880.1541882.

[9] H. Hajji, “Statistical analysis of network traffic for adaptive faults detection,” IEEE Transactions on Neural Networks, vol. 16, no. 5, pp. 1053–1063, Sep. 2005, doi:

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 7s
DOI: https://doi.org/10.17762/ijritcc.v11i7s.6976
***Article Received**: 28 March 2023 **Revised**: 14 May 2023 **Accepted**: 30 May 2023*

10.1109/TNN.2005.853414.

[10] M. Thottan, G. Liu, and C. Ji, "Anomaly Detection Approaches for Communication Networks," in Algorithms for Next Generation Networks, G. Cormode and M. Thottan, Eds. London: Springer, 2010, pp. 239–261. doi: 10.1007/978-1-84882-765- 3_11.

[11] Chaudhary, D. S. . (2021). ECG Signal Analysis for Myocardial Disease Prediction by Classification with Feature Extraction Machine Learning Architectures. Research Journal of Computer Systems and Engineering, 2(1), 06:10. Retrieved from https://technicaljournals.org/RJCSE/index.php/journal/article/view/12

[12] P.-N. Tan, M. Steinbach, and V. Kumar, Introduction to data mining. Boston: Pearson Addison Wesley, 2005.

[13] M. F. Augusteijn and B. A. Folkert, "Neural network classification and novelty detection," International Journal of Remote Sensing, vol. 23, no. 14, pp. 2891–2902, Jan. 2002, doi: 10.1080/01431160110055804.

[14] I. Diaz and J. Hollmen, "Residual generation and visualization for understanding novel process conditions," in Proceedings of the 2002 International Joint Conference on Neural Networks. IJCNN'02 (Cat. No.02CH37290), May 2002, vol. 3, pp. 2070– 2075 vol.3. doi: 10.1109/IJCNN.2002.1007460.

[15] O. Sharma, M. Girolami, and J. Sventek, "Detecting worm variants using machine learning," in Proceedings of the 2007 ACMCoNEXT conference, New York, NY, USA, Dec. 2007, pp. 1–12. doi: 10.1145/1364654.1364657.

[16] P. Soucy and G. W. Mineau, "A simple KNN algorithm for text categorization," in Proceedings 2001 IEEE International Conference on Data Mining, Nov. 2001, pp. 647–648. doi: 10.1109/ICDM.2001.989592.

[17] M. Beckmann, N. F. F. Ebecken, and B. S. L. P. de Lima, "A KNN Undersampling Approach for Data Balancing," Journal ofIntelligent Learning Systems and Applications, vol. 07, no. 04, Art. no. 04, 2015, doi: 10.4236/jilsa.2015.74010.

[18] S. Raschka, "Naive Bayes and Text Classification I - Introduction and Theory," arXiv:1410.5329 [cs], Feb. 2017, Accessed: Dec. 13, 2021. [Online]. Available: http://arxiv.org/abs/1410.5329

[19] A. A. Sebyala, T. Olukemi, L. Sacks, and D. L. Sacks, "Active Platform Security through Intrusion Detection Using Naive Bayesian Network For Anomaly Detection," 2002.

[20] Amjad Rehman Khan, Muhammad Kashif, Rutvij H. Jhaveri , Roshani Raut ,Tanzila Saba, and Saeed Ali Bahaj,"Deep Learning for Intrusion Detection and Security of Internet of Things (IoT): Current Analysis, Challenges, and Possible Solutions",Hindawi Security and Communication Networks Volume 2022, Article ID 4016073, 13 pages. Available: https://doi.org/10.1155/2022/4016073

[21] T. Saba, A. Rehman, T. Sadad, H. Kolivand, and S. A. Bahaj,"Anomaly-based intrusion detection system for IoT networksthrough deep learning model," Computers & Electrical Engineering, vol. 99, Article ID 107810, 2022.

[22] Y. Ali and H. Ullah Khan, "GTM approach towards engineering a features-oriented evaluation framework for secure authentication in IIoT environment," Computers & Industrial Engineering, vol. 168, Article ID 108119, 2022.

[23] A. Toprak, "Extreme Learning Machine (ELM)-Based Classification of Benign and Malignant Cells in Breast Cancer," Med Sci Monit, vol. 24, pp. 6537–6543, Sep. 2018, doi: 10.12659/MSM.910520.

[24] Anna, G., Jansen, M., Anna, J., Wagner, A., & Fischer, A. Machine Learning Applications for Quality Assurance in Engineering Education. Kuwait Journal of Machine Learning, 1(1). Retrieved from http://kuwaitjournals.com/index.php/kjml/article/view/109

[25] "Papers with Code - MQTT-IoT-IDS2020 Dataset." https://paperswithcode.com/dataset/mqtt-iot-ids2020 (accessed Apr. 25, 2022).

[26] H. Hindy, E. Bayne, M. Bures, R. Atkinson, C. Tachtatzis, and X. Bellekens, "Machine Learning Based IoT Intrusion Detection System: An MQTT Case Study (MQTT-IoT-IDS2020 Dataset)," arXiv:2006.15340 [cs], Nov. 2020, Accessed: Apr. 25, 2022. [Online]. Available: http://arxiv.org/abs/2006.15340

[27] Y. Al-Hamar, H. Kolivand, M. Tajdini, T. Saba, and V. Ramachandran, "Enterprise credential spear-phishing attack detection," Computers & Electrical Engineering, vol. 94, Article ID 107363, 2021.

[28] B. Liao, Y. Ali, S. Nazir, L. He, and H. U. Khan, "Security analysis of IoT devices by using mobile computing: a systematic literature review," IEEE Access, vol. 8, pp. 120331–120350, 2020.

[29] A. R. Khan, "Facial emotion recognition using conventional machine learning and deep learning methods: current achievements, analysis and remaining challenges," Information, vol. 13, no. 6, p. 268, 2022.

[30] Natalia Volkova, Machine Learning Approaches for Stock Market Prediction , Machine Learning Applications Conference Proceedings, Vol 2 2022.

[31] Verma, R. ., & Sharma, N. . (2023). Impact of Inclusion of Information and Communication Technologies in School Facilities and Effective Learning of Students in Green School. International Journal of Intelligent Systems and Applications in Engineering, 11(2s), 27–34. Retrieved from https://ijisae.org/index.php/IJISAE/article/view/2503

[32] A. Andalib and V. T. Vakili, "An autonomous intrusion detection system using an ensemble of advanced learners," in Proceedings of the 2020 28th Iranian Conference on Electrical Engineering (ICEE), pp. 1–5, IEEE, Tabriz, Iran, August 2020.

[33] H. Hindy, R. Atkinson, C. Tachtatzis, J. N. Colin, E. Bayne, and X. Bellekens, "Utilising deep learning techniques for effective zero-day attack detection," Electronics, vol. 9, no. 10, p. 1684, 2020.

[34] Vishwakarma M., Kesswani N.,"A two-stage intrusion detection system (TIDS) for Internet of Things", Advances in Deep Learning, Artificial Intelligence and Robotics, Springer (2022), pp. 89-97

[35] Anthi E., Williams L., Słowińska M., Theodorakopoulos G.,

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 7s
DOI: https://doi.org/10.17762/ijritcc.v11i7s.6976
Article Received: 28 March 2023 Revised: 14 May 2023 Accepted: 30 May 2023

Burnap P.,"A supervised intrusion detection system for smart home IoT devices", IEEE Internet Things J., 6 (5) (2019), pp. 9042-9053

[36] Seth S., Chahal K.K., Singh G.,"A novel ensemble framework for an intelligent intrusion detection system", IEEE Access, 9 (2021), pp. 138451-138467