Evaluation of Machine Learning Algorithms for Intrusion Detection System in WSN

Mohammed S. Alsahli[1], Marwah M. Almasri[2], Mousa Al-Akhras[3], Abdulaziz I. Al-Issa[4], Mohammed Alawairdhi[5]
College of Computing and Informatics, Saudi Electronic University, Riyadh 11673, KSA[1, 2, 3, 4, 5]
College of Computing and Informatics, Saudi Electronic University; King Abdullah II School of Information Technology
The University of Jordan, Riyadh 11673, KSA; Amman 11942, Jordan[3]

***Abstract*—Technology has revolutionized into connecting “things” together with the rebirth of the global network called Internet of Things (IoT). This is achieved through Wireless Sensor Network (WSN) which introduces new security challenges for Information Technology (IT) scientists and researchers. This paper addresses the security issues in WSN by establishing potential automated solutions for identifying associated risks. It also evaluates the effectiveness of various machine learning algorithms on two types of datasets, mainly, KDD99 and WSN datasets. The aim is to analyze and protect WSN networks in combination with Firewalls, Deep Packet Inspection (DPI), and Intrusion Prevention Systems (IPS) all specialized for the overall protection of WSN networks. Multiple testing options were investigated such as cross validation and percentage split. Based on the finding, the most accurate algorithm and the least time processing were suggested for both datasets.**

Keywords*—*Internet of Things (IoT); Wireless Sensor Network (WSN); Information Technology (IT); Denial of Service (DoS); Artificial Intelligence (AI); Machine Learning (ML)

I. Introduction

With the rapid expansion of technology, new threats and security issues arise, which become a hot area for research. Wireless Sensor Network (WSN) is composed of distributed wireless sensor nodes that collect raw data from the surrounding environment. Each Sensor node is equipped with a radio transceiver, a small microcontroller, and a power source [1]. These nodes are very small and have limited processing capabilities. They are designed based on low-cost and low-energy consumption that provide limited processing power and limited communication as represented in Fig. 1. Due to the sensors’ limitation in memory, processing power, and energy consumption, there are several potential security challenges inherently exist and should be properly addressed. The primary challenge is to protect the WSN without the availability of massive processing power and energy. Traditional security measures such as encryption is difficult to be implemented at the senor’s level due to its limited processing capabilities.

With the increased and sophisticated attack types on networks and applications, it is difficult to protect them against such attacks manually or by common Off-The-Shelf software such as firewalls, antivirus, Intrusion Detection System (IDS) or Intrusion Prevention System IPS). This makes artificial intelligence (AI) and machine learning (ML) algorithms popular and ultimately essential in such scenarios. AI in general and ML in specific can be used to protect WSN by identifying and classifying potential attacks by learning previously detected patterns of attacks.

Machine learning is becoming more popular in recent years. It enables machines or computers to work and react similar to what humans do. These systems improve with experience by learning the expected behavior. AI can be applied in many applications such as natural language processing and generation, speech recognition, virtual agent, machine learning, deep learning, biometrics, robotic process automation, text analytics and Neuro-Linguistic Programming (NLP), as well as in many domains such as healthcare, business, education, autonomous vehicles, robotics, government, and public safety and security. Moreover, AI becomes very useful in predictive analysis and plays a fundamental role in the software field and content creation.

This paper investigates different datasets with different machine learning algorithms, namely Naïve Bayes, improved Naïve Bayes, IBK, and Random Forest algorithms in multiple scenarios. The purpose is to identify the best method to mitigate the risks, threats, and security vulnerabilities associated with WSN networks.

The rest of this paper is organized as follows. Section II discusses related work. Section III presents the underlying concepts and proposed methodology. Section IV shows the experimental results. Section V discusses and analyzes the findings. Finally, section VI concludes the paper.

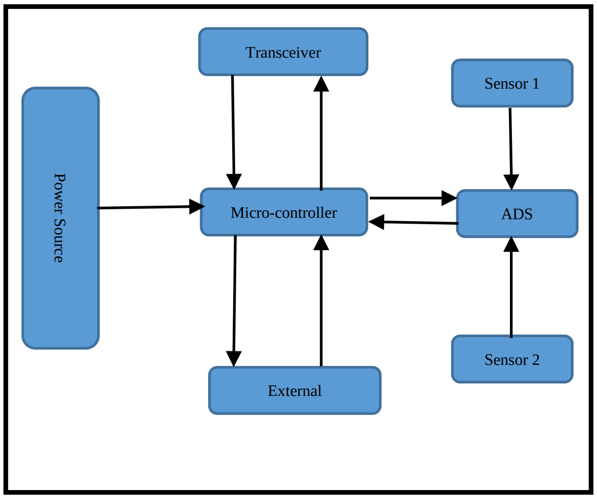

Fig. 1. WSN Mechanism.

II. Related Work

This section presents some researches about various attacks in WSN. In [2], authors have addressed Denial of Service (DoS) cyber-attacks on Wireless Sensor Networks (WSN) and how to mitigate these attacks. The researchers used specialized datasets for WSN constructed for classifying the types of attacks for their research. Four DoS attacks were considered: Flooding attack, Blackhole attack, Scheduling attack, and Gray-hole attack. The main purpose was to help WSN manufacturers to create and develop a system that detects and protects against DoS attacks in WSN. They have also discussed the challenges of protecting these networks due WSN limitations such as low processing, low power, and limited storages. They emphasized on the importance of mitigating and protecting against new and unprecedented attacks [2].

Moreover, the authors in [3] have focused on the classification's accuracy improvement of the Naïve Bayes algorithm, by finding more accurate probability estimation. This helps in solving the lack of the training data. Their approach was applied during the training phase without increasing the classification time. The first phase was building the classical Naïve Bayes classifier then fine-tune it in the second phase. Each training instance was classified, and if it is misclassified, it will contribute in fine tuning the probability value. Therefore, it will be correctly classified in the next round. Based on the findings, results showed an improved classification accuracy of many datasets.

Many researches have defined Wireless Sensor Network (WSN). It is typically composed of sensor nodes. These nodes gather data about the environment and send it back to the sink or the base station node. These data can be in different formats such as thermal, acoustic, optical, weather, pressure, chemical, and much more. It is extremely challenging task to develop an algorithm that is suitable for many applications scenarios in a diverse WSN environment; especially, considering data reliability and aggregation, localization, clustering, fault detection, and security [4].

Furthermore, the authors have highlighted the importance of utilizing ML in WSN for the following reasons [4]:

1) Using ML techniques could help in observing dynamic environments.

2) In some cases, WSN gathers new data in out-of-reach or threatening locations.

3) Accurate models are hard to be obtained in WSN since they are usually applied in sophisticated environments

4) Using ML techniques could be beneficial in extracting essential correlations.

The authors in [5] have emphasized the growing number of services that are providing facilities to humans which make using WSN valuable in many applications such as security systems, fire safety, various military applications, monitoring environmental conditions, and monitoring health condition. However, these WSNs encounter some weaknesses because of the nodes' exposure to various security attacks due to their limitations in power, processing, memory storage, bandwidth, data transmission via other nodes and multiple hops, its distributed nature, and self-organization. These attacks occurs at different levels of the OSI models. Therefore, it is important to build a security defense and monitoring system to protect against these attacks [5].

Similarly, the authors in [6] have discussed WSNs and their crucial role in different applications and usage; the vulnerabilities of the WSN due to their constrained resources. How DoS attack can be carried out at different layers of the network architecture. The authors focused specifically on the network layer because of the diversity of the attack at this layer. The authors reviewed many studies that use machine learning techniques pertaining to the network layer DoS attacks in WSN [6].

IDS and their important role in protecting against malicious attacks that affect the performance of the network have been addressed in [7]. The authors described Mobile Ad hoc networks (MANETs), WSN, and Internet of Things (IoT). The significance of the IDS and the need to protect such networks. Their proposed an IDS that has two stages. One that collects data using sniffers to generate correctly classified instances and in the second stage, a super node process data from different IDSs to differentiate benign from malicious nodes [7].

III. Underlying Concept and Methodology

This section presents the dataset types as well as the used machine learning techniques.

A. Datasets

A dataset is a collection of records that is gathered in a controlled lab environment. In this paper, two different datasets were used. The first dataset is called "KDDCup99 Dataset" which was derived from the DARPA 1998 dataset [8], [9]. It was selected and used to detect network breaches from a network security perspective. A network breach is the abuse of data and information to bypass the security rules and established regulations.

The authors in [10], have explained that the discovery of this interruption is a set of strategies and related activities that enable the progression of perceived methods for the identification of security classification. This dataset was provided by the archive, which was for a data mining competition held in aligning with KDD-99.

The author in [11] indicates that the features were to create a model that detects the bad connections or attacks as well as normal connections. The complete listing of the features defined for the connection records is listed in Table I.

TABLE I. DESCRIPTION OF KDDCUP99 DATASET FEATURES

	Feature Name	*Description*
1.	Duration	Number of seconds of the connection
2.	protocol_type	Type of the protocol, e.g., TCP, UDP, etc.
3.	Service	Network service on the destination, e.g., http, telnet, etc.
4.	Flag	Normal or error status of the connection
5.	src_bytes	Number of data bytes from source to destination
6.	dst_bytes	Number of data bytes from destination to source
7.	Land	1-connection is from/to the same host/port; 0-otherwise
8.	wrong_fragment	Number of ‘wrong’ fragments
9.	Urgent	Number of urgent packets
10.	Hot	The count of access to system directories, creation and execution of programs
11.	num_failed_logins	Number of failed login attempts
12.	logged_in	1 - successfully logged in; 0 otherwise
13.	num_compromised	Number of “compromised” conditions
14.	root_shell	1 - root shell is obtained; 0 otherwise
15.	su_attempted	1 – ‘su root’ command attempted; 0 – otherwise
16.	num_root	number of ‘root’ accesses
17.	num_file_creations	Number of file creation operations
18.	num_shells	Number of shell prompts
19.	num_access_files	Number of write, delete, and create operations on access control files
20.	num_outbound_cm ds	Number of outbound Commands in a ftp session
21.	is_hot_login	1 - the login belongs to the ‘hot’ list (e.g., root, adm, etc.) ; 0 – otherwise
22.	is_guest_login	1 - the login is a ‘guest’ login (e.g., guest, anonymous, etc.) ; 0 – otherwise
23.	Count	Number of connections to the same host as the current connection in the past 2 seconds
24.	srv_count	Number of connections to the same service as the current connection in the past 2 seconds
25.	serror_rate	% of connections that have ‘SYN’ errors to the same host
26.	srv_serror_rate	% of connections that have ‘SYN’ errors to the same service
27.	rerror_rate	% of connections that have ‘REJ’ errors to the same host
28.	srv_rerror_rate	% of connections that have ‘REJ’ errors to the same service
29.	same_srv_rate	% of connections to the same service and to the same host
30.	diff_srv_rate	% of connections to different services and to the same host
31.	srv_diff_host_rate	% of connections to the same service and to different hosts
32.	dst_host_count	Number of connections to the same host to the destination host as the current connection in the past 2 seconds
33.	dst_host_srv_count	Number of connections from the same service to the destination host as the current connection in the past 2 seconds
34.	dst_host_same_srv_rate	% of connections from the same service to the destination host
35.	dst_host_diff_srv_rate	% of connections from the different services to the destination host
36.	dst_host_same_src_port_rate	% of connections from the port services to the destination host
37.	dst_host_srv_diff_ host_rate	% of connections from the different hosts from the same service to destination host
38.	dst_host_serror_rate	% of connections that have ‘SYN’ errors to same host to the destination host
39.	dst_host_srv_ serror_rate	% of connections that have ‘SYN’ errors from same service to the destination host
40.	dst_host_rerror_rate	% of connections that have ‘REJ’ errors from the same host to the destination host
41.	dst_host_srv_ rerror_rate	% of connections that have ‘REJ’ errors from the same service to the destination host

The second used dataset is called “WSN Dataset” [12], which is specialized for WSN. It is used to detect different types of DoS attacks as well as normal behavior. The dataset is collected with different features and divided into different classes such as Blackhole, Grayhole, Scheduling, Flooding, and Normal. Low Energy Aware Cluster Hierarchy (LEACH) is the routing protocol that is used to collect the dataset that contains hundreds of thousands of records in WSN. It is designed to keep energy consumption low which is very important to maintain and improve the lifetime of WSN [13]. The problem or the limitation of LEACH is that it is only suitable for a small size WSN [13]. It assumes that all sensors can communicate with each other and with the sink (base station) as shown in Fig. 2. Table II represents the different WSN dataset attributes.

B. Machine Learning Techniques

Machine learning techniques are broadly categorized as unsupervised and supervised learning, which are for clustering, and classification/regression, respectively, as depicted in Fig. 3. Classification is a problem-solving technique for analyzing datasets or data models using algorithms such as Naïve and IBK. Regression is commonly used as a statistical tool to predict potential outcomes. The following subsections demonstrate various machine learning algorithms that were implemented on the above mentioned datasets.

1) Naïve Bayes: Naïve Bayes (NB) is a machine learning algorithm for AI software and computers. NB is based on mathematical calculation of probabilities that uses datasets (raw data or simple facts) to learn a concept. NB is used in a wide range of real applications and automated decision-making processes. A Naïve Bayes classifier is an algorithm that uses Bayes theorem features to classify objects. A NB is also known as simple Bayes or an independent Bayes. These classifiers use regular (or Naïve) independence intervals between the attributes of a data point.

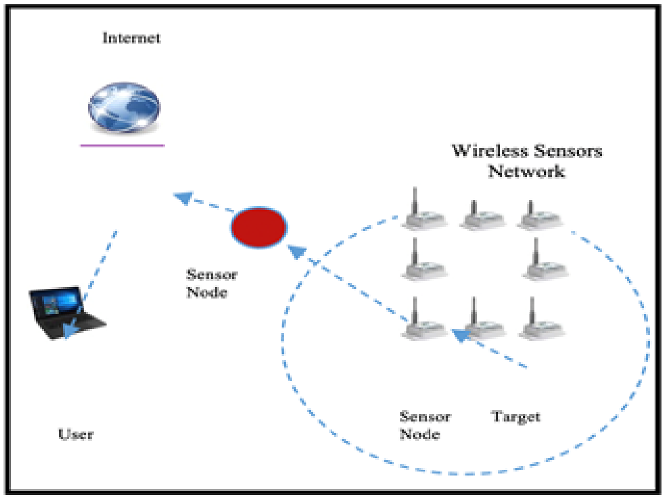

Fig. 2. WSN Network.

TABLE II. WSN-DS DATASET ATTRIBUTES [12]

#	*Attribute Name*	*Attribute Description*
1	Node ID	A unique ID to distinguish the sensor node in any round and at any stage
2	Time	The current simulation time of the node
3	Is CH	A flag to distinguish whether the node is CH or not
4	Who CH	The ID of the CH in the current round
5	Distance to CH	The distance between the node and its CH
6	Energy Consumption	The amount of energy consumed in the previous round
7	ADV_CH send	The number of advertise CH’s broadcast messages sent to the nodes
8	ADV_CH receives	The number of advertise CH messages received from CHs
9	Join_REQ send	The number of join request messages sent by the nodes to the CH
10	Join_REQ receives	The number of join request messages received by the CH from the nodes
11	ADV_SCH send	The number of advertise TDMA schedule broadcast messages sent to the nodes
12	ADV_SCH receives	The number of TDMA schedule messages received from CHs
13	Rank	The order of this node within the TDMA schedule
14	Data sent	The number of data packets sent from a sensor to its CH
15	Data Received	The number of data packets received from CH
16	Data sent to BS	The number of data packets sent to the BS
17	Distance CH to BS	The distance between the CH and the BS
18	Send Code	The cluster sending code
19	Attack Type	Type of the node. It is a class of five possible values, namely, Blackhole, Grayhole, Flooding, and Scheduling, in addition to normal, if the node is not an attacker

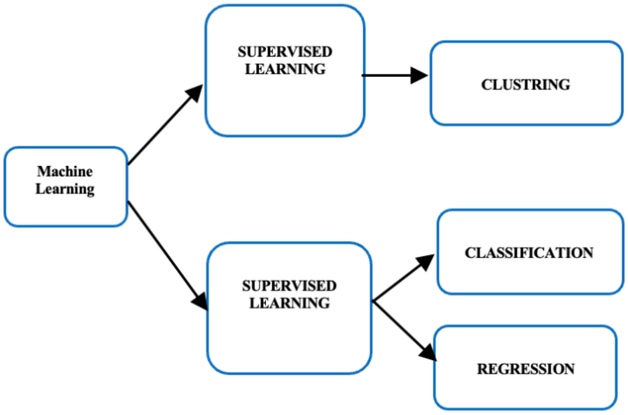

Fig. 3. Machine Learning.

The most common and widespread use of these Bayes algorithm is the use of spam filters or text and medical analysis. As these classifiers are easy to implement, they are most commonly used for machine learning. As stated by [14], Naïve Bayes classification uses probability theory to classify the data and makes use of Bayes theorem in its algorithm. The main feature of this classifier is that there can be an adjustment of the probability of an event as new data is introduced. It also assumes all the attributes that are in consideration are independent of each other. A Naïve Bayes classifier is not a single algorithm, but instead, it is a combination of specific machine learning algorithms in which statistical independence methods are used. A Naïve Bayes classifier makes a proper decision rule classification as long as the required class is more probable than any other present class. This fact is deemed accurate, as there is a slight inaccuracy in the probability estimation most of the times [3].

2) Fine Tune Naïve Bayes (FTNB): With respect to the Naïve Bayes classification, the tuning of parameters is limited, and it is recommended to improve the quality of the pre-processing and feature selection processes. The classifier performance and prediction can be improved by tuning and adjusting the classifier parameters, applying classifier combination techniques, or by monitoring the data fed to the classifier- either adding more data, refining existing one, or improving them [3].

3) Data Parsing (pre-processing): According to [15], the data is a string of raw text presented for each data point. A series of processes and steps convert this data into a structured vector such that the offset shows one feature and the value in the offset is correspondent to the frequency. Stemming, synonym finding and use of neutral words in the raw data text are one of the ways to improve the data parsing or the data processing methods.

a) Selection of Features: According to [16], the use cases for a Naïve Bayes classification like spam filtering are observed and utilized by showing how they fail or quickly can be improved. For assumption, an above average spam filter has a feature like a word frequency in all caps and words in titles or the occurrence of exclamation symbol in the title. The best feature for improvement is the use of long words or a group of more than a single word.

4) IBK algorithm: Instance Base Learner (IBK) algorithm is used in distance measure and classifying instances based on K-nearest neighbors to make predictions [17]. The computation in the test phase is very high and takes a long time, especially for a huge number or instances in the dataset. The default value of neighbors is 1. Sometimes called 1-NN [18].

5) Random forest algorithm: Random Forest or random decision forest algorithm is used for classification and regression of an ensemble of the collection of datasets. In WEKA program, Random Forest can only do the classification part, not the regression task. It operates by building a great number of decision trees in the training phase and perform the classification task. In WEKA, there is no output of the mean prediction or regression of each tree. Random Forest classification mean mapping input data in the dataset or instances to a category. This is also called categorization of the instances. The algorithm that does the classification, especially in the concrete implementation, is called the classifier [19].

IV. Experiments and Result

This section discusses and demonstrates the experiments conducted and their results. Both datasets have been classified using the above-mentioned machine learning algorithms (section III-B) using Cross-validation and percentage split techniques. Cross validation is a standard analysis tool used to verify the validity of the data mining model. It works by dividing the dataset into a number of folds or pieces and hold each fold in turn for testing and training all of the other pieces in the system. In dividing the dataset into layers or folds, it ensures that each layer or fold had the correct portion of class values [20]. Additionally, Percentage split determines the percentage used for training the system [20]. For our experiments, 66% was used for training and 34% was used for testing. The following subsections demonstrate the results obtained by each algorithm conducted on both datasets using cross-validation and percentage split techniques.

A. Naïve Bayes (NB) Algorithm

1) Cross-validation technique: Table III shows the results of running NB algorithm on both datasets (KDDCUP99 and WSN-DS) using cross-validation technique. Table IV demonstrates the weighted average accuracy using cross-validation technique in terms of several factors such as:

- True Positive Rate (TP): the rate that the system or an algorithm correctly classifies an instance as a positive class.
- True Negative Rate (TN): the rate that the system or an algorithm correctly classifies an instance as a negative class.
- False Positive Rate (FP): the rate that the system or an algorithm falsely (wrongly) classifies an instance as a positive class/.
- False Negative Rate (FN): the rate that the system or an algorithm falsely (wrongly) classifies an instance as a negative class.
- Precision: the ratio of correctly classified instances as positive to the instances that are classified by the algorithm as positive.
- Recall: the ratio of correctly classified instances as positive to the positive instances (whether classified correctly or not).
- Receiver Operating Characteristics (ROC): is a technique used as graph or curve to represent or visualize the performance of the classifiers. It is widely used in machine learning, data mining, and decision making. Also, it is used as a method of comparing diagnostic tests.

2) Percentage split technique: In this experiment, 66% of the data was used for training and 34% for testing. Table V shows the results of running Naïve Bayes (NB) algorithm on both datasets (KDDCUP99 and WSN-DS) using the percentage split technique. In addition, Table VI demonstrates the weighted accuracy average using the percentage split technique in terms of TP, TN, Precision, and ROC.

B. IBK Algorithm

1) Cross-validation technique: Table VII shows the results of running IBK algorithm on both datasets (KDDCUP99 and WSN-DS) using the cross- validation technique. Table VIII demonstrates the weighted accuracy average using the cross-validation technique.

2) Percentage split technique: In this experiment, 66% of the data was used for training and 34% for testing. Table IX shows the results of running IBK algorithm on both datasets (KDDCUP99 and WSN-DS) using the percentage split technique. Table X demonstrates the weighted accuracy average using the percentage split technique in terms of several factors.

C. Random Forest Algorithm

1) Cross-validation technique: Table XI shows the results of running the Random Forest algorithm on both datasets (KDDCUP99 and WSN-DS) using the cross-validation technique. Table XII demonstrates the weighted accuracy average using the cross-validation technique.

2) Percentage split technique: In this experiment, 66% of the data was used for training and 34% for testing. Table XIII shows the results of running IBK algorithm on both datasets (KDDCUP99 and WSN-DS) using the percentage split technique. Table XIV demonstrates the weighted accuracy average using the percentage split technique in terms of several factors.

TABLE III. THE RESULTS OF NAÏVE BAYES (NB) ALGORITHM USING THE CROSS-VALIDATION TECHNIQUE

	Dataset		
	KDDCUP99		*WSN-DS*
Correctly Classified Instances	459019	92.9151 %	459019
Incorrectly Classified Instances	35001	7.0849 %	35001
Kappa statistic	0.8828		0.8828
Mean absolute error	0.0061		0.0061
Root mean squared error	0.0765		0.0765
Relative absolute error	11.955 %		11.955 %
Root relative squared error	47.6941 %		47.6941 %
Total Number of Instances	494020		494020

TABLE IV. THE WEIGHTED ACCURACY AVERAGE OF NAÏVE BAYES (NB) ALGORITHM USING THE CROSS-VALIDATION TECHNIQUE

Weighted Avg. of	*TP Rate*	*FP Rate*	*Precision*	*Recall*	*F-Measure*	*MCC*	*ROC Area*	*PRC Area*
KDDCup99	0.929	0.000	0.989	0.929	0.951	0.948	1.000	0.991
WSN Dataset	0.954	0.012	0.966	0.954	0.957	0.847	0.980	0.971

TABLE V. THE RESULTS OF NAÏVE BAYES (NB) ALGORITHM USING THE PERCENTAGE SPLIT TECHNIQUE

	Dataset		
	WSN		*KDDCUP99*
Correctly Classified Instances	121606	95.4634%	121606
Incorrectly Classified Instances	5779	4.5366 %	5779
Kappa statistic	0.7678		0.7678
Mean absolute error	0.0182		0.0182
Root mean squared error	0.1324		0.1324
Relative absolute error	26.2165 %		26.2165 %
Root relative squared error	71.0237 %		71.0237 %
Total Number of Instances	127385		127385

TABLE VI. The Weighted Accuracy Average of Naïve Bayes (NB) Algorithm using the Percentage Split Technique

Weighted Avg. of	*TP Rate*	*FP Rate*	*Precision*	*Recall*	*F-Measure*	*MCC*	*ROC Area*	*PRC Area*
KDDCup99	0.930	0.000	NA	0.930	NA	NA	1.000	0.991
WSN Dataset	0.955	0.011	0.967	0.955	0.958	0.851	0.981	0.972

TABLE VII. The Results of IBK Algorithm using the Cross-validation Technique

	Dataset		
	KDDCUP99		*WSN*
Correctly Classified Instances	493796	99.9547 %	493796
Incorrectly Classified Instances	224	0.0453 %	224
Kappa statistic	0.9992		0.9992
Mean absolute error	0		0
Root mean squared error	0.0063		0.0063
Relative absolute error	0.0791 %		0.0791 %
Root relative squared error	3.9104 %		3.9104 %
Total Number of Instances	494020		494020

TABLE VIII. The Weighted Accuracy Average of IBK Algorithm using the Cross-validation Technique

Weighted Avg. of	*TP Rate*	*FP Rate*	*Precision*	*Recall*	*F-Measure*	*MCC*	*ROC Area*	*PRC Area*
KDDCup99	1.000	0.000	1.000	1.000	1.000	0.999	1.000	0.999
WSN Dataset	0.994	0.025	0.994	0.994	0.994	0.970	0.985	0.992

TABLE IX. The Results of IBK Algorithm using the Percentage Split Technique

	Dataset		
	KDDCUP99		**WSN**
Correctly Classified Instances	167869	99.9417 %	167869
Incorrectly Classified Instances	98	0.0583 %	98
Kappa statistic	0.999		0.999
Mean absolute error	0.0001		0.0001
Root mean squared error	0.0071		0.0071
Relative absolute error	0.1024 %		0.1024 %
Root relative squared error	4.4419 %		4.4419 %
Total Number of Instances	167967		167967

TABLE X. The Weighted Accuracy Average of IBK Algorithm using the Percentage Split Technique

Weighted Avg. of	*TP Rate*	*FP Rate*	*Precision*	*Recall*	*F-Measure*	*MCC*	*ROC Area*	*PRC Area*
KDDCup99	1.000	0.000	1.000	1.000	1.000	0.999	1.000	0.999
WSN Dataset	0.994	0.025	0.994	0.994	0.994	0.970	0.985	0.992

TABLE XI. The Results of Random Forest Algorithm using the Cross-validation Technique

	Dataset		
	KDDCUP99		*WSN*
Correctly Classified Instances	493915	99.9787 %	167869
Incorrectly Classified Instances	105	0.0213 %	98
Kappa statistic	0.9996		0.999
Mean absolute error	0.0001		0.0001
Root mean squared error	0.004		0.0071
Relative absolute error	0.1064 %		0.1024 %
Root relative squared error	2.5242 %		4.4419 %
Total Number of Instances	494020		167967

TABLE XII. THE WEIGHTED ACCURACY AVERAGE OF RANDOM FOREST ALGORITHM USING THE CROSS-VALIDATION TECHNIQUE

Weighted Avg. of	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area
KDDCup99	1.000	0.000	NA	1.000	NA	NA	1.000	1.000
WSN Dataset	0.997	0.016	0.997	0.997	0.997	0.985	0.997	0.999

TABLE XIII. THE RESULTS OF RANDOM FOREST ALGORITHM USING THE PERCENTAGE SPLIT TECHNIQUE

	Dataset		
	KDDCUP99		*WSN*
Correctly Classified Instances	167915	99.969 %	167915
Incorrectly Classified Instances	52	0.031 %	52
Kappa statistic	0.9995		0.9995
Mean absolute error	0.0001		0.0001
Root mean squared error	0.0046		0.0046
Relative absolute error	0.1225 %		0.1225 %
Root relative squared error	2.8772 %		2.8772 %
Total Number of Instances	167967		167967

TABLE XIV. THE WEIGHTED ACCURACY AVERAGE OF RANDOM FOREST ALGORITHM USING THE PERCENTAGE SPLIT TECHNIQUE

Weighted Avg. of	*TP Rate*	*FP Rate*	*Precision*	*Recall*	*F-Measure*	*MCC*	*ROC Area*	*PRC Area*
KDDCup99	1.000	0.000	NA	1.000	NA	NA	1.000	1.000
WSN Dataset	0.997	0.015	0.997	0.997	0.997	0.985	0.997	0.999

V. DISCUSSION AND PERFORMANCE EVALUATION

In this section, all results will be discussed and analyzed. Overall performance evaluation will be presented as well. Naïve Bayes algorithm, using the cross-validation technique, has classified most of the instances correctly on both datasets. TP in KDDCup99 is about 92.9% and in WSN-DS is 95.3%. The errors or incorrectly classified instances were 7.08 and 4.064, respectively. Therefore, NAÏVE algorithm is more accurate with WSN dataset than KDDCup99 dataset. Moreover, the weighted accuracy average of both datasets is very similar. Using the percentage split technique with the former algorithm on both datasets showed more accurate results as compared with the cross-validation.

Moreover, IBK algorithm was run on both datasets using cross-validation. Both processes took no time at all, less than one second. As can be seen from the results of the correctly classified instances, both datasets were very close even though the number of instances in each dataset are not the same. The TP in KDDCUP99-DS is about (100%) and in WSN-DS is (99.4%). The errors or incorrectly classified instances were (0.552%) in WSN-DS and (0.0453%) in KDDCup99-DS. Whereas the correctly classified instances in WSN-DS is (99.4%) and in KDDCup99-DS is (99.9%) which is an excellent accuracy in both datasets, almost (100%). This is also reflecting on the weighted average of both datasets against the IBK algorithm. The numbers are very similar, almost the same (100%).

With the percentage split, using IBK algorithm was very accurate with KDDCup99 and WSN datasets. The errors or incorrectly classified instances were (0.584%) in WSN-DS and (0.058%) in KDDCup99-DS. And the correctly classified instances in WSN-DS is (99.4%) and in KDDCup99-DS is (99.9%) which is an excellent accuracy in both datasets, almost (100%). To sum up, IBK using percentage split test algorithm is very accurate with KDDCup99 dataset and with WSN dataset compared with the cross validation.

Furthermore, the Random Forest algorithm has been run on both datasets using the cross validation and percentage split options. With the cross validation, the TP in KDDCUP99-DS is about (100%) and in WSN-DS is (99.7%). The errors or incorrectly classified instances were (0.2779%) in WSN and (0.0213%) in KDDCup99. Also, the correctly classified instances in WSN-DS is (99.7%) and in KDDCup99-DS is (99.9%) which is an excellent accuracy in both datasets, almost (100%). For the percentage split, both datasets took few seconds to process (6.24 and 8.45 respectively). The TP in KDDCup99-DS is about (100%) and in WSN-DS is (99.7%). The errors or incorrectly classified instances were (0.2724%) in WSN-DS and (0.031%) in KDDCup99-DS. The correctly classified instances in WSN-DS is (99.7%) and in KDDCup99 is (99.9%) which is an excellent accuracy in both datasets, almost (100%). It can be concluded that Random Forest using percentage split test algorithm is very accurate with KDDCup99 dataset and with WSN dataset.

As an overall performance evaluation among all algorithms and test options for KDDCup99 dataset, the NAÏVE Bayes algorithm with cross-validation test option is the least accurate (92.92%), meaning it has the least correctly classified instances. On the other hand, the Random Forest algorithm with cross-validation test option (99.98%) was the most accurate. Similarly, for WSN dataset, the NAÏVE Bayes algorithm with cross-validation test option is the least accurate results (95.35%), meaning it has the least correctly classified instances and the Random Forest algorithm with cross-validation test option is the most accurate one (99.73%).

Moreover, the accuracy and processing time were recorded for both datasets using all test options as shown in Fig. 4, 5, 6 and 7. The least time taken was using the IBK algorithm using percentage split test option on WSN dataset (0.05) seconds, then with the KDDCup99 dataset algorithm using percentage split test option (0.08) seconds. As for accuracy measurement, the Random Forest algorithm is the most accurate algorithm in both datasets with all test options. The highest accuracy was registered using cross validation on KDDCup99 dataset (99.9787 %), then on WSN dataset (99.7276 %) using the percentage split test option as shown in Fig. 4 and Fig. 6, respectively.

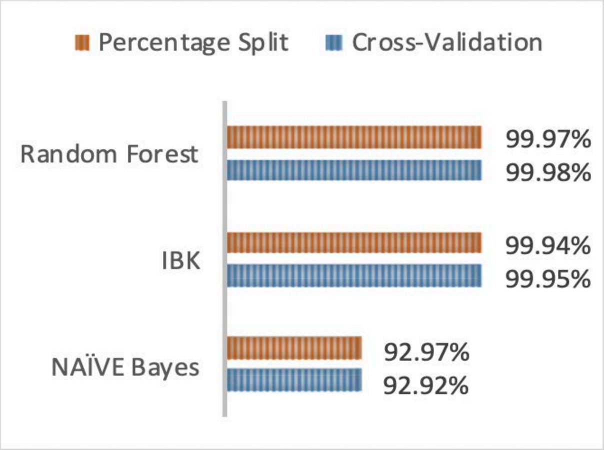

Fig. 4. Comparison of Accuracy on KDDCup99 Dataset.

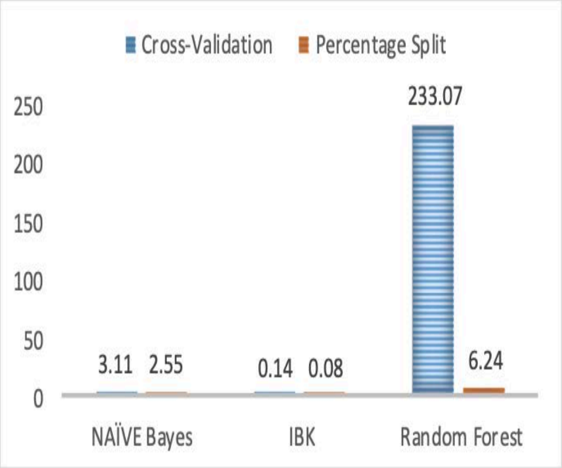

Fig. 5. Comparison of the Processing Time on KDDCup99 Dataset.

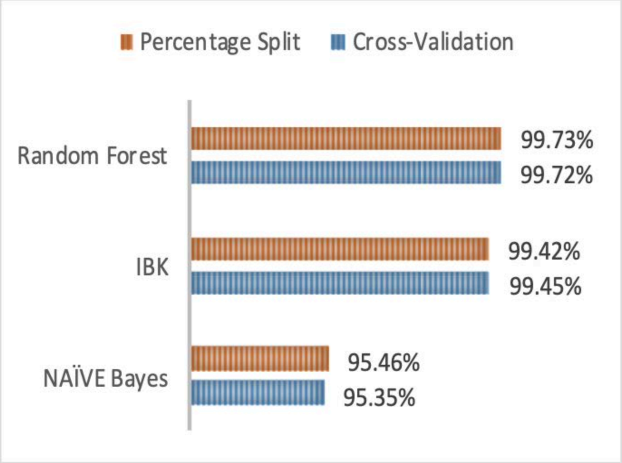

Fig. 6. Comparison of Accuracy on WSN Dataset.

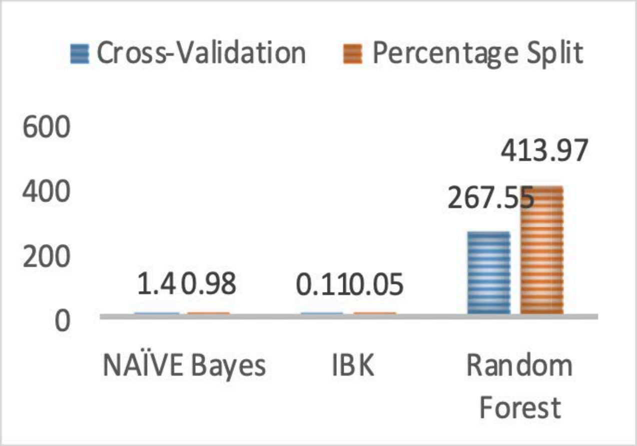

Fig. 7. Comparison of the Processing Time on WSN Dataset.

VI. Conclusion

Due to the importance of protecting WSN against rogue entities of hackers and intruders, taking into considerations all constraints such as limited power, storage, and processing capabilities, a model/dataset needs to be trained to mitigate new or modified attack types in networks.

This paper has analyzed and compared different machine learning algorithms against two datasets (WSN and KDD99) using WEKA tool. The purpose was to further assist in analyzing and protecting WSN networks in combination with Firewalls, Deep Packet Inspection (DPI), and Intrusion Prevention Systems (IPS) that are specialized in protecting WSN networks. Multiple testing options were investigated such as cross validation and percentage split. Based on the finding, the most accurate algorithm and the least time consuming were suggested for both datasets. Future research is needed to create more datasets to characterize various types of attacks in the wireless sensor networks.

References

[1] P. Kurer, D. M, and H. S. Guruprasad, “Energy Aware Dynamic Clustering and Hierarchical Route based on LEACH for WSN,” International Journal of Computer Networking Wireless and Mobile Communications, vol. 3, no. 3, pp. 79-86, 2013.

[2] A. I. Al-issa, M. Al-Akhras, M. ALsahli, and M. Alawairdhi, “*Using Machine Learning to Detect DoS Attacks in Wireless Sensor Networks,”* Paper presented at the 2019 IEEE Jordan International Joint Conference on Electrical Engineering and Information Technology (JEEIT), April 2019.

[3] K. Hindi, “Fine tuning the Naïve Bayesian learning algorithm,” *AI Communications, vol.* 27, no. 2, pp. 133-141, 2014. doi: 10.3233/AIC-130588.

[4] M. A. Alsheikh, S. Lin, D. Niyato, and H. Tan, “Machine Learning in Wireless Sensor Networks: Algorithms, Strategies, and Applications,” *IEEE Communications Surveys & Tutorials, vol. 16*, no. 4, pp. 1996-2018, 2014. doi: 10.1109/COMST.2014.2320099.

[5] B. Ashwini, S. Abhale, and S. Manivannan, “Supervised Machine Learning Classification Algorithmic Approach for Finding Anomaly Type of Intrusion Detection in Wireless Sensor Network,” *Optical Memory and Neural Network*s, vol. 29, no. 3, pp. 244-256, 2020. Available: https://link.springer.com/article/10.3103/S1060992X20030029#citeas.

[6] S. Gunduz, B., Arslan, and M. Demirci, “*A Review of Machine Learning Solutions to Denial-of-Services Attacks in Wireless Sensor Networks*,” 2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA), 2015. Available: https://ieeexplore.ieee.org/document/7424301/authors#authors.

[7] A. Amouri, V. T. Alaparthy, and S. D. Morgera, “A Machine Learning Based Intrusion Detection System for Mobile Internet of Things,” *Sensors (Basel),* vol. 20, no. 2, 2020. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/31947567/.

[8] A. M. Al Tobi and I. Duncan, “KDD 1999 generation faults: a review and analysis,” *Journal of Cyber Security Technology*, vol. 2, no. 3-4, pp. 164-200., 2018. doi: 10.1080/23742917.2018.1518061.

[9] E. Kabir, J. Hu, H. Wang, and G. Zhuo, “A novel statistical technique for intrusion detection systems,” *Future Generation Computer Systems, vol.* 79, no. 1, pp. 303-318, 2018. doi: https://doi.org/10.1016/j.future.2017.01.029.

[10] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani,, “A detailed analysis of the KDD CUP 99 data set,” Paper presented at the 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, 2009.

[11] C. Elkan, “Results of the KDD'99 classifier learning,” *SIGKDD Explor. Newsl., vol. 1*, no. 2, pp. 63–64, 2000. doi: 10.1145/846183.846199.

[12] I. Almomani, B. Al-Kasasbeh, and M. Al-Akhras, “WSN-DS: A Dataset for Intrusion Detection Systems in Wireless Sensor Networks.,” *Journal of Sensors*, 2016. doi: 10.1155/2016/4731953.

[13] I. Almomani and B. Al-Kasasbeh, “*Performance analysis of LEACH protocol under Denial of Service attacks,”* Paper presented at the 2015 6th International Conference on Information and Communication Systems (ICICS), 2015.

[14] S. L. Ting, W. H. Ip, and A. Tsang, “Is Naïve Bayes a Good Classifier for Document Classification?” *International Journal of Software Engineering and its Applications*, *vol.* 5, no. 3, pp. 37-46., 2011.

[15] D. Meretakis and B. Wüthrich, “*Extending naïve Bayes classifiers using long itemsets,”* Paper presented at the KDD '99, 1999.

[16] I. Androutsopoulos, J. Koutsias, K. Chandrinos, and C. Spyropoulos, “*An Experimental Comparison of Naive Bayesian and Keyword-Based Anti-Spam Filtering with Personal E-mail Messages*,” Proceedings of the 23rd annual international ACM SIGIR conference on Research and development, 2000. doi: 10.1145/345508.345569.

[17] K. El-Hindi and M. Al-Akhras, “Smoothing Decision Boundaries to Avoid Overfitting in Neural Network Training,” *Neural Network World, vol. 21*, no. 4, pp. 311-325, 2011.

[18] I. I. Baskin, G. Marcou, D. Horvath, and A. Varnek, “*Classification Models,” Tutorials in Chemoinformatics*, Varnek, A. (Ed.), 2017.

[19] F. Syeda, M. A. B. Mirza, A. Baig, and M. Pawar, “Performance Evaluation of Different Data Mining Classification Algorithm and Predictive Analysis,” *IOSR Journal of Computer Engineering, vol.* 10, no. 6, pp. 1-6, 2013. doi: 10.9790/0661-1060106.

[20] T. Borovicka, M. Jirina, and P. Kordík, “Selecting Representative Data Sets,” *Advances in data mining knowledge discovery and applications*, pp. 43-70, 2012.