DOI: 10.15849/IJASCA.230720.05

Int. J. Advance Soft Compu. Appl, Vol. 15, No. 2, July 2023
Print ISSN: 2710-1274, Online ISSN: 2074-8523
Copyright © Al-Zaytoonah University of Jordan (ZUJ)

An Enhanced Intrusion Detection System for Protecting HTTP Services from Attacks

Hani Al-Mimi[1], Nesreen A. Hamad[2], Mosleh M. Abualhaj[3], Mohammad Sh. Daoud[4], Ali Al-dahoud[5], Mohammad Rasmi[6]

[1,2,5]Faculty of Science and Information Technology, Al-Zaytooanh University of Jordan, Amman, 11733, Jordan

[3]Department of Networks and Cybersecurity, Al-Ahliyya Amman University, Amman, 19328, Jordan

[4]College of Engineering, Al Ain University, Abu Dhabi, United Arab Emirates

[6]Department of Cyber Security, Zarqa University, Zarqa, 13110, Jordan

[1]Corresponding Author: Hani Al-Mimi. Email: hani.mimi@zuj.edu.jo

Abstract

Cybercriminals constantly develop new sophisticated methods to breach the security of their targets, thereby increasing cyberattack cases. HTTP is one of the earliest and most susceptible network services. However, it has several security flaws that have been exploited repeatedly throughout time. Constructing a robust intrusion detection system that prevents unauthorized access to network resources is crucial to recognize HTTP attacks and protecting data. Several novel ways have been recently proposed as a panacea for intrusion detection. However, building a secure HTTP system remains challenging because attackers continuously adapt their strategies to circumvent the system's security features. The current study employed machine learning (ML) classifiers to build a model that categorizes the HTTP traffic as an attack or normal traffic. The proposed model is called ML-HTTP. The UNSW_NB15 dataset was utilized to evaluate the performance of the ML-HTTP model. Among several classifiers investigated with the ML-HTTP model, both random forest and logistic regression classifiers achieved the highest accuracy (96.32%), recall (97.01%), precision (97.63%), and Matthew's correlation coefficient (91.86%).

Keywords: *UNSW_NB15 dataset; HTTP attacks; intrusion detection system; machine learning.*

1 Introduction

HTTP is the primary communication protocol used by web applications, and the majority of Internet traffic is transmitted via HTTP. While many individuals believe that their web browser connects them to everything online, data centers also encounter a significant amount of HTTP traffic, and many businesses generate substantial revenue through online sales. However, as HTTP's popularity has grown, so have the associated risks.

Received 25 January 2023; Accepted 23 May 2023

Like any protocol, HTTP is vulnerable to attacks [1][2][3][4]. In fact, according to [5], 46% of all online web apps contain serious vulnerabilities, and a concerning 87% include medium vulnerabilities. Furthermore, dangerous web cyberattacks increased by 300% in 2021 alone. Common web and HTTP threats include SYN flood, cache bypassing, cross-site scripting, and SQL injections. These vulnerabilities can pose significant risks to online applications and potentially compromise web servers. If not adequately addressed, they can result in a loss of client trust and ultimately lead to a loss of business [5][6][7][8][9]. Numerous options exist for protecting web apps from the aforementioned threats, and these systems are often effective against known attack patterns. However, identifying new forms of attacks requires exploring new attack patterns. In the present study, we utilized machine learning (ML) techniques to identify new HTTP protocol threats.

ML has proven to be a valuable tool in enhancing the detection of HTTP service threats. Intelligent security technologies, such as Intrusion Detection and Prevention Systems (IDPS), utilize ML approaches to prevent HTTP service threats. IDPS continuously monitors and analyzes system events to detect any illegal attempts to access the system's resources in real-time or near-real-time and informs administrators accordingly. In this study, we employed ML approaches to develop a self-learning model capable of identifying new HTTP threats [10][11][12]. Specifically, we used supervised ML approaches in the proposed model. Like the algorithms used in IDPS, these supervised ML approaches operate on organized and labeled data. The proposed model, ML-HTTP, implements several supervised ML algorithms to determine the best approach for detecting HTTP system attacks. In summary, this work makes several contributions. Firstly, it conducts a thorough analysis of recent research on detecting and mitigating attacks on HTTP services. Secondly, it presents a self-learning machine learning model that is designed to detect new attacks on HTTP services and is capable of handling various types of attacks on HTTP. Finally, the proposed model achieves impressive results in terms of accuracy, recall, precision, and Matthews correlation coefficients.

The remainder of this article is organized as follows: Section 2 covers some of the topics that aid readers in understanding the proposed method. These topics include the UNSW_NB15 dataset, supervised ML techniques, Min-Max scaler methodology, and K-fold cross-validation strategy. Section 3 summarizes the methods recommended in the literature. In Section 4, the proposed technique is presented. Section 5 details the implementation environment. The metrics used to evaluate the proposed model are elaborated in Section 6. Section 7 presents the results of the proposed model. Section 8 concludes the paper.

2 Background

This section elaborates on some of the topics that make the study understandable for the reader. These topics include UNSW_NB15 dataset, supervised ML algorithms, the Min-Max scaler technique, and the K-fold cross-validation technique.

2.1 UNSW_NB15 Dataset

The UNSW-NB15 dataset focuses on network intrusions. The Ixia PerfectStorm tool, housed in the cyber range lab of the Australian Centre for Cyber Security, is responsible for generating the raw network packets that comprise the UNSW-NB 15 dataset. These packets are used to generate a hybrid of real modern normal activities and synthetic contemporary attack behaviors. The tcpdump utility is used to record 100 GB worth of the raw data (e.g., pcap files). This data collection contains nine distinct types of attacks: Fuzzers, Analysis, Backdoors, DoS, Exploits, Generics, Reconnaissance, Shellcode, and Worms. The Argus and Bro-IDS tools are utilized to create 49 characteristics together with the class label. A total of 2,540,044 records are kept in four CSV files: UNSW_NB15 1.csv, UNS_WNB15 2.csv, UNSW_NB15 3.csv, and UNSW_NB15 4.csv. The UNSW NB15 training-set.csv file and the UNSW NB15 testing-set file are created from a partition of this dataset with only 41 features each. As the name implies, the UNSW NB15 training-set.csv and UNSW NB15 testing-set files are used for training and testing, respectively. The UNSW NB15 training-set.csv has 175,341 records, but the UNSW NB15 testing-set only has 82,332 records. These records are from various attacks and normal traffic and dispersed throughout a wide variety of services, including SMTP, DHCP, RADIUS, HTTP, IRC, SNMP, DNS, SSH, FTP, SSL, and POP3 (some services are unknown) [13] [14] [15].

The present study is interested only in the attacks on the HTTP service. Therefore, the HTTP service records are extracted from the UNSW-NB15 dataset (for training and testing datasets). Moreover, all other records of the other services are removed. Thus, new subdatasets called HTTP_UNSW-NB15 training and HTTP-UNSW-NB15 testing are produced. These two datasets (training and testing datasets) are combined as one dataset for training and testing. This combination dataset has 27011 records of attacks and normal traffic. The produced HTTP_UNSW-NB15 dataset is used in the present paper. The attack traffic in HTTP_UNSW_NB15 contains eight types: Fuzzers, DoS, Exploits, Generic, Reconnaissance, Analysis, Backdoor, and Worms. Table 1 displays the number of records of each attack type.

Table 1: Number of records for each attack in the HTTP_UNSW_NB15 dataset

Attack Type	**Number of records**
Analysis	558
Backdoor	92
DoS	1709
Exploits	11481
Fuzzers	1087
Generic	502
Reconnaissance	2073
Worms	148
Normal (No Attack)	9361
Total	27011

2.2 ML Techniques

This section discusses the supervised ML methods used in this article. These methods are Naive Bayes (NB), k-nearest neighbors (K-NN), decision tree (DT), support vector machine (SVM), logistic regression (LR), and random forest (RF).

2.2.1 NB Method

NB is a prime example of the principle that the simplest solutions are typically the most effective. Despite the advancements in ML over the past decade, NB has shown to be not only easy but also quick, accurate, and dependable. This method is a probabilistic ML technique based on Bayes' theorem employed for various classification problems. Bayes' theorem is a simple formula (see Eq. (1)) used to calculate conditional probabilities. Conditional probability is the chance of an event occurring, given that another event has occurred. It informs us the following: how frequently A1 occurs when B1 occurs, written as P(A1|B1); how often B1 occurs when A1 occurs, written as P(B1|A1); how probable A1 is on its own, written as P(A1); how probable B1 is on its own, written as P(B1). In particular, Bayes' theorem is a method for determining a probability when we know other probabilities [16] [17].

$$P(A1|B1) = \frac{P(A1|B1)P(A1)}{P(B1)} \quad (1)$$

2.2.2 K-NN Method

K-NN is a widely used and simple classification method that categorizes new data points based on the similarity of their neighbors. This approach produces a competitive outcome as the algorithm calculates the distances between a specific data point and the K closest data points in the dataset, followed by a vote for the category with the highest frequency. The Euclidean distance, as shown in Eq. (2), is the most common distance measurement used in K-NN. The final model is represented by labeled data arranged in space, and the method is frequently applied in genetics and forecasting. K-NN is known for its ability to reduce overfitting, but it requires selecting an optimal value for K, usually determined by the square root of the number of samples in the dataset [16][18].

$$\text{Euclidean Distance between M and K} = \sqrt{(I_2 - I_1)^2 + (J_2 - J_1)^2} \quad (2)$$

2.2.3 DT Method

DTs are a type of supervised ML algorithms that uses a set of rules to make predictions. Similar to a flowchart, DTs are structured as a tree with each internal node representing a feature, each branch representing a decision rule, and each leaf node reflecting the conclusion. The root node is at the top of the tree, and the tree is recursively partitioned to create a set of simple decision rules that can be used to predict the class or value of the target variable based on past data. When a new record is presented to the DT, its class label is predicted by following the tree's branches until reaching a leaf node. DTs are particularly useful when the relationships between input and output variables are complex and nonlinear. However, overfitting can be a concern, and the DT's performance may depend on the selection of the tree's depth and splitting criteria [16] [19].

2.2.4 SVM Method

To add to what you said, SVM can handle both linear and non-linear classification problems through the use of different kernel functions. Some common kernel functions include linear, polynomial, and radial basis function kernels. SVM is also known for being effective in handling high-dimensional data, such as in image classification tasks. One of the potential drawbacks of SVM is that it can be sensitive to the choice of kernel function and hyperparameters, which may require careful tuning to achieve optimal performance [16] [17].

2.2.5 LR Method

LR is a popular ML algorithm that is used to model the dependent variable using a logistic function. This method is particularly useful for binary data analysis, where the dependent variable has only two possible classifications, such as whether a cancer is malignant or not. In LR, the Sigmoid function is employed to return the likelihood of a particular label. This function converts expected values to probabilities and can map any real value between 0 and 1 to another value. The binomial variant of LR is suitable when the target variable has only two potential values, and it can be used to predict whether HTTP traffic is an attack or not [16] [20].

2.2.5 RF Method

RF is a popular machine learning method that belongs to the supervised learning technique. It is based on ensemble learning, which is the process of integrating multiple classifiers to solve a complicated issue and improve the model's performance. RF is known to require less training time than other algorithms while accurately predicting the output. RF is a type of classifier that consists of multiple DTs on various subsets of the provided dataset. The average result of these many DTs is used to increase the prediction accuracy of the dataset. Unlike depending on a single DT, RF considers the forecast from each tree and predicts the final output based on the majority vote of predictions. The precision of the model increases with the increasing number of trees in the forest, and it eliminates the issue of overfitting [16] [20].

2.3 Min-Max Scaler

The performance of many ML algorithms is considerably improved when the numerical input variables are scaled to a standard range. This scenario applies to algorithms such as LR, which uses the weighted sum of the input, and k-NN, which utilizes distance measurements. Techniques that employ distance measures are also included in this category. Min-Max scaler allows scaling of the data in a dataset to a given range by using the minimum and maximum value of each feature in the dataset, see Eq. (3). In the equation, Cn represents the newly calculated value, K represents the current value, Kmin represents the minimum value for the feature, and Kmax represents the maximum value of the feature [21].

$$C_n = (K - K_{min}) / (K_{max} - K_{min}) \qquad (3)$$

2.4 K-Fold Cross-Validation

K-fold cross-validation is a technique for analyzing and testing the performance of an ML model. It aids in comparing and selecting a suitable model for a certain predictive modeling issue. K-fold is simple to comprehend and execute and is often used to calculate the model’s efficiency scores. These features make K-fold a potent instrument for picking the optimal model for a given job. K-fold cross-validation operates as follows: (i) The number of folds (k) is determined. k is often 5 or 10. However, any value smaller than the length of the dataset can be used. (ii) The dataset is divided into k (if possible) equal sections (called folds). (iii) k minus one fold is selected as the training set. The remaining fold is the testing set. On each cross-validation iteration, a new model must be trained independently by the model trained on the previous iteration. (iv) Validation on the testing set is performed. (v) The validation result is saved. (vi) Steps 3–6 are repeated k times. The leftover fold is used as the test set each time. In the end, the model should be verified on each fold. (vii) The findings from step 6 should be averaged to obtain the final score [22] [23].

3 Related Works

In this section, we discuss the relevant efforts that employed ML approaches for detecting cyberattacks on HTTP.

An anomaly detection method for HTTP communications was proposed by Park S. et al. This method uses a convolutional autoencoder (CAE) in conjunction with character-level binary image modification. The CAE comprises an encoder and a decoder with convolutional neural network structures that are mirror images of one another. It attempts to generate an image that is very close to the original when given an image reconstructed from a message. The CAE is trained to achieve this goal by reducing the binary cross entropy (BCE) between the typical message’s input and output images. After receiving sufficient training, the suggested system can identify an abnormal message if the BCE of the message is more than a certain threshold value. The results of the experiments reveal that the suggested method outperforms traditional ML methods, such as a one-class SVM and an isolation forest. Both of these methods use input characteristics that are heuristically picked. In addition, enhanced performance may be accomplished by employing a complex CAE structure and a different decision variable, particularly binary cross varentropy, in place of BCE to obtain the desired results. Park S. et al. used character embedding in the image transformation to evaluate the validity of character-level image modification. This approach demands a computing burden but yields a slight performance increase [24].

Yi Xie S. et al. used an original server-side security strategy to combat the Web proxy-based distributed DoS attack. This strategy extracts the behavior characteristics of the proxy-to-server traffic by utilizing the temporal and spatial locality. Thus, the scheme is independent of the traffic intensity and frequently varying Web contents. A nonlinear mapping function was implemented to shield weak signals from the disruption caused by infrequently occurring large values. A novel hidden semi-Markov model parameterized by Gaussian-mixture and Gamma distributions was developed to represent the time-varying traffic behavior of web proxies. The new technique reduces the number of factors that requires calculation. It can also better characterize the dynamic development of the proxy-to-server traffic than the static statistics. The necessity of both fine- and coarse-grained detection is met by presenting two diagnostic techniques operating at different sizes. A new attack response approach suggested in the study of Yi Xie S. et al. is called soft control. Instead of dismissing suspicious traffic with hostility, it modifies the behavior of the individuals involved, allowing them to behave typically. This strategy may protect the quality of the services provided to genuine users. The results of the experiments demonstrated that the recommended strategy is successful [25] [26].

A method using semantic approaches in web application security was presented by Munir R. F. et al. The authors claimed this method has never been discussed or presented before. In this method, the ontology of the HTTP was discussed to protect the HTTP against attacks on communication protocols. The authors concentrate on attacks against communication protocols, such as anomalous HTTP messages, HTTP request smuggling, and HTTP answer splitting, within the scope of their work. As evidenced by the assessment findings, the suggested method outperforms the existing methods when dealing with these attacks. Moreover, it exhibits an excellent detection rate and a low false positive (FP) rate [27].

An SDN-assisted DDoS attack defense approach was proposed by Hong K. et al. This system can identify and neutralize slow HTTP DDoS attacks. It also depends on a slow

HTTP DDoS Defense Application (SHDA) embedded within an SDN controller. Instead of being handled by a web server, suspicious partial HTTP requests are received and managed by the SHDA. Once it has determined that incomplete HTTP requests come from an attacker through timeout-based DDoS detection, the SHDA mitigates the DDoS attack by requesting the SDN controller to update the flow rules to block the flow coming from the attacker at the network switches. This step prevents the flow from being received by the switches. The simulation findings showed that the provided DDoS defensive strategy can efficiently prevent slow HTTP DDoS attacks, thereby enabling a web server to continue functioning normally [28].

An anomaly-based DDoS attack detection approach was suggested and applied by Başkaya D and Samet R using ML methods on online systems. An investigation of the impact of various preprocessing methods and ML methods on identifying different forms of DDoS attacks was conducted within the framework of the technique presented. The research showed that when combined with feature reduction preprocessing, multilayer perception (MLP) is the most effective method for detecting DDoS attacks on online systems, with an accuracy rate of 99.2%. The procedures for identifying DDoS attacks in the prepared datasets were used for detecting DDoS attack types in the NSL-KDD dataset. The results vary according to the ML algorithms and the preprocesses used. This finding is similar to the results obtained for the prepared datasets. The average accuracy rate was 89.9% when MLP and RF were used in the calculation [29].

The central aim of the present study is to propose a model for identifying appropriate supervised ML methods for detecting cyberattacks on HTTP. Thus, the proposed model is called ML-HTTP. The ML-HTTP examines six techniques to detect cyberattacks on HTTP. These techniques are NB, K-NN, DT, SVM, LR, and RF. The method with the greatest performance is determined for the proposed model.

4 Proposed ML-HTTP Attack Detection Model

The proposed ML-HTTP method for detecting attacks on HTTP services is discussed in this section. The preprocessing of the HTTP_UNSW_NB15 dataset for training and testing the ML-HTTP method is also included.

4.1 HTTP-UNSW_NB15 Dataset Preprocessing

Data transformation and normalization operations should be made on the HTTP-UNSW_NB15 dataset prepared for applying the ML algorithms.

4.1.1 Irrelevant Features

As stated earlier, the HTTP_UNSW_NB15 dataset has been cleaned to contain only the HTTP records. As a consequence, some features should be eliminated because they are, totally, irrelevant to the HTTP service. Table 2 summarize the irrelevant features.

Table 2: Irrelevant features

Feature	Description	Reason
proto	Transport protocol	Contains only TCP with the HTTP service
service	http, ftp, smtp, ssh, dns, ftp-data ,irc and (-) if not much used service	Contains only DNS service, all other services were removed.
is_ftp_login	If the ftp session is accessed by user and password then 1 else 0	Related to FTP service

ct_ftp_cmd	No of flows that has a command in ftp session	Related to FTP service
is_sm_ips_ports	If source and destination IP addresses equal and port numbers equal then, this variable takes value 1 else 0	All values are equal to zero with the HTTP service

4.1.2 Data Transformation

Data transformation is an essential data preparation technique that must be applied to data to provide easy-to-comprehend patterns and allow relevant data extraction. The datasets contain both numerical and categorical characteristics (string data types). However, machines cannot directly analyze categorical data. For further processing, the categorical data must be translated into numerical data. The HTTP_UNSW NB15 dataset contains several categorical characteristics that require numeric transformation. The suggested technique utilizes the well-known label encoding mechanism to accomplish this task [30] [31]. Table 3 displays the transformed attribute and its new numerical values. In addition, the objective of this effort is to categorize traffic as either attack or regular, without revealing the sort of attack. All attack kinds are designated only as "Attack" in the output column, so that the output column has only two labels, "Normal" and "Attack", which have been rendered as 0 and 1, respectively.

Table 3: Transformed attribute

#	Feature	Before Transformation	After Transformation
4	state	FIN	0
		CON	1
		RST	2

4.1.3 Data Normalization

Normalization is a widely used data preparation technique that involves changing the values of numeric columns in a dataset to a common scale. It is particularly useful when the attributes of a dataset have different ranges, as it can improve the performance and reliability of an ML model. One way to normalize a dataset is by using the Min-Max scaler technique, which shifts and rescales the values of the attributes so that they range from 0 to 1 [21] [31]. The proposed ML-HTTP method utilizes the Min-Max scaler to normalize the HTTP_UNSW_NB15 dataset. Table 4 provides a sample of the normalized dataset, demonstrating the transformation of the original attribute values to a common scale. Fig. 1 illustrates the UNSW_NB15 dataset data preprocessing steps.

Table 4: After data normalization

No.	Instances	Label
1	0.02786188 0 0.5 0.002331002 0.00100553 0.00018548 0.000109546 0.00201431 0.138392857 0.995535714 0.001613374 0.000359817 0.001552795 0.000502765 0.016635096 0.012409106 0.010727721 0.000571096 0 0.35438469 0.421055616 0 0.449218911 0.254581607 0.36372598 0.028985507 0.057422969 0.005813953 2.73E-05 0.083333333 0.333333333 0 0 0 0.02173913 0 0 0.033333333 0 0.076923077 0	0
2	0.039132719 0 0.5 0.003108003 0.00100553 0.000208156 0.000120839 0.001569324 0.138392857 0.995535714 0.001264561 0.000279825 0.001552795 0.000502765 0.019130279 0.017333264 0.01118341 0.000761494 0 0.416022955 0.61173727 0 0.553808808 0.379408996 0.376647297 0.023646072 0.065826331 0.005813953 3.64E-05 0.083333333 0.333333333 0 0 0 0 0 0 0.033333333 0 0 0	0

No.	Instances	Label
3	0.003708626 0 0.5 0.002331002 0.000754148 0.000191294 8.23E-05 0.013869699 0.138392857 0.995535714 0.012923421 0.002024474 0.001552795 0.000502765 0.002195865 0.002183257 0.001214409 0.00011732 0 0.769795828 0.076306753 0 0.091760019 0.015036849 0.114763648 0.03051106 0.053921569 0.005813953 1.01E-05 0.166666667 0.333333333 0.021276596 0.111111111 0 0.02173913 0 0 0.033333333 0.016949153 0.153846154 0 1	1
4	0.00679653 0 0.5 0.002331002 0.000754148 0.00019769 0.000117639 0.007564622 0.138392857 0.995535714 0.007230896 0.001542543 0.001552795 0.000502765 0.004043444 0.003329707 0.002041655 0.000169722 0 0.025813118 0.239605654 0 0.204392502 0.120775195 0.160083768 0.032036613 0.086834734 0.005813953 3.58E-05 0.083333333 0.333333333 0.021276596 0.111111111 0 0.043478261 0 0 0.033333333 0.016949153 0.076923077 0	1
5	0.004275192 0 0.5 0.002331002 0.000502765 0.000259323 1.71E-05 0.010610899 0.995535714 0.995535714 0.014404892 0.000465195 0.001552795 0.000251383 0.002534858 0.003110886 0.001258089 0.000164844 0 0.257288582 0.010810551 0 0.096652853 0.025474318 0.110334695 0.04805492 0.00140056 0.005813953 0 0.166666667 0.333333333 0 0 0 0 0 0 0.033333333 0 0.076923077 0	1

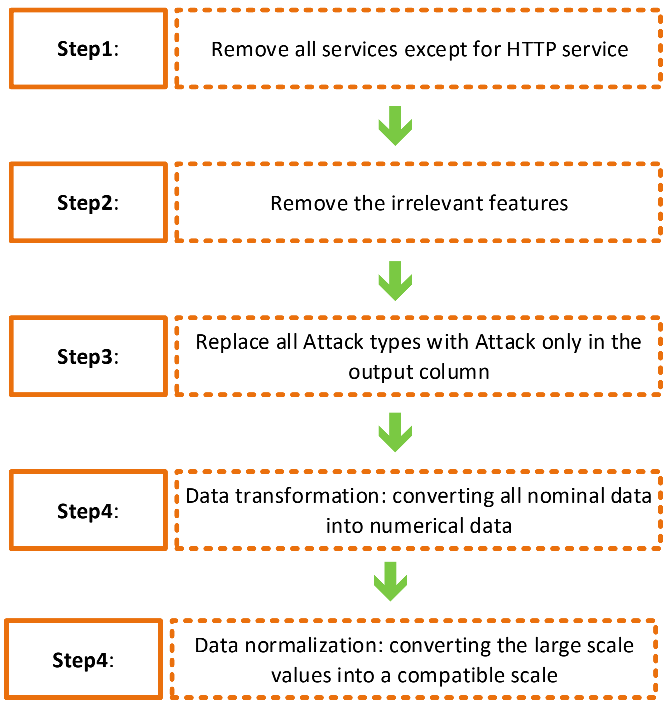
Step1:
Remove all services except for HTTP service
Step2:
Remove the irrelevant features
Step3:
Replace all Attack types with Attack only in the output column
Step4:
Data transformation: converting all nominal data into numerical data
Step4:
Data normalization: converting the large scale values into a compatible scale

Fig. 1: HTTP_UNSW_NB15 dataset preprocessing steps

4.2 Proposed ML-HTTP Model

ML provides several methods for constructing models that use data to make predictions. The ML-HTTP model employs the most prevalent methods in ML to identify HTTP service attacks. These methods include: NB, K-NN, DT, SVM, LR, and RF. The ML-HTTP model has two primary steps to fulfill its primary objective of detecting attacks on the HTTP service. In Stage 1, the UNSW NB15 dataset is prepared for the ML-HTTP model. In Stage 2, the proposed ML-HTTP model trains and evaluates the aforementioned ML techniques to achieve the intended objective.

4.2.1 Stage 1: UNSW_NB15 Preparation

The UNSW_NB15 dataset must be prepared so that it can be readily interpreted by the ML-HTTP model. Several processes are undertaken in the preparation phase. The UNSW_NB15 dataset is filtered to generate the HTTP_UNSW NB15 dataset by containing only HTTP service records. Then, all attack kinds in the output columns are renamed "Attack" because the ML-HTTP model is intended to assess whether HTTP traffic is an attack or normal traffic, independent of the attack kind. Then, all of the nonnumerical data in the HTTP-UNSW NB15 dataset are converted to numbers to conform to ML standards by using the label encoding methodology. The HTTP-UNSW NB15 dataset's data are scaled using the MinMaxscaler technique. Fig. 2 depicts the actions taken during Stage 1. The HTTP-UNSW NB15 dataset is now prepared for Stage 2.

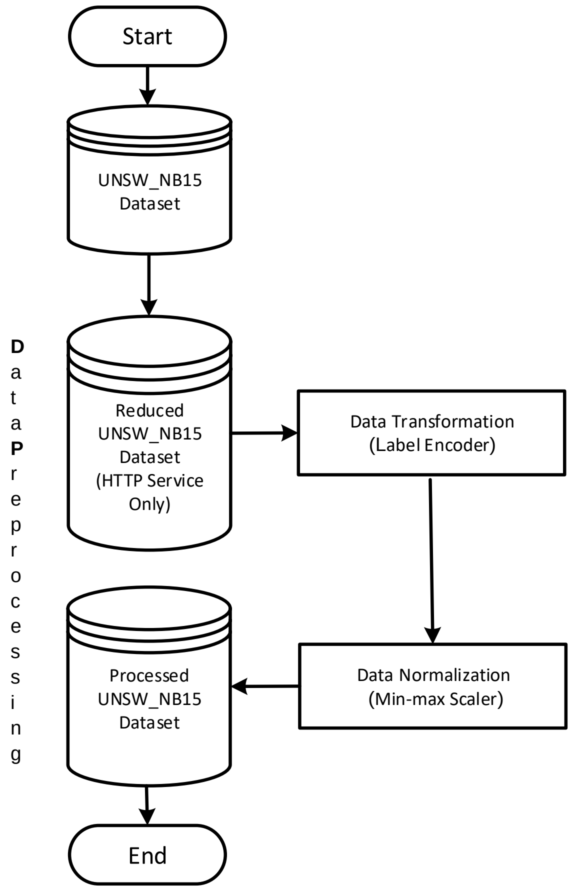
Start
UNSW_NB15 Dataset
Data Preprocessing
Reduced UNSW_NB15 Dataset (HTTP Service Only)
Data Transformation (Label Encoder)
Data Normalization (Min-max Scaler)
Processed UNSW_NB15 Dataset
End

Fig. 2: Stage 1 UNSW_NB15 preparation

4.2.2 Stage 2 ML-HTTP model training and testing

After the preparation is completed in Stage 1, the HTTP-UNSW_NB15 dataset is ready to execute Stage 2. In Stage 2, the HTTP-UNSW NB15 dataset is used to train the six ML algorithms (NB, K-NN, DT, SVM, LR, and RF) utilized by the ML-HTTP model. The performance of various algorithms is evaluated to select the optimal one. Training and testing are conducted using the K-fold technique (discussed above). The HTTP-UNSW NB15 dataset is partitioned into five groups: four of which are utilized for training, whereas one is used for testing. The training and testing groups are continuously rotated until all groups are utilized for training and testing. In this manner, the ML-HTTP model is evaluated accurately without bias. The final response to the ML-HTTP model is either regular traffic or attack traffic. Fig. 3 shows the steps performed in Stage 2.

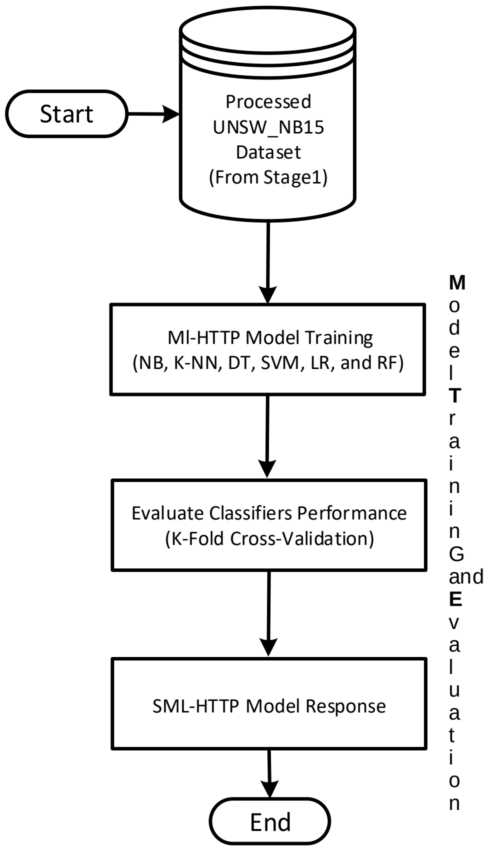
Start
Processed UNSW_NB15 Dataset (From Stage1)
Ml-HTTP Model Training (NB, K-NN, DT, SVM, LR, and RF)
Evaluate Classifiers Performance (K-Fold Cross-Validation)
SML-HTTP Model Response
End
Model TraininG and Evaluation

Fig. 3: Stage 2 ML-HTTP model training and testing

5 ML-HTTP Model Implementation

Python was utilized to apply the suggested ML-HTTP model. Python is a well-known and widely used programming language developed by the Python Software Foundation. It offers the implementation of various ML algorithms, such as NB, K-NN, DT, SVM, LR, and RF. In addition, Python has several tools and modules that support various machine approaches. We deployed some of these tools, such as sklearn and pandas, in implementing and evaluating the suggested model ML-HTTP [32] [33] [34]. The proposed model was implanted using Python 3.10.6 language, Intel Core i7-10870H CPU, 16 GB RAM, and 64-bit MS Windows 10 OS.

6 ML-HTTP Model Evaluation Metrics

This section covers the metrics used to evaluate the ML-HTTP model proposal. For a given collection of test data, the confusion matrix is used to evaluate the performance of a classification model. True positive (TP), true negative (TN), FP, and false negative (FN) are the constituents of the confusion matrix. TP indicates that the model has successfully identified an attack. TN indicates that the model has correctly identified that no attack exists. FP indicates that the model has failed to classify the presence of an attack. FN indicates that the model has failed to identify that no attack exists [35] [36]. Many metrics may be computed using the confusion matrix to assess the proposed ML-HTTP model. Accuracy (Acc), recall (Re), precision (Pr), and Matthew's correlation coefficient (MCC) are some of the most used and practical measures. Acc is regarded as one of the finest classification model evaluation metrics. In our suggested ML-HTTP model, it is the ratio of correct HTTP attack predictions to the total number of HTTP attack predictions (correct and incorrect). Eq. (4) is used to calculate the Acc of the proposed ML-HTTP model. Re is an additional crucial parameter for evaluating classification models. In our suggested ML-HTTP model, it is the ratio of the number of correct real HTTP attack predictions to the total number of actual HTTP attack predictions (correct and incorrect). Eq. (5) is used to calculate the ML-HTTP model presenting Re. Pr is also regarded as a crucial parameter for assessing a classification model. In our suggested ML-HTTP model, it is the ratio of the number of correct real positive HTTP attack predictions to the total number of positive HTTP attack predictions (accurate and incorrect). Eq. (6) may be used to determine the ML-HTTP model suggesting Pr. MCC is the last metric that should be used to evaluate an ML model. MCC is used to evaluate the quality of a binary classification model, such as the ML-HTTP model we suggested. The ML-HTTP model suggesting MCC is calculated using Eq. (7) [16] [30] [35] [36]. Note: In the equations, the ML-HTTP is abbreviated as MH.

$$\text{ML-HTTP}_{\text{Acc}} = (\text{MH}_{\text{TP}} + \text{MH}_{\text{TN}}) / (\text{MH}_{\text{TP}} + \text{MH}_{\text{TN}} + \text{MH}_{\text{FP}} + \text{MH}_{\text{FN}}) \quad (4)$$

$$\text{ML-HTTP}_{\text{Re}} = (\text{MH}_{\text{TP}}) / (\text{MH}_{\text{TP}} + \text{MH}_{\text{FN}}) \quad (5)$$

$$\text{ML-HTTP}_{\text{Pr}} = (\text{MH}_{\text{TP}}) / (\text{MH}_{\text{TP}} + \text{MH}_{\text{FP}}) \quad (6)$$

$$\text{ML-HTTP}_{\text{MCC}} = \quad \text{MH}_{MCC} = \frac{((\text{MH}_{TP} * \text{MH}_{TN}) - (\text{MH}_{FP} * \text{MH}_{FN}))}{\sqrt{(\text{MH}_{TP} + \text{MH}_{FP}) * (\text{MH}_{TP} + \text{MH}_{FN}) * (\text{MH}_{TN} + \text{MH}_{FP}) * (\text{MH}_{TN} + \text{MH}_{FN})}} \quad (7)$$

7 ML-HTTP Model Evaluation Results

This section presents the results of the proposed ML-HTTP model's evaluation. The suggested ML-HTTP model was assessed using the Acc, Re, Pr, and MCC metrics stated in the preceding section. Figures 4 to 7 depict the Acc, Re, Pr, and MCC of the proposed ML-DNS model using the six evaluated methods: DT, K-NN, NB, SVM, LR, and RF. The figures show that the RF and LR approaches achieved the greatest Acc (96.32%), Re (100%), Pr (99.99%), and MCC (99.97%) among the six methods. The NB technique had the worst outcomes across three of the four measures. In addition, the RF and LR approaches beat the other techniques with respect to the Acc metric, which is one of the most trustworthy metrics among the four metrics in the suggested ML-HTTP model and the utilized dataset.

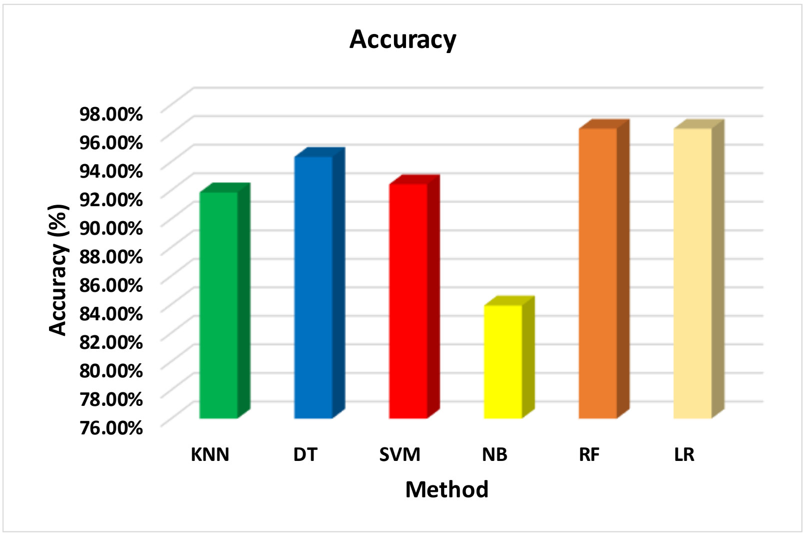

Fig. 4: Acc of the ML-HTTP

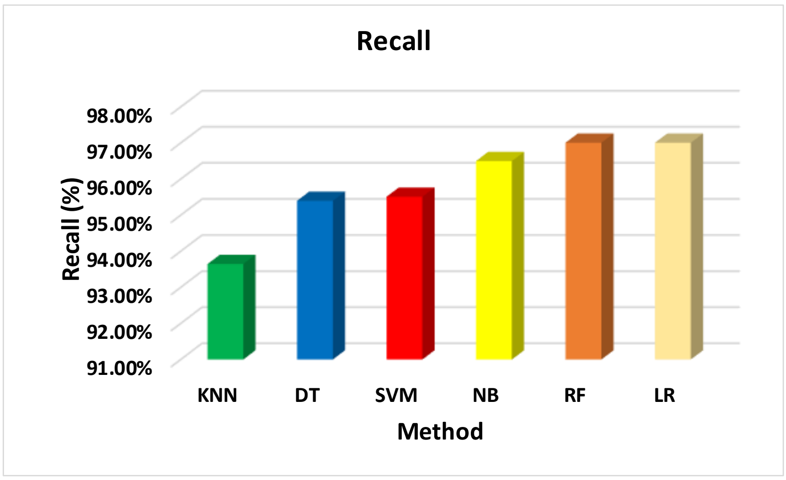

Fig. 5: Re of the ML-HTTP

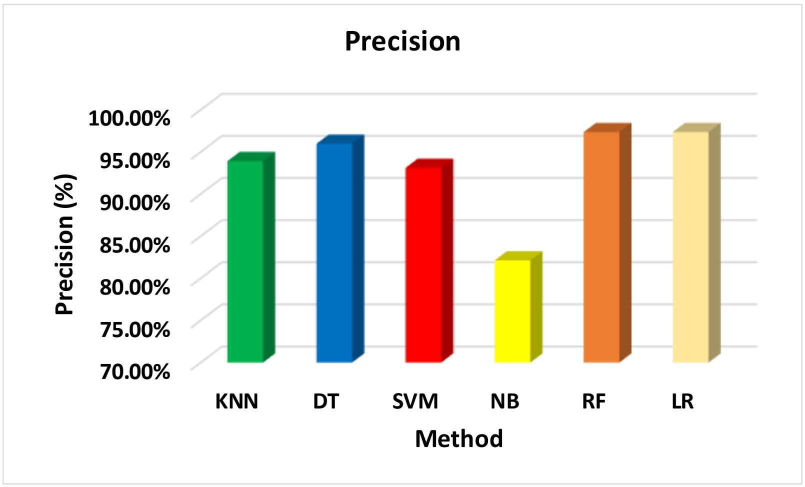

Fig. 6: Pr of the ML-HTTP

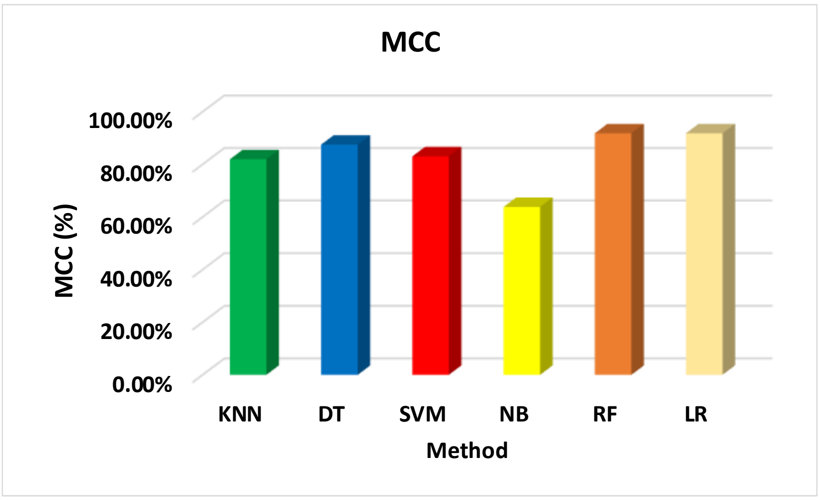

Fig. 7: MCC of the ML-HTTP

8 Conclusion

In recent years, security has become a key concern because of the increase in unauthorized network access and network resource abuse. Moreover, HTTP attacks have become such a concern. Thus, recognizing them as quickly as possible is crucial to allow the system network to sustain as little damage as feasible. In recent years, ML classifiers have been widely used in IDS because of their adaptability, generalization capability, and resilience. In the present study, ML classifiers, namely, SVM, K-NN, NB, DT, LR, and RF, were used to identify network intrusion. Their performance was tested using the UNSW_NB15 dataset. This study aimed to assess how well these classifiers can identify network intrusion. After the dataset was first preprocessed, the model was trained and assessed based on the discovered pertinent features. Based on the empirical data, the best-performing classifiers were the RF and LR. Both achieved the greatest Acc (96.32%), Re (100%), Pr (99.99%), and MCC (99.97%) among the six methods. In the future, the effectiveness of the suggested model will be assessed using additional datasets, and an optimization method will be used to pick the most pertinent HTTP properties from the datasets for the model.

ACKNOWLEDGEMENTS

This paper is funded by Al-Zaytoonah University of Jordan with the grant number 22/12/2019-2020.

References

[1] M. Trevisan, D. Giordano, I. Drago, and A. S. Khatouni. (2019). Measuring HTTP/3: Adoption and performance. in *2021 19th Mediterranean Communication and Computer Networking Conference (MedComNet)*, 2021, pp. 1–8.

[2] B. Pollard, *HTTP/2 in Action*. Simon and Schuster, 2019.

[3] A. Dhanapal and P. Nithyanandam. (2017). An effective mechanism to regenerate HTTP flooding DDoS attack using real time data set. in *2017 International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT)*, pp. 570–575.

[4] D. Nashat and S. Khairy. (2021). Detecting Http Flooding Attacks Based on Uniform Model. in *2021 International Conference on Networking and Network Applications (NaNA),* pp. 94–98.

[5] “Website Security Stats You Should Know.” https://ithemes.com/blog/website-security-stats/ (accessed Mar. 03,).

[6] P. Goldschmidt and J. Kučera. (2021). Defense against syn flood dos attacksˇ using network-based mitigation techniques. in *2021 IFIP/IEEE International Symposium on Integrated Network Management (IM),* pp. 772–777.

[7] M. Jang, S. Lee, J. Kung, and D. Kim. (2020) Defending against flush+ reload attack with DRAM cache by bypassing shared SRAM cache. *IEEE Access,* vol. 8, pp. 179837–179844.

[8] K. Pranathi, S. Kranthi, A. Srisaila, and P. Madhavilatha. (2018). Attacks on web application caused by cross site scripting. in *Second International Conference on Electronics, Communication and Aerospace Technology (ICECA),* 2018, pp. 1754–1759.

[9] T. Zhang and X. Guo. (2020). Research on SQL Injection Vulnerabilities and Its Detection Methods. in *2020 4th Annual International Conference on Data Science and Business Analytics (ICDSBA),* pp. 251–254.

[10] H. A. Alharbi, H. I. Alshaya, M. M. Alsheail, and M. H. Koujan. (2021) Analyzing Graduation Project Ideas by using Machine Learning. *International Journal of Interactive Mobile Technologies,* vol. 15, no. 23.

[11] M. Maabreh, I. Obeidat, E. A. Elsoud, A. Alnajjar, R. Alzyoud, and O. Darwish. (2022). Towards Data-Driven Network Intrusion Detection Systems: Features Dimensionality Reduction and Machine Learning. *International Journal of Interactive Mobile Technologies*, vol. 17, no. 14.

[12] S. Sharma, P. Zavarsky, and S. Butakov. (2020). Machine learning based intrusion detection system for web-based attacks. in *2020 IEEE 6th Intl Conference on Big Data Security on Cloud (BigDataSecurity), IEEE Intl Conference on High Performance and Smart Computing,(HPSC) and IEEE Intl Conference on Intelligent Data and Security (IDS),* pp. 227–230.

[13] S. M. Kasongo and Y. Sun. (2020). Performance analysis of intrusion detection systems using a feature selection method on the UNSW-NB15 dataset. *Journal of Big Data,* vol. 7, pp. 1–20.

[14] N. Moustafa and J. Slay. (2015). UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). in *2015 military communications and information systems conference (MilCIS),* pp. 1–6.

[15] Z. Zoghi and G. Serpen. (2021). Unsw-nb15 computer security dataset: Analysis through visualization. *ArXiv Prepr. ArXiv210105067.*

[16] Ü. Çavuşoğlu. (2019). A new hybrid approach for intrusion detection using machine learning methods. *Applied. Intelligence.,* vol. 49, pp. 2735–2761.

[17] T. M. Ma, K. Yamamori, and A. Thida. (2020). A comparative approach to Naïve Bayes classifier and support vector machine for email spam classification. in *2020 IEEE 9th Global Conference on Consumer Electronics (GCCE),* pp. 324–326.

[18] P. Wang, Y. Zhang, and W. Jiang. (2021). Application of K-Nearest Neighbor (KNN) Algorithm for Human Action Recognition. in *2021 IEEE 4th Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC)*, vol. 4, pp. 492–496.

[19] H. Elaidi, Y. Elhaddar, Z. Benabbou, and H. Abbar. (2018). An idea of a clustering algorithm using support vector machines based on binary decision tree,” in *International Conference on Intelligent Systems and Computer Vision (ISCV)*, pp. 1–5.

[20] R. D. Ravipati and M. Abualkibash. (2019). Intrusion detection system classification using different machine learning algorithms on KDD-99 and NSL-KDD datasets-a review paper. *International Journal of Computer Science and Information Technology IJCSIT* vol. 11.

[21] M. M. Ahsan, M. P. Mahmud, P. K. Saha, K. D. Gupta, and Z. Siddique. (2021). Effect of data scaling methods on machine learning algorithms and model performance. *Technologies*, vol. 9, no. 3, p. 52.

[22] T.-T. Wong and N.-Y. Yang. (2017). Dependency analysis of accuracy estimates in k-fold cross validation. *IEEE Transactions on Knowledge and Data Engineering.* vol. 29, no. 11, pp. 2417–2427.

[23] Ö. Karal. (2020). Performance comparison of different kernel functions in SVM for different k value in k-fold cross-validation. in *2020 Innovations in Intelligent Systems and Applications Conference (ASYU)*, pp. 1–5.

[24] S. Park, M. Kim, and S. Lee. (2018). Anomaly detection for http using convolutional autoencoders. *IEEE Access*, vol. 6, pp. 70884–70901.

[25] Y. Xie, S. Tang, Y. Xiang, and J. Hu. (2012). Resisting web proxy-based http attacks by temporal and spatial locality behavior. *IEEE Transactions on Parallel Distributed Systems.* vol. 24, no. 7, pp. 1401–1410.

[26] A. Hnaif, K. M. Jaber, M. A. Alia, and M. Daghbosheh. (2021). Parallel scalable approximate matching algorithm for network intrusion detection systems. *Internationl Arab Journal of Information Technology*, vol. 18, no. 1, pp. 77–84.

[27] R. F. Munir, N. Ahmed, A. Razzaq, A. Hur, and F. Ahmad. (2011). Detect HTTP specification attacks using ontology. in *Frontiers of Information Technology*, 2011, pp. 75–78.

[28] K. Hong, Y. Kim, H. Choi, and J. Park. (2017). SDN-assisted slow HTTP DDoS attack defense method. *IEEE Communications Letters.* vol. 22, no. 4, pp. 688–691.

[29] D. Başkaya and R. Samet. (2020). Ddos attacks detection by using machine learning methods on online systems. in *2020 5th International Conference on Computer Science and Engineering (UBMK)*, pp. 52–57.

[30] M. M. Abualhaj, A. A. Abu-Shareha, M. O. Hiari, Y. Alrabanah, M. Al-Zyoud, and M. A. Alsharaiah. (2022). A Paradigm for DoS Attack Disclosure using Machine Learning Techniques,” *International Journal of Advanced Computer Science and Applications.* vol. 13, no. 3.

[31] S. Capobianco, L. M. Millefiori, N. Forti, P. Braca, and P. Willett. (2021). “Deep learning methods for vessel trajectory prediction based on recurrent neural networks. *IEEE Transactions on Aerospace and Electronic Systems.* vol. 57, no. 6, pp. 4329–4346.

[32] Aaryan, B. Kanisha, V. Jayaraman, and B. Kanisha. (Apr. 2022). Forecasting stock market price using LSTM-RNN. in *2022 2nd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)*, pp. 1557–1560. doi: 10.1109/ICACITE53722.2022.9823818.

[33] A. Sweigart. (2020). *Beyond the Basic Stuff with Python: Best Practices for Writing Clean Code*. No Starch Press,.

[34] M. Kolhar, F. Al-Turjman, A. Alameen, and M. M. Abualhaj. (2020). A Three Layered Decentralized IoT Biometric Architecture for City Lockdown During COVID-19 Outbreak. *IEEE Access.* vol. 8, pp. 163608–163617.

[35] M. M. S. Pangaliman, F. R. G. Cruz, and T. M. Amado. (2018). Machine learning predictive models for improved acoustic disdrometer. in *2018 IEEE 10th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment and Management (HNICEM),* pp. 1–5.

[36] N. Ajithkumar, P. Aswathi, and R. R. Bhavani. (2017). Identification of an effective learning approach to landmine detection. in *2017 1st international conference on electronics, materials engineering and nano-technology (IEMENTech)*, pp. 1–5.

Notes on contributors

Hani Al-mimi is currently an Assistant Professor in the Faculty of Science and Information Technology at Al-Zaytoonah University of Jordan. His research interests include areas such as Computer Networks, Artificial Intelligence, Wireless and Mobile Networks, Computer Security, Cyber Security, and Cryptography.

Nesreen A. Hamad is an Instructor at the Department of Artificial Intelligence at Al-Zaytoonah University of Jordan, Jordan. Currently, she is a Ph.D. Student at the Department of Computer Science at Jordan University, Jordan. She received her M.Sc. degree in Computer Science from the University of Jordan, Jordan. Also, she received her B.Sc. degree in Computer Science from Al-Zaytoonah University of Jordan, Jordan. Her research interests include interconnection networks, metaheuristic algorithms, computational intelligence, bioinformatics, and data mining.

Mosleh M. Abu-Alhaj is a Full Professor and a senior lecturer in Al-Ahliyya Amman University. He received his first degree in Computer Science from Philadelphia University, Jordan, in July 2004, master degree in Computer Information System from the Arab Academy for Banking and Financial Sciences, Jordan in July 2007, and doctorate degree in Multimedia Networks Protocols from Universiti Sains Malaysia in 2011. His research area of interest includes VoIP, Multimedia Networking, and Congestion Control. Apart from research, Dr. Abu-Alhaj also does consultancy services in the above research areas and directs the Cisco academy team at Al-Ahliyya Amman University.

Mohammad Sh. Daoud received the Ph.D. degree in computer science from De Montfort University, U.K. He is currently an Associate Professor with the College of Engineering, Al Ain University, United Arab Emirates. His research interests include artificial intelligence, swarm systems, Secured Systems and Networks, and smart applications.

Ali Al-Dahoud is a staff member in IT Faculty/ Al Zaytoonah University of Jordan since 1996 and he is a Full Professor in Computer Science since 2011.

He served as CIS department chair, and the Dean of Science and IT 2012-2016, Dean of Students Affair 2019-2021 and he worked as the director of Foreign Relationships Office since 2017 until 2022, Al-Dahoud published more than 90 papers, he is the founder of ICIT and he served in many other international conferences and journals.

Mohammad Rasmi is currently serving as an associate professor in the Faculty of Science, Department of Cyber security, Zarqa University. He received his PhD in Network Security from Universiti Sains Malaysia in 2013. He received a Master degree in Computer Information System from the AABFS in 2004, and BSc degree in computer science from Zarqa Private University in 1999. His research interests include network forensics, web security, E-government strategy, cloud computing and software engineering.