IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received September 19, 2021, accepted October 12, 2021, date of publication October 15, 2021,
date of current version October 25, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3120626

Intrusion Detection System Using Machine Learning for Vehicular Ad Hoc Networks Based on ToN-IoT Dataset

ABDALLAH R. GAD[1,2], AHMED A. NASHAT[2], AND TAMER M. BARKAT[2]
[1]Department of Communication and Electronics Engineering, October High Institute for Engineering and Technology, 6th of October City 12596, Egypt
[2]Department of Electrical Engineering, Faculty of Engineering, Fayoum University, Faiyum 63514, Egypt

Corresponding author: Abdallah R. Gad (abdallah.gad@must.edu.eg)

ABSTRACT Vehicular ad hoc networks (VANETs) are a subsystem of the proposed intelligent transportation system (ITS) that enables vehicles to communicate over the wireless communication infrastructure. VANETs are used in multiple applications, such as improving traffic safety and collision prevention. The use of VANETs makes the network vulnerable to various types of attacks, such as denial of service (DoS) and distributed denial of service (DDoS). Many researchers are now interested in adding a high level of security to VANETs. Machine learning (ML) methods were used for constructing a high level of security capabilities based on intrusion detection systems (IDSs). Furthermore, the vast majority of existing research is based on NSL-KDD or KDD-CUP99 datasets. Recent attacks are not present in these datasets. As a result, we employed a realistic dataset called ToN-IoT that derived from a large-scale, heterogeneous IoT network. This work tested various ML methods in both binary and multi-class classification problems. We used the Chi-square (Chi^2) technique was used for feature selection and the Synthetic minority oversampling technique (SMOTE) for class balancing. According to the results, the XGBoost method outperformed other ML methods.

INDEX TERMS Intrusion detection system (IDS), Internet of Things (IoT), ToN-IoT dataset, machine learning (ML), vehicular ad hoc networks (VANETs).

I. INTRODUCTION

Now, Smart-autonomous vehicles (SAVs) have long been a core part of the Intelligent transportation system (ITS) idea and a core part of the future of the automotive industry. SAVs play a significant role in improving many factors, such as road safety, driving experience, and decision-making based on the collected information. The road infrastructure is constructed from information collected from SAVs and various communication technologies. The road infrastructure is based on connectivity between vehicles with road-side units (RSUs). RSUs are used by ITS to reduce accidents and increase the performance of driving [1]. Figure 1 shows the general architecture for VANETs.

Several types of attacks can affect VANETs since vehicles are connected over a wireless medium. Vehicular ad hoc network (VANET) performance is significantly reduced by these attacks that cause serious issues for drivers. As a result, securing VANET traffic against alteration, monitoring, and removal of vital messages has been a big concern and one of academia's and industries' top objectives. This influence over users' data could be used to their attacker's advantage to destroy the network [3]. Many dangers exist, such as denial of service (DoS), which introduces unwanted traffic into a network and denies or stops legitimate users from accessing resources. In addition, malware is a malicious code that obtains a benefit by exploiting a weakness in the network devices [3].

Securing vehicles from intrusions is a difficult problem since vehicles have traditionally been constructed without considering complete security needs. Relying on the premise that vehicles function autonomously and without communication capabilities. Because of the increased connections that provide an extensive attack surface, resource restrictions, and complexity of current cars. Traditional practical security countermeasures such as encryption techniques and access control are irrelevant to autonomous vehicles [4]. Reactive systems, such as IDS, have recently received more attention.

The associate editor coordinating the review of this manuscript and approving it for publication was Zhaojie Ju.

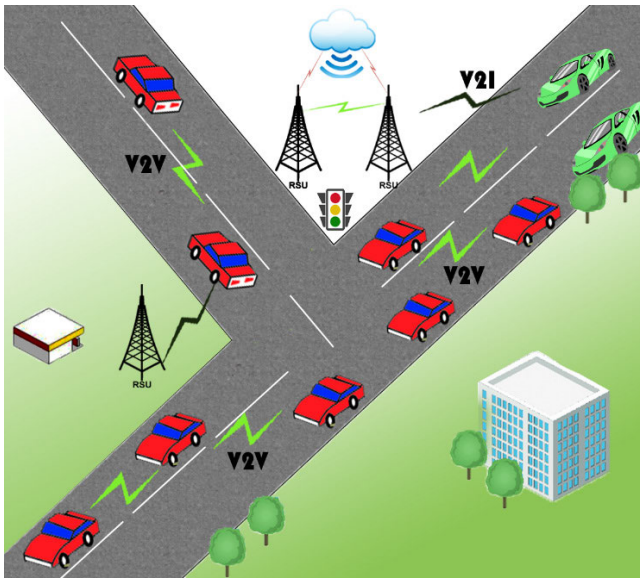

FIGURE 1. Vehicular ad hoc networks general architecture [2].

They can identify possible cyber-attacks on VANETs besides misbehaviors in linked vehicular networks. IDS makes their complementary solutions to proactive security countermeasures [5].

To successfully implement VANETs in ITS, innovative methods for securing VANET traffic should be devised and implemented. Every attack class in VANETs has a set of attributes that define its profile. In VANETs, machine learning (ML) methods are commonly used to analyze huge quantities of data and derive valuable rules for event detection, categorization, and prediction. ML methods have been deployed in various applications such as medical diagnosis, speech recognition, traffic prediction, recommendation systems, and intrusion detection. Machine learning (ML) is a desirable solution to detect intrusions in advance with acceptable speed and accuracy in VANETs [6]. Numerous ML methods have been implemented for IDS in VANETs, such as neural networks (NN), decision tree (DT), and support vector machine (SVM) [7].

The following steps will be followed during the research experimentation:

1. Selecting a new dataset called ToN-IoT [8]. The dataset has been examined carefully by dropping the flow identifier attributes to avoid bias toward attacks and overfitting.

2. Also, the ToN-IoT dataset has many challenges, such as a class imbalance problem, categorical features, and missing values. We present a hybrid model for solving problems related to the ToN-IoT network dataset.

3. For comparison, we tested various ML methods on the network ToN-IoT dataset; ML methods are logistic regression (LR), naive Bayes (NB), decision tree (DT), support vector machine (SVM), k-nearest neighbor (kNN), random forest (RF), and XGBoost. The following are the key contributions of this research:

1. Suggest ML-based IDS for VANETs with assessment metrics compared with other studies.

2. KDD-CUP99, NSL-KDD, and UNSW-NB15 datasets are used to assess most existing detection systems. These datasets are outdated, and they do not include contemporary IoT attacks. However, the suggested model's efficiency is assessed using a realistic ToN-IoT dataset. Accuracy, precision, recall, F1-score, and false-positive rate (FPR) are used as evaluation metrics.

3. Solving problems related to the ToN-IoT dataset, such as missing values, class imbalance, and irrelevant features that affect the performance of the IDS model.

4. The Chi^2 technique was used to select the best features of the dataset.

5. The SMOTE technique was used to solve, the class imbalance problem.

6. Applying various ML methods on the ToN-IoT network dataset and selecting the best classification method with high evaluation metrics that differentiate between normal and attack instances.

The paper is arranged as follows: the following section gives a quick overview of IDS for VANET. Section 3 gives a brief about the ToN-IoT dataset. Section 4 gives a brief about the used ML methods. The experimental methodology setup and feature preprocessing of the ToN-IoT dataset are presented in section 5. The experimental results are described in section 6. Finally, the conclusion is presented in section 7.

II. RELATED WORK

Generally, various researchers have proposed IDS using machine learning (ML) or deep learning (DL) methods. The used methods are k-nearest neighbor (kNN) [9], random forest (RF) [10], and support vector machine (SVM) [11]. Other researchers used DL methods such as convolution neural networks [12] and recurrent neural networks [13]. However, when employing these ML or DL methods, data must be processed to avoid many problems, such as missing values, categorical attributes, and attribute scaling. Without preprocessing of data, the outputs are insufficiently reliable. Over-fitting, under-fitting, and other problems related to the fact that we do not clean or process data carefully.

Many researchers have been interested in securing VANETs in recent years [14]–[20]. Many security vulnerabilities exist in VANET, which can cause these VANET applications to stop working. The goal of the IDS is to identify both internal and external threats with high accuracy [21], [22]. IDSs can be classified as signature-based IDS, anomaly-based IDS, and others [23] based on previously employed methodologies. Several techniques have recently been developed; the most promising are ML-based approaches [14], [16], [23]. Most pure IDSs, on the other hand, produce a high number of false positives and have low detection accuracy. Several studies in the literature have examined the capacity to apply ML for intrusion detection. In the realm of IDS, ML has shown promising outcomes.

Several researchers have used ML methods been to learn complicated patterns from acquiring data.

ML methods have been extensively used to address IDS concerns in various networks. For example, in the early work [14], the RF approach was used to automatically create infiltration patterns. Intrusions were then discovered by comparing the network activity to the constructed patterns. The authors used their knowledge to assess the model's performance. The authors used the KDD-CUP99 dataset. Both down-sampling and oversampling strategies were used to cope with the issue of imbalanced data. The experiment was then run in the WEKA environment, with 66% of the samples serving as train data and 34% serving as test data. The experimental output achieved a 94.7% detection rate and a 2% FPR.

The authors [24] used a genetic algorithm in conjunction with the kNN approach. The training dataset had a well-balanced number of samples. Alternatively, the testing set had 100 samples of each class in the dataset. Using GA, 30 chromosomes were produced and trained. All 35 features were fed into the kNN technique. According to the proposed model, the top accuracy for identifying attacks was 97.42% when only the best 19 features were evaluated.

Similarly, the authors [14] used an RF classifier for detecting undesired behavior in high-speed traffic data. The authors customized the Apache spark technology. The findings revealed that the framework could improve real-time network intrusion detection with large capacity and fast speed.

Al-Jarrah *et al.* [25] looked into the effect of feature selection techniques on RF performance. For this objective, they integrated RF with forwarding and backward ranking feature selection strategies. The KDD-CUP99 was filtered and eliminated redundant data. They applied several preprocessing techniques, such as normalization, discretization, and balancing. The RF–forward ranking approach was effective in the experiments. In addition to RF and kNN approaches, SVM was selected to detect malicious network intrusions in [26]. NB and DT were utilized in [27].

Many hybrid IDSs have been presented. Kim *et al.* [28] suggested a hybrid IDS that has 2 stages. Hierarchically, the suggested IDS combines a misuse detection model and an anomaly detection approach. First, the C4.5 algorithm was used for creating the misuse detection model. For splitting subsets, different models of one class SVM were used. Compared to traditional approaches, the results demonstrated improvements in terms of false-positive rate (FPR) and detection rate. Furthermore, the authors [29] suggested a hybrid multi-level IDS that efficiently detects existing and new threats by combining SVM and extreme machine learning. They also developed a modified k-means technique for constructing a modest and appropriate training dataset to increase the performance of the utilized classifiers. The results reveal that the strategy reduced training time while also improving IDS' overall performance.

The authors [23] suggested a hybrid IDS using C5.0's stacking ensemble and SVM. They tested their technique using the NSL-KDD dataset and the Australian Defence Force Academy datasets (ADFA).

The authors [30] devised ML-based IDS for detecting intruders in VANETs both globally and locally. To secure cluster heads, they employed an artificial neural network (ANN) approach to identify malicious multi-point relays. They used a lightweight SVM. In comparison to previous ML-based strategies, the results showed that the applied strategy was more resilient and trustworthy.

III. TON-IOT DATASET

ToN-IoT is the dataset used in this study. ToN-IoT contains a variety of data sources collected from the entire IIoT system, including telemetry data from connected devices, Linux operating system records, Windows operating system records, and network traffic for the IIoT system. The heterogeneous data were gathered from a medium-scale IoT network. UNSW Canberra IoT Labs and the Cyber Range created ToN-IoT. The network ToN-IoT dataset can be reached at the ToN-IoT repository [8]. Furthermore, ToN-IoT datasets were represented in CSV format with a labeled column representing attack or normal behavior and a sub-category attack-type, which indicates the different types of attacks, such as ransomware, password attack, scanning, denial of service (DoS), distributed denial of service (DDoS), data injection, backdoor, Cross-site Scripting (XSS), and man-in-the-middle (MITM). These various attacks were initiated and collected over the IIoT network against various IoT and IIoT sensors. Details of the dataset can be accessed in [8].

The attacks found in the ToN-IoT network dataset fall into 1 of 9 categories:

- **Scanning**: the attacker gathers data about the system using scanning. The data include available services on a victim system and open ports. The attacker executes scanning before launching any type of attack.
- **Cross-Site Scripting (XSS)**: A web server in an IoT application is affected by running malicious software like XSS. Another problem that affects IoT systems is that XSS attackers can compromise the information and procedures used for authenticating IoT systems.
- **Denial of Service (DoS)** is a popular flooding attack. The attacker executes a series of malicious to interrupt services. Service unavailability is the target of DoS attacks.
- **Distributed Denial of Service (DDoS)** is frequently performed via a network of infected systems known as bots. This attack is carried out by inundating target IoT resources with many connections, depleting the resources.
- **Backdoor** is a type of attack in which backdoor malware enables an adversary to obtain unwanted remote access to compromised IIoT systems. The adversary uses the backdoor to take possession of compromised IIoT computers and uses botnets to initiate DDoS attacks.
- **Injection Attack**: The attacker tries executing malicious software or inject malicious data into an IoT system.

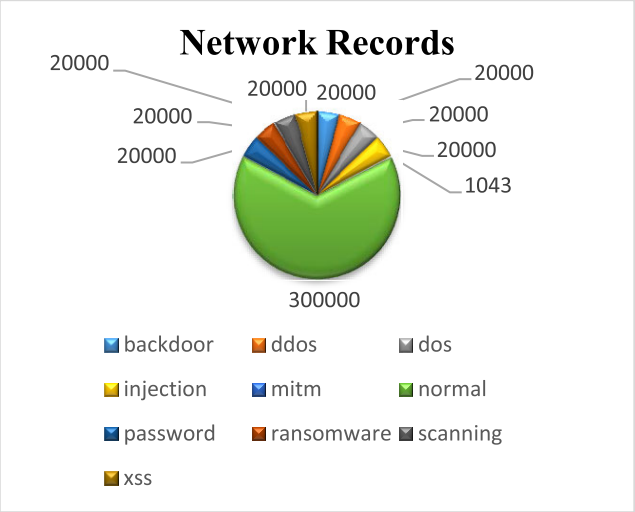

FIGURE 2. Statistics records of the network ToN-IoT dataset.

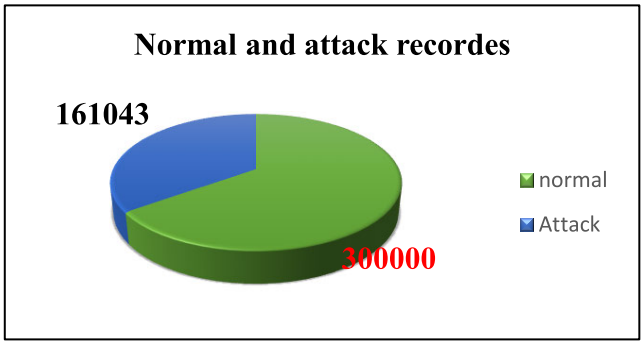

FIGURE 3. Statistics records of the network ToN-IoT dataset.

Additionally, the injection attack will interrupt the normal activity and control instructions in an IoT system.

- **Password Cracking Attack**: The attacker went to crack the password using various password cracking techniques, such as the dictionary and brute force. The attacker will be able to circumvent authentication methods and hack IoT devices because of this attack.
- **Man-In-The-Middle (MITM)** is a popular network attack where the MITM attacker can make the intercept of data flow into an IoT network. Port stealing is one of the most recent MITM attacks.
- **Ransomware** is a complex form of malware that prevents a legitimate user from accessing a device or service and then attempts to sell the decryption key, which enables access to the system. IoT applications and devices are promising goals since they conduct important tasks. When access is denied, it can lead to terrible significance, such as important financial damage to stakeholders.

All data points in the network ToN-IoT dataset are made up of 44 attributes and an attack-type labeled as normal or attack. As presented in Figures 2 and 3, we show the normal and attack statistics for network data records in the train-test ToN-IoT dataset.

IV. CANDIDATE SUPERVISED ML METHODS

Several machine learning (ML) methods have been tested on the ToN-IoT dataset. The selected methods are used to train and test with different parameters in the feature-engineering phase for intrusion detection purposes. We evaluated the different classifiers using accuracy, precision, recall, F1-score, false-positive rate (FPR), and the confusion matrix. Candidate approaches were chosen because they are extensively used in the security arena. The used approaches have shown strong performance in the creation of IDSs, and are successful in a range of sectors. We explore the following 8 strategies, in particular logistic regression (LR), naive Bayes (NB), decision tree (DT), support vector machine (SVM), k-nearest neighbor (kNN), random forest (RF), AdaBoost, as well as XGBoost.

Here is a quick rundown of these methods:

1- **Logistic Regression (LR)** [31] is a modification of the linear regression method. Despite its name, LR is regularly used for classification problems, since it can estimate the chance that an observation belongs to a specific class. It's useful in applications such as spam filtering and intrusion detection. If the predicted probability is higher than the defined threshold, the method will forecast that the instance fits the attack since it is higher than the threshold. Otherwise, it will predict that the instance belongs to the normal class. Based on the following Eq. 1, LR calculates the probability.

$$h_\theta(x) = \sigma\left(\theta^T X\right) \tag{1}$$

where h_θ is the hypothesis, x is the input feature vector, θ is the LR parameters, and σ (r is a sigmoid function that is used for the threshold definition. The sigmoid is defined as:

$$\sigma(r) = \frac{1}{1 + e^{-r}} \tag{2}$$

where r is the term $(\theta^T X)$ in the previous equation, the output is between (0:1).

2- **Naive Bayes** (NB) [32] is a probabilistic-based technique that uses the Bayes-theorem to do classification. NB is easy to implement. It has good results obtained in various cases. NB assumes normally distributed data and determines the conditional probability of a class. Furthermore, Bayes' theorem gives a systematic way to compute the conditional probability based on feature independence assumptions. Eq (3) expresses Bayes' theorem:

$$P(L \mid X) = \frac{P(X \mid L) P(L)}{P(X)} \tag{3}$$

where $P(L \mid X)$ the posterior probability of class L is, $P(L)$ is the prior probability, $P(X \mid L)$ is the likelihood function, and $P(X)$ is the probability. The training set is used to estimate these parameters.

3- **k-Nearest Neighbor (kNN)** [33], dissimilar to NB and LR, kNN is a non-parametric approach that makes no

expectations about the distribution of the underlying data. kNN is also a basic strategy that uses particular metrics to classify new instances from a test set to the nearest instance in the training set. Specifically, kNN searches the training set for a group of k observations closest to the test observation and assigns a label based on the most common class among the k neighbors. The number of neighbors and distance are the two fundamental parameters of the kNN technique. The Euclidean Distance was chosen since it is a commonly used distance measure.

Euclidean Distance Eq. is defined as (4):

$$d(x, y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \quad (4)$$

where $d(x, y)$ is a Euclidean distance function between the two samples, x_i is the initial observation, and y_i is the second sampling of the information, and n represents the observations.

4- **Decision Tree (DT)** [34] is a building approach that resembles a tree through branches and leaves. The internal node indicates classification rules; the branch shows results and the class label refers to the leaves. Information gain is used as a metric for selecting the optimal attributes used in the core node and branches in the training phase. The greatest information gain score is then used to build the decision node. By constructing a new sub-tree under the decision node, the cycle continues. If all items in the designated subgroups have the same value, the process will come to a halt, and the ultimate value will be computed as the output value. The cycle will be utilized to halt when there is just one node in the subgroup and no identifiable characteristic is detected. In this study, the Gini impurity is employed as splitting criteria, which chooses a feature for splitting at each phase of the tree training, as shown in (5):

$$G(D) = \sum_{I=1}^{C} (P(i) * (1 - P(i)) \quad (5)$$

where D is the training dataset, C is a collection of class labels, and p (i) is the proportion of samples having the class label I in C. When there is just one class in C, the Gini impurity is zero.

5- **Random Forest (RF)** [35] is a supervised ML technique that gives excellent outputs, even without any adjustment of parameters, and is exceedingly adaptable. Its appeal stems from the fact that it can be used to solve both classification and regression issues, which make up most ML jobs nowadays. RF is an ensemble learning approach that combines many decision trees to produce a model that predicts the data class more accurately and consistently. The decision tree has an overfitting problem, which is resolved by random forests. The Gini impurity as presented in Eq. (5) is also used as a split criterion.

6- **Support Vector Machine (SVM)** [36] is a classification technique that can handle both linear and non-linear datasets and is mostly characterized by a separating hyperplane. SVM's main objective is to identify a hyperplane that increases the difference between the classes. There are various kernel functions for characterizing the hyperplane, ranging from a linear kernel that tries discovering a simple linear separator between the classes to a non-linear kernel, such as radial basis function (RBF) kernel. In the experimentation, we used the commonly agreed RBF kernel. RBF is a non-linear mapping that seeks the linear optimum separation hyperplane by transforming the unique training data to a higher dimension.

$$K(x, y) = e^{-\frac{||x-y||^2}{2\sigma^2}} \quad (6)$$

where $\boldsymbol{\sigma}$ is the variance and the SVM hyper-parameter, ||x-y|| is the Euclidean distance between two points.

7- **AdaBoost** [37] is an iterative supervised learning algorithm that takes into account the predictions of numerous weak classifiers regularly. Modifying data sets and combining them using a weighted majority vote. It operates by giving greater weight to occurrences that are difficult to categorize and less weight to those that are already well classified. This boosting-based classifier is used in network security, specifically for intrusion detection.

8- **XGBoost** [38] is designed based on gradient-boosted decision trees. XGBoost is a high-speed and good performance algorithm compared to other ML algorithms. It represents a method for machine boosting or applying boosting to machines. For tree boosting methods, XGBoost (eXtreme Gradient Boosting) aids in maximizing memory and hardware resources. It has the advantages of improving the algorithm and modifying the model. It performs Taylor expansion for the cost function by the second derivative for making the results more accurate. The XGBoost method optimizes the objective function using an additive training approach, which implies that the latter phase's optimization process depends on the results of the preceding stage. The model's t-th objective function has been written as

$$obje^t = \sum_{i=1}^{n} l(y_i, y_i^{\wedge t-1}) + f_t(x_i)) + \Omega(f_t) + \mathrm{C} \quad (7)$$

where l represents the loss term of the t-th round, C is defined as a constant term, and Ω is the regularization term, shown as

$$\Omega(f_t) = \gamma T_t + 0.5\lambda \sum_{j=1}^{T} w_j^2 \quad (8)$$

where γ *and* λ are customization parameters. In general, the greater these two numbers are, the more basic the tree's structure is. Over-fitting issues can also be properly addressed.

Performing a second-order Taylor expansion on (7):

$$obje^t = \sum_{i=1}^{n} [l(y_i, y_i^{\wedge t-1}) + g_i f_t(x_i) .5 h_i f_t^2(x_i)] + \Omega(f_t) + \mathrm{C} \quad (9)$$

where g and h are the first derivatives and the second derivative. g and h can be defined as:

$$g_i = \partial_{y_i^{\wedge t-1}} l(y_i, y_i^{\wedge t-1}) \tag{10}$$

$$h_i = \partial^2_{y_i^{\wedge t-1}} l(y_i, y_i^{\wedge t-1}) \tag{11}$$

By Substitute the previous XGBoost Eq's and taking the derivative. Then:

$$w_j^* = -\frac{\sum g_i}{\sum h_i + \lambda} \tag{12}$$

$$obje^* = -0.5 \sum_{j=1}^{T} \frac{(\sum g_i)^2}{\sum h_i + \lambda} + \gamma.T \tag{13}$$

where $obje^*$ shows the loss function score. The smaller the $obje^*$, the better the structure of the tree. w^* is defined as the solution of weights.

V. EXPERIMENTAL METHODOLOGY

We present IDS for VANETs based on various data preparation and preprocessing techniques, as discussed in this section.

1. **Data preprocessing**: Cleaning and preparation are the core essential phases before feeding the data into ML methods for achieving high performance. In our experiments, we have many challenges in the dataset, such as missing values, categorical features, and class imbalance. There are needless features that may affect the selected ML method's performance. We applied different preparation techniques based on the literature and evaluated the selected ML methods based on permutations of various preprocessing and normalization techniques.
   - Imputation of missing values: Massive ToN-IoT dataset frequently contains missing values. These values must be treated correctly to construct a useful analysis. Imputation of missing values in the proposed model is replaced with the most common value in each feature, which contains missing values.
   - Converting categorical features into numerical: The ToN-IoT dataset contains several categorical features. The categorical features must be converted into numerical values. For this aim, one-hot encoding was employed. This work uses one-hot encoding to transform categorical features.
   - Class Imbalance: The SMOTE technique was used for balancing the attack class.
   - Timestamp, IP address, source port, and destination port features were dropped from the data matrix as they may cause overfitting to several ML methods in the training phase.

   The main phases applied during the feature-engineering development are preprocessing based on the discussed challenges and data normalization. The evaluations of ML methods are based on various feature-engineering techniques for achieving high performance:
   - **Class imbalance**: Class imbalance distributions plague the ToN-IoT dataset. Oversampling, undersampling, and hybrid approaches were offered as solutions to the unbalanced problem. Oversampling, which involves replicating the minority class points. Some researchers have used it. However, the disadvantage of this technique is that it overfits these points. Others use undersampling, which removes some points from the dominant class. The difficulty with this strategy is that some features that are eliminated may be crucial in representing the class. We employ a hybrid technique in which minority class points are duplicated and certain majority class points are removed. Synthetic minority oversampling technique (SMOTE) [39], [40] improves basic random oversampling by supplying synthetic minority class samples and addresses the overfitting issue that might arise with simple random oversampling. Because SMOTE generates new data points rather than duplicating existing ones. A linear combination of two comparable minority samples is utilized to produce additional minority data points. Between the minority sample and its nearest neighbors, new feature values are consistently interpolated.
   - **Feature selection**: To detect intrusions, feature selection is necessary. Getting a score for each prospective feature and selecting the optimal (k) features is the process of feature selection. The frequency of a feature is counted in training for each positive and negative class instance separately, then a function of both is obtained. For intrusion detection, various features must be checked, some of which will be beneficial, while others will be useless. The elimination of non-essential features improves accuracy, reduces computing time, and reduces the overfitting problem, resulting in improved performance. The used feature selection technique was Chi^2, which is a filter method. The Chi^2 technique was achieved better performance for many classification problems. The Chi^2 technique is a statistical approach for determining a score based on feature dependency. This feature selection strategy is used to exclude features that are not dependent on the class labels as others. By using the null hypothesis, it assumes that any two features are independent and searches for the most relevant features. A greater Chi^2 value means that the feature is more significant [41].

$$X^2 = \sum_{i=1}^{m} \sum_{j=1}^{n} \frac{(O_i - E_i)^2}{E_i} \tag{14}$$

where m reflects the number of features, n reflects the number of classes, O_i is the observed frequency, and E_i is the expected frequency.

- **Data normalization**: The ToN-IoT has features with various values and some features have bigger values than other features, which have smaller values. The different values that are out of range can lead to incorrect outcomes because a technique could be skewed toward features with higher values. Therefore, by scaling the feature vector, data normalization plays a vital role in avoiding outweighing features with higher values over features with lower values. Many normalization techniques are used, such as Min-max and standard scalar. Each one has its behavior on the feature vector. Min-Max is used for scaling feature values between [0:1] as presented in Eq. (19).

$$Z = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{15}$$

where x is the feature value, Z is the value after normalization, x_{max} *and* x_{min} are the maximum and minimum values of the feature.

2. **Training Procedure:**
All datasets are represented in CSV format, first, we separated the dataset into two sets, the first set contains training, and validation with 70% of the complete dataset, the second set contains the unseen test dataset for measuring the performance of the chosen ML methods. The splitting phase was done before applying any feature- engineering on the dataset to avoid data leakage, for further evaluation of the chosen ML methods, cross-validation with 5 folds was used to tune parameters in the training phase. The performance of chosen ML methods is assessed with various evaluation metrics, which will be presented in section VI. In Figure 4, we summarize the above phases concerned with assessing the performance of various ML methods using ToN-IoT datasets.
3. **Classifier performance evaluation:** several metrics were used for evaluating the efficiency of various ML methods based on the network ToN-IoT dataset. The selected evaluation methods were used as they give a full description of results for ML-based intrusion detection [42].

The first Metric is Accuracy, which presents the overall efficiency of a technique as the percentage of correctly classified instances as normal or attacks. The second Metric is precision which presents the percentage of correctly identified attacks out of all detected attacks. The third metric is recall which presents the percentage of the correctly detected attacks to the overall attacks in the test dataset. The fourth metric is the F1-score, which computes the weighted average of the accuracy and recall. The False-positive rate (FPR) is the number of normal instances, which is predictable as an attack instance on the total number of normal instances [42]. These selected metrics are described as follows:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{16}$$

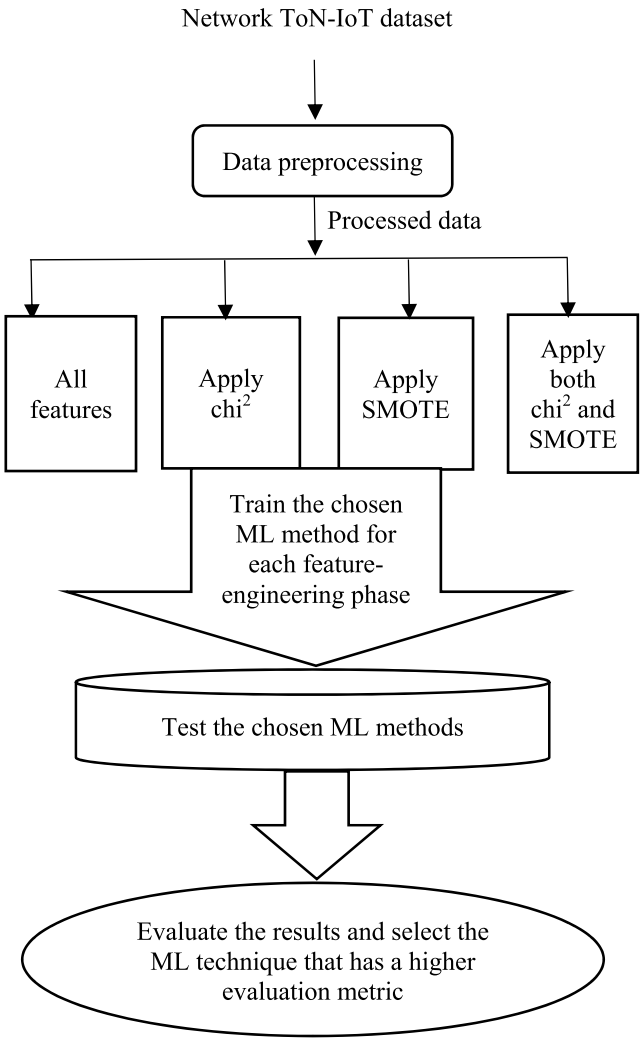

FIGURE 4. The process for evaluating ML methods on the ToN-IoT dataset.

$$\text{Precision} = \frac{TP}{TP + FP} \tag{17}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{18}$$

$$\text{F1}-score = 2 \times \frac{(precison * recall)}{(precision + recall)} \tag{19}$$

$$\text{FPR} = \frac{FP}{FP + TN} \tag{20}$$

where true positive (TP) is the total number of real attacks, which are properly classified as an attack. True negative (TN) is the total number of real normal records that are properly categorized as normal behavior. False negative (FN) is the total number of real attack samples, which are wrongly detected as normal. False positive (FP) is the total number of real normal samples that are wrongly identified as attacks.

VI. DATASET EVALUATION

This section addresses the efficiency of the used ML methods for VANTEs using the newly released ToN-IoT dataset. As discussed in earlier sections, we select the optimal

TABLE 1. The evaluation metric results for binary classification normal records versus attack records. Note: For each, the best metric value is indicated in red and bold font.

Models	Train Accuracy	Accuracy	Precision	Recall	F1-score	FPR	Confusion Matrix
LR	0.868	0.867	0.781	0.860	0.818	0.130	[[78334 11664] [6770 41545]]
NB	0.465	0.464	0.395	0.998	0.566	0.822	[[15982 74016] [84 48231]]
DT	0.981	0.980	0.960	0.982	0.971	0.022	[[88032 1966] [851 47464]]
RF	0.980	0.979	0.959	0.984	0.971	0.023	[[87949 2049] [792 47523]]
AdaBoost	0.910	0.909	0.827	0.935	0.878	0.105	[[82287 7470] [3127 45429]]
kNN	0.989	0.988	0.986	0.979	0.983	0.007	[[89325 673] [995 47320]]
SVM	0.869	0.868	0.784	0.859	0.820	0.127	[[78406 11351] [6844 41712]]
XGBoost	**0.992**	**0.991**	**0.984**	**0.991**	**0.987**	**0.009**	**[[89220 778] [442 47873]]**

TABLE 2. The evaluation metric results for binary classification normal records versus attack records using the Chi^2 technique, without using the SMOTE technique.

Models	Train Accuracy	Accuracy	Precision	Recall	F1-score	FPR	Confusion Matrix
LR	0.863	0.859	0.761	0.869	0.811	0.147	[[76787 13211] [6327 41988]]
NB	0.693	0.692	0.532	0.967	0.687	0.456	[[48922 41076] [1574 46741]]
DT	0.974	0.972	0.960	0.959	0.960	0.021	[[88069 1929] [1978 46337]]
RF	0.973	0.972	0.959	0.960	0.959	0.022	[[88005 1993] [1925 46390]]
AdaBoost	0.907	0.906	0.825	0.928	0.873	0.106	[[80465 9533] [3485 44830]]
kNN	0.984	0.982	0.983	0.965	0.974	0.009	[[89170 828] [1694 46621]]
SVM	0.861	0.860	0.760	0.874	0.813	0.148	[[76690 13308] [6065 42250]]
XGBoost	**0.984**	**0.983**	**0.984**	**0.967**	**0.975**	**0.008**	**[[89247 751] [1598 46717]]**

parameters mentioned in the literature [43], [44] and initialize our ML parameters with them for obtaining the best ML method performance. This work provides one of the first hands-on assessments of the selected ML methods when applied to the new network ToN-IoT datasets as a starting point for future research. The experiments behind this work were executed in Python version 3.8. The execution of all experiments was based on Windows 10 with a Core i7 and 16 GB of memory. The results used for evaluating the efficiency of the chosen ML methods using the network ToN-IoT dataset are described below.

- Binary classification

In this section, we present the results for the network ToN-IoT dataset. 5-fold cross-validation was applied to all selected ML methods. The accuracy, precision, recall, F1-score, and FPR are presented in addition to the confusion matrix to assess the chosen ML methods.

In general, XGBoost shows significant results based on various feature-engineering techniques that were applied to the dataset. Firstly, we impute missing values and convert categorical features using one-hot encoding. The number of features reached 108 features, and then we apply the Min-Max normalization technique. The results show that the training accuracy is 0.992, testing accuracy is 0.991, recall is 0.984, precision is 0.991, and F1-score is 0.987. In the case of FPR, the kNN shows significance with 0.007. Table 1 shows the results for all used ML methods with 108 features. In contrast, the second-best technique was the kNN method. The results of kNN show that the training accuracy is 0.989, testing accuracy is 0.988, recall is 0.986, precision is 0.979, and F1-score is 0.983. The worst method is naive Bayes (NB).

TABLE 3. The evaluation metric results for binary classification of normal records versus attack records using the SMOTE technique.

Models	Train Accuracy	Accuracy	Precision	recall	F1-score	FPR	Confusion Matrix
LR	0.869	0.863	0.762	0.883	0.818	0.148	[[76699 13299] [5634 42681]]
NB	0.588	0.464	0.395	0.998	0.566	0.822	[[15983 74015] [84 48231]]
DT	0.982	0.979	0.953	0.988	0.970	0.026	[[87635 2363] [582 47733]]
RF	0.982	0.979	0.951	0.989	0.970	0.027	[[87548 2450] [514 47801]]
AdaBoost	0.923	0.911	0.816	0.961	0.883	0.116	[[79516 10482] [1863 46452]]
kNN	0.991	0.990	0.981	0.990	0.985	0.001	[[89067 931] [504 47811]]
SVM	0.870	0.864	0.764	0.883	0.819	0.147	[[76788 13210] [5652 42663]]
XGBoost	0.993	0.990	0.976	0.997	0.986	0.013	[[88789 1209] [125 48190]]

TABLE 4. The evaluation metric results for binary classification normal records versus attack records using the Chi^2 technique and the SMOTE technique.

Models	Train Accuracy	Accuracy	Precision	Recall	F1-score	FPR	Confusion Matrix
LR	0.865	0.860	0.760	0.873	0.813	0.148	[[76690 13308] [6069 42246]]
NB	0.756	0.692	0.532	0.967	0.687	0.456	[[48918 41080] [1573 46742]]
DT	0.976	0.971	0.933	0.987	0.959	0.038	[[86576 3422] [633 47682]]
RF	0.976	0.971	0.932	0.988	0.959	0.039	[[86502 3496] [579 47736]]
AdaBoost	0.908	0.897	0.803	0.935	0.864	0.123	[[78900 11098] [3136 45179]]
kNN	**0.985**	**0.982**	**0.959**	**0.989**	**0.974**	**0.023**	**[[87960 2038] [515 47800]]**
SVM	0.865	0.860	0.760	0.874	0.813	0.148	[[76690 13308] [6066 42249]]
XGBoost	0.986	0.982	0.954	0.996	0.975	0.026	[[87670 2328] [190 48125]]

The variety of data in ToN-IoT datasets might explain these variations in the ML technique's performance. RF and DT are nearly the same results.

After testing ML methods on the whole dataset. The Chi^2 was applied as a feature selection technique. After evaluating ML methods based on a different number of features using Chi^2. We select 20 features from all 108 features since the best evaluation metric is obtained with only 20 features. XGBoost shows significant results after Chi2, nearly the same as testing with all features. The results show that the training accuracy is 0.984, testing accuracy is 0.983, recall is 0.984, precision is 0.967, F1-score is 0.975, and FPR is 0.008.

Table 2 shows the results for all used ML methods with the Chi^2 feature selection technique. The second-best technique was the kNN method. The results of the kNN show that the training accuracy is 0.990, testing accuracy is 0.988, recall is 0.984, precision is 0.982, F1-score is 0.983, and FPR is 0.009. The worst method is NB. The Chi^2 method obtains the best result with NB compared with the whole dataset.

Since ToN-IoT suffers from class imbalance problem, another testing methodology was done based on SMOTE technique, XGBoost and kNN have the same best result with 0.990 accuracies. In the case of other evaluation metrics, for XGBoost, recall is 0.976, precision is 0.997, F1-score is 0.986, and FPR is 0.013. For kNN, recall is 0.981, precision is 0.990, F1-score is 0.985, and FPR has the best metric compared to other ML methods with 0.001. XGBoost is better than kNN since XGBoost has less training and testing time. Table 3 shows the results for all used ML methods using SMOTE technique.

Finally, based on binary classification, we evaluate the selected ML methods based on Chi^2 and SMOTE techniques.

TABLE 5. The evaluation metric results for multi-class classification normal records versus different attack records.

Models	Train Accuracy	Accuracy	Precision	Recall	F1-score	FPR
LR	0.777	0.777	0.777	0.777	0.777	0.046
NB	0.711	0.712	0.712	0.712	0.712	0.136
DT	0.938	0.934	0.934	0.934	0.934	0.022
RF	0.940	0.937	0.937	0.937	0.937	0.021
AdaBoost	0.399	0.399	0.399	0.399	0.399	0.505
kNN	0.981	0.979	0.979	0.979	0.979	0.009
SVM	0.779	0.780	0.780	0.780	0.780	0.046
XGBoost	**0.986**	**0.983**	**0.983**	**0.983**	**0.983**	**0.008**

TABLE 6. The evaluation metric results for multi-class classification using the Chi^2 technique without using the SMOTE technique.

Models	Train Accuracy	Accuracy	Precision	Recall	F1-score	FPR
LR	0.759	0.759	0.759	0.759	0.759	0.049
NB	0.709	0.710	0.710	0.710	0.710	0.113
DT	0.933	0.928	0.928	0.928	0.928	0.031
RF	0.935	0.931	0.931	0.931	0.931	0.029
AdaBoost	0.498	0.497	0.497	0.497	0.497	0.424
kNN	0.980	0.977	0.977	0.977	0.977	0.014
SVM	0.760	0.761	0.761	0.761	0.761	0.048
XGBoost	**0.985**	**0.982**	**0.982**	**0.982**	**0.982**	**0.009**

TABLE 7. The evaluation metric results for multi-class classification using the SMOTE technique.

Models	Train Accuracy	Accuracy	Precision	Recall	F1-score	FPR
LR	0.668	0.571	0.571	0.571	0.571	0.477
NB	0.626	0.558	0.558	0.558	0.558	0.475
DT	0.882	0.913	0.913	0.913	0.913	0.055
RF	0.877	0.911	0.911	0.911	0.911	0.058
AdaBoost	0.417	0.238	0.238	0.238	0.238	0.908
kNN	0.976	0.976	0.976	0.976	0.976	0.018
SVM	0.770	0.753	0.753	0.753	0.753	0.049
XGBoost	**0.980**	**0.979**	**0.979**	**0.979**	**0.979**	**0.018**

kNN shows significant results with Chi^2 and SMOTE. kNN training accuracy is 0.985, testing accuracy is 0.982, recall is 0.959, precision is 0.989, F1-score is 0.974, and FPR is 0.023. Table 4 shows the results for all used ML methods with the Chi^2 technique, and the SMOTE technique.

- **Multi-class classification:**

As declared earlier in Section 3, the network ToN-IoT dataset has a feature type that shows the attack sub-category for multi-class classification problems. ToN-IoT has 10 sub-classes. In this section, we evaluate candidate's ML methods for testing multi-classification problems to assess their performance. Multi-class classification problem necessitates various considerations while evaluating candidate ML methods. To begin with, LR is frequently used to handle binary classification problems and cannot be used directly in multi-class classification problems. As a result, the one-vs-rest (OvR) technique is used to construct LR for multi-class classification. The evaluation metrics used to compare all models are accuracy, precision, recall, F-score, and confusion matrix. Table 5 summarizes the multi-classification findings for all 108 features.

XGBoost attains good results compared to the other ML methods. The training accuracy for XGBoost is 0.986, FPR is 0.008, and 0.983 for all other metrics, kNN achieves the second-top results. The scores are 0.981 for training accuracy and 0.979 for all other metrics. The AdaBoost classifier has the worst metrics compared to all tested ML methods. In terms of training and testing time, SVM takes the longest.

For multi-class classification problems, after testing all selected ML methods in the whole dataset, we apply the Chi^2 feature selection technique. We evaluate all ML methods using Chi^2 with the optimal 20 features from all 108 features. Such as binary classification, XGBoost shows significant results. The training accuracy is 0.985; FPR is 0.008, and 0.982 for all other metrics. Table 6 shows the results for all used ML methods with the Chi^2 technique. The second-best technique was kNN; the training accuracy for kNN is 0.980. The testing accuracy, recall, precision, and F1-score are all 0.977. The worst model is AdaBoost with a training accuracy of 0.498 and 0.497 for all other metrics.

TABLE 8. The evaluation metric results for multi-class classification using the Chi^2 technique and SMOTE technique.

Models	Train Accuracy	Accuracy	Precision	Recall	F1-score	FPR
LR	0.609	0.509	0.509	0.509	0.509	0.542
NB	0.513	0.414	0.414	0.414	0.414	0.624
DT	0.871	0.875	0.875	0.875	0.875	0.109
RF	0.870	0.875	0.875	0.875	0.875	0.110
AdaBoost	0.436	0.336	0.336	0.336	0.336	0.750
kNN	0.971	0.976	0.976	0.976	0.976	0.019
SVM	0.751	0.747	0.747	0.747	0.747	0.050
XGBoost	**0.980**	**0.978**	**0.978**	**0.978**	**0.978**	**0.019**

Another testing methodology was employed based on the SMOTE technique. As presented in Table 7 XGBoost has the best results compared to other used ML methods. Table 7 shows the results for all used ML methods with the SMOTE technique.

Finally, based on the multi-class classification problem, we evaluate the selected ML methods based on the Chi^2 and SMOTE techniques. XGBoost shows significant results with Chi^2 and SMOTE techniques. XGBoost training accuracy is 0.980, FPR is 0.019, and 0.978 for all other metrics. Table 8 shows the results for all used ML methods with Chi^2 technique, and SMOTE technique.

VII. CONCLUSION

This article presented IDS for VANETs based on the network ToN-IoT dataset. Our model includes many basic elements, and the used dataset for training and testing was the network records for ToN-IoT. The network records of the ToN-IoT dataset have a class imbalance problem and missing values. Using ToN-IoT, our work can cover multiple attacks more than previous attacks in outdated datasets such as KDD-CUP99, NSL-KDD, and UNSW-NB15. Therefore, several system blocks are exploring, preprocessing, feature selection, class imbalance solution, training ML methods, and the last block is testing ML methods. The Chi^2 technique was used for feature selection. It reduced the number of features to 20 features, resulting in faster training time, less complexity of our model, and the best performance compared to the whole dataset. The SMOTE technique was used for class balancing. It reduced the bias of the dominant class, reduced overfitting, and gave a good performance. The combination of $Chi^2$and SMOTE as preprocessing techniques led us to a good performance. Several evaluation metrics (i.e., accuracy, precision, recall, F1-score, FPR, and confusion matrix) were used for assessing the performance of the used ML methods. After evaluating the selected ML methods, we concluded that XGBoost has the best performance in binary classification and multi-class classification problems compared to all other ML methods. In future work, we plan to deploy a model, which obtains the best metrics in Apache spark and Kafka Hadoop. Deep learning methods will apply to the ToN-IoT dataset and will use optimization algorithms for dimensionality reduction.

REFERENCES

[1] A. Alsarhan, A. Y. Al-Dubai, G. Min, A. Y. Zomaya, and M. Bsoul, "A new spectrum management scheme for road safety in smart cities," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 11, pp. 3496–3506, Nov. 2018.
[2] L. Wang, Z. Chen, and J. Wu, "An opportunistic routing for data forwarding based on vehicle mobility association in vehicular ad hoc networks," *Informatio*, vol. 8, no. 4, p. 140, 2017.
[3] J. Cui, L. S. Liew, G. Sabaliauskaite, and F. Zhou, "A review on safety failures, security attacks, and available countermeasures for autonomous vehicles," *Ad Hoc Netw.*, vol. 90, Jul. 2019, Art. no. 101823.
[4] G. Loukas, T. Vuong, R. Heartfield, G. Sakellari, Y. Yoon, and D. Gan, "Cloud-based cyber-physical intrusion detection for vehicles using deep learning," *IEEE Access*, vol. 6, pp. 3491–3508, 2018.
[5] K. M. A. Alheeti, A. Gruebler, and K. McDonald-Maier, "Intelligent intrusion detection of grey hole and rushing attacks in self-driving vehicular networks," *Computers*, vol. 5, no. 3, p. 16, 2016.
[6] M. W. P. Maduranga and R. Abeysekera, "Machine learning applications in IoT based agriculture and smart farming: A review," *Int. J. Eng. Appl. Sci. Technol.*, vol. 4, no. 12, pp. 24–27, May 2020.
[7] A. Alshammari, M. A. Zohdy, D. Debnath, and G. Corser, "Classification approach for intrusion detection in vehicle systems," *Wireless Eng. Technol.*, vol. 9, no. 4, pp. 79–94, 2018.
[8] N. Moustafa. (2020). *ToN-IoT Dataset*. [Online]. Available: https://cloudstor.aarnet.edu.au/plus/s/ds5zW91vdgjEj9i
[9] W. Li, P. Yi, Y. Wu, L. Pan, and J. Li, "A new intrusion detection system based on KNN classification algorithm in wireless sensor network," *J. Elect. Comput. Eng.*, vol. 2014, no. 5, pp. 1–8, 2014.
[10] N. Farnaaz and M. A. Jabbar, "Random forest modeling for network intrusion detection system," *Proc. Comput. Sci.*, vol. 89, pp. 213–217, May 2016.
[11] S. A. Mulay, P. R. Devale, and G. V. Garje, "Intrusion detection system using support vector machine and decision tree," *Int. J. Comput. Appl.*, vol. 3, no. 3, pp. 40–43, Jun. 2010.
[12] N. Ding, Y. Liu, Y. Fan, and D. Jie, "Network attack detection method based on convolutional neural network," in *Proc. Chin. Intell. Syst. Conf.* in Lecture Notes in Electrical Engineering. Singapore: Springer, 2020, pp. 610–620.
[13] S. Nayyar, S. Arora, and M. Singh, "Recurrent neural network based intrusion detection system," in *Proc. Int. Conf. Commun. Signal Process. (ICCSP)*, Jul. 2020, pp. 136–140.
[14] H. Zhang, S. Dai, Y. Li, and W. Zhang, "Real-time distributed-random-forest-based network intrusion detection system using apache spark," in *Proc. IEEE 37th Int. Perform. Comput. Commun. Conf. (IPCCC)*, Nov. 2018, pp. 1–7.
[15] R. G. Engoulou, M. Bellaïche, S. Pierre, and A. Quintero, "VANET security surveys," *Comput. Commun.*, vol. 44, pp. 1–13, May 2014.
[16] J. Zhang, M. Zulkernine, and A. Haque, "Random-forests-based network intrusion detection systems," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 38, no. 5, pp. 649–659, Sep. 2008.
[17] D. Huang, S. Misra, M. Verma, and G. Xue, "PACP: An efficient pseudonymous authentication-based conditional privacy protocol for VANETs," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 3, pp. 736–746, Sep. 2011.
[18] J. K. Liu, T. H. Yuen, M. H. Au, and W. Susilo, "Improvements on an authentication scheme for vehicular sensor networks," *Expert Syst. Appl.*, vol. 41, no. 5, pp. 2559–2564, 2014.

[19] W. Li and H. Song, "ART: An attack-resistant trust management scheme for securing vehicular ad hoc networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 4, pp. 960–969, Apr. 2016.
[20] N. K. Chaubey, "Security analysis of vehicular ad hoc networks (VANETs): A comprehensive study," *Int. J. Secur. Its Appl.*, vol. 10, no. 5, pp. 261–274, May 2016.
[21] N. Kumar and N. Chilamkurti, "Collaborative trust aware intelligent intrusion detection in VANETs," *Comput. Elect. Eng.*, vol. 40, no. 6, pp. 1981–1996, 2014.
[22] A. Daeinabi, A. G. P. Rahbar, and A. Khademzadeh, "VWCA: An efficient clustering algorithm in vehicular ad hoc networks," *J. Netw. Comput. Appl.*, vol. 34, no. 1, pp. 207–222, 2011.
[23] A. Khraisat, I. Gondal, P. Vamplew, J. Kamruzzaman, and A. Alazab, "Hybrid intrusion detection system based on the stacking ensemble of C5 decision tree classifier and one class support vector machine," *Electronics*, vol. 9, no. 1, p. 173, Jan. 2020.
[24] C. Yin, S. Huang, P. Su, and C. Gao, "Secure routing for large-scale wireless sensor networks," in *Proc. Int. Conf. Commun. Technol. (ICCT)*, Apr. 2003, pp. 1282–1286.
[25] O. Y. Al-Jarrah, A. Siddiqui, M. Elsalamouny, P. D. Yoo, S. Muhaidat, and K. Kim, "Machine-learning-based feature selection techniques for large-scale network intrusion detection," in *Proc. IEEE 34th Int. Conf. Distrib. Comput. Syst. Workshops*, Jun. 2014, pp. 177–181.
[26] M. S. Rani and S. B. Xavier, "A hybrid intrusion detection system based on C5. 0 decision tree algorithm and one-class SVM with CFA," *Int. J. Innov. Res. Comput. Commun. Eng.*, vol. 3, no. 6, pp. 5526–5537, Jun. 2015.
[27] N. B. Amor, S. Benferhat, and Z. Elouedi, "Naive Bayes vs decision trees in intrusion detection systems," in *Proc. ACM Symp. Appl. Comput. (SAC)*, 2004, pp. 420–424.
[28] G. Kim, S. Lee, and S. Kim, "A novel hybrid intrusion detection method integrating anomaly detection with misuse detection," *Expert Syst. Appl.*, vol. 41, no. 4, pp. 1690–1700, 2014.
[29] W. L. Al-Yaseen, Z. A. Othman, and M. Z. A. Nazri, "Multi-level hybrid support vector machine and extreme learning machine based on modified k-means for intrusion detection system," *Expert Syst. Appl.*, vol. 67, pp. 296–303, Jan. 2017.
[30] Y. Zeng, M. Qiu, Z. Ming, and M. Liu, "Senior2Local: A machine learning-based intrusion detection method for VANETs," in *Smart Computing and Communication* (Lecture Notes in Computer Science). Cham, Switzerland: Springer, 2018, pp. 417–426.
[31] C. Ioannou and V. Vassiliou, "An intrusion detection system for constrained WSN and IoT nodes based on binary logistic regression," in *Proc. 21st ACM Int. Conf. Modeling, Anal. Simulation Wireless Mobile Syst.*, Oct. 2018, pp. 259–263.
[32] S. Mukherjee and N. Sharma, "Intrusion detection using naive Bayes classifier with feature reduction," *Proc. Technol.*, vol. 4, pp. 119–128, Feb. 2012.
[33] M. Almseidin, M. Alzubi, S. Kovacs, and M. Alkasassbeh, "Evaluation of machine learning algorithms for intrusion detection system," in *Proc. IEEE 15th Int. Symp. Intell. Syst. Informat. (SISY)*, Sep. 2017, pp. 277–282.
[34] B. Ingre, A. Yadav, and A. K. Soni, "Decision tree based intrusion detection system for NSL-KDD dataset," in *Information and Communication Technology for Intelligent Systems (ICTIS)*, vol. 2. Cham, Switzerland: Springer, 2018, pp. 207–218.
[35] P. Negandhi, Y. Trivedi, and R. Mangrulkar, "Intrusion detection system using random forest on the NSL-KDD dataset," in *Emerging Research in Computing, Information, Communication and Applications*. Singapore: Springer, 2019, pp. 519–531.
[36] M. Mohammadi, T. A. Rashid, S. H. T. Karim, A. H. M. Aldalwie, Q. T. Tho, M. Bidaki, A. M. Rahmani, and M. Hosseinzadeh, "A comprehensive survey and taxonomy of the SVM-based intrusion detection systems," *J. Netw. Comput. Appl.*, vol. 178, Mar. 2021, Art. no. 102983.
[37] Y. Zhou, T. A. Mazzuchi, and S. Sarkani, "M-AdaBoost—A based ensemble system for network intrusion detection," *Expert Syst. Appl.*, vol. 162, Dec. 2020, Art. no. 113864.
[38] H. Jiang, Z. He, G. Ye, and H. Zhang, "Network intrusion detection based on PSO-XGBoost model," *IEEE Access*, vol. 8, pp. 58392–58401, 2020.
[39] J. Wang, M. Xu, H. Wang, and J. Zhang, "Classification of imbalanced data by using the SMOTE algorithm and locally linear embedding," in *Proc. 8th Int. Conf. Signal Process.*, Nov. 2006, pp. 1–4.
[40] S. Bagui and K. Li, "Resampling imbalanced data for network intrusion detection datasets," *J. Big Data*, vol. 8, no. 1, pp. 1–41, Dec. 2021.
[41] H. Liu and R. Setiono, "Chi2: Feature selection and discretization of numeric attributes," in *Proc. 7th IEEE Int. Conf. Tools with Artif. Intell.*, Nov. 2002, pp. 388–391.
[42] M. Haggag, M. M. Tantawy, and M. M. S. El-Soudani, "Implementing a deep learning model for intrusion detection on apache spark platform," *IEEE Access*, vol. 8, pp. 163660–163672, 2020.
[43] J. Han, J. Pei, and M. Kamber, *Data Mining: Concepts Techninqs*. Amsterdam, The Netherlands: Elsevier, 2011.
[44] A. Alsaedi, N. Moustafa, Z. Tari, A. Mahmood, and A. Anwar, "TON_IoT telemetry dataset: A new generation dataset of IoT and IIoT for data-driven intrusion detection systems," *IEEE Access*, vol. 8, pp. 165130–165150, 2020.

ABDALLAH R. GAD received the B.S. degree in electronics and communications engineering from the Misr University for Science and Technology, Giza, Egypt, in 2013, and the M.Sc. degree in communications and electronics engineering from Fayoum University, Faiyum, Egypt, in 2017, where he is currently pursuing the Ph.D. degree. He is currently working as a Teaching Assistant at the Department of Communication and Electronics Engineering, October High Institute for Engineering and Technology, 6th of October City, Egypt. His research interests include machine learning, deep learning, and network security.

AHMED A. NASHAT received the B.Sc. degree in communication and electronics engineering from Cairo University, Egypt, in 1982, the M.Sc. degree in communication and electronics engineering from the King Fahd University of Petroleum and Minerals, Saudi Arabia, in 1985, and the Ph.D. degree in communication engineering from New Mexico State University, Las Cruces, NM, USA, in 1990. From 1990 to 1995, he worked with Telstra Research Laboratories, Australia. He is currently working as an Associate Professor at Fayoum University, Egypt. His research interests include application of digital signal processing techniques to spectral analysis, smart antennas, signal detection and estimation theory, radar and sonar systems, pattern recognition analysis, image processing, digital filters, and adaptive noise cancellation.

TAMER M. BARKAT received the B.Sc. and M.Sc. degrees in communication and computer engineering from Helwan University, Egypt, in 2000 and 2004, respectively, and the Ph.D. degree in communication engineering from Cairo University, Egypt, in 2008. He is currently working as an Associate Professor at Fayoum University, Egypt. His research interests include mobile and communication networks, digital communications, security of computer networks, wireless sensor networks, intrusion detection systems, and wireless networks.