IEEE Open Journal of
Vehicular Technology

Received 4 December 2021; accepted 19 December 2021. Date of publication 24 December 2021;
date of current version 17 January 2022. The review of this article was coordinated by Editor Rongxing Lu.

*Digital Object Identifier 10.1109/OJVT.2021.3138354*

# Machine Learning Based Misbehaviour Detection in VANET Using Consecutive BSM Approach

**AEKTA SHARMA AND ARUNITA JAEKEL (Member, IEEE)**

School of Computer Science, University of Windsor, Windsor, ON N9B 3P4, Canada

CORRESPONDING AUTHOR: ARUNITA JAEKEL (e-mail: arunita@uwindsor.ca)

The work of Arunita Jaekel was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC). This work was supported by (NSERC) DG, under Grant# RGPIN-2015-05641.

**ABSTRACT** Vehicular ad-hoc network (VANET) is an emerging technology for vehicle-to-vehicle communication vital for reducing road accidents and traffic congestion in an Intelligent Transportation System (ITS). VANET communication is vulnerable to various attacks and cryptographic techniques are commonly used for message integrity and authentication of vehicles. However, cryptograhpic techniques alone may not be sufficient to protect against insider attacks. Many VANET safety applications rely on periodic broadcast of basic safety messages (BSMs) from surrounding vehicles that contain important status information about a vehicle such as its position, speed, and heading. If an attacker (misbehaving vehicle) injects false position information in a BSM, it can lead to serious consequences including traffic congestion or even accidents. Therefore, it is imperative to accurately detect and identify such attackers to ensure safety in the network. This paper presents a novel data-centric approach to detect *position falsification* attacks, using machine learning (ML) algorithms. Unlike existing techniques, the proposed approach combines information from 2 consecutive BSMs for training and testing. Simulations using the Vehicular Reference Misbehavior (VeReMi) dataset demonstrate that the proposed model clearly outperforms existing approaches for identifying a range of different attack types.

**INDEX TERMS** Misbehavior detection, machine learning, position falsification attack, vehicular ad-hoc network, vehicular communication.

## I. INTRODUCTION

According to the 2018 Global status report on road safety by the World Health Organisation (WHO), road accidents are the leading cause of death for children and young adults aged 5-29 years [1]. Vehicular ad hoc networks (VANETs) [2] form an integral part of future Intelligent Transportation System (ITS) [3] designed to create a safe and efficient transportation network, through secure and reliable communication among various network components. A VANET architecture, as shown in Fig. 1, consists of different types of nodes, including vehicles, road-side units (RSUs), and other infrastructure nodes. The infrastructure nodes provide different services to the participating vehicles. For example, RSUs can facilitate communication between the nodes, and Central Authority/Authorization Party, provides support such as registering a node in the network and revoking access in case of misbehaviour [4]. Vehicles in the network are equipped with On-Board Units (OBUs), with processing and wireless communication capabilities for secure information exchange. Although VANET is a highly dynamic, ad hoc wireless network, the infrastructure nodes, including RSUs, are often also connected using a backbone wired network to facilitate communication among geographically distant nodes.

VANET communication provides tremendous opportunities for improving vehicle safety and enhancing comfort and convenience of both drivers and passengers [5]. Safety applications, e.g., blind-spot warnings, collision avoidance, etc., require situation awareness and rely on up-to-date information on the status of surrounding vehicles. To support such applications, each vehicle broadcasts its status to neighboring

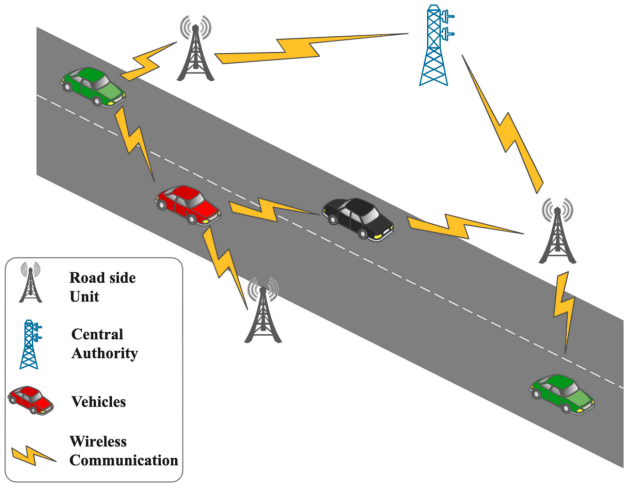



**FIGURE 1.** An example of Vehicular ad-hoc network.

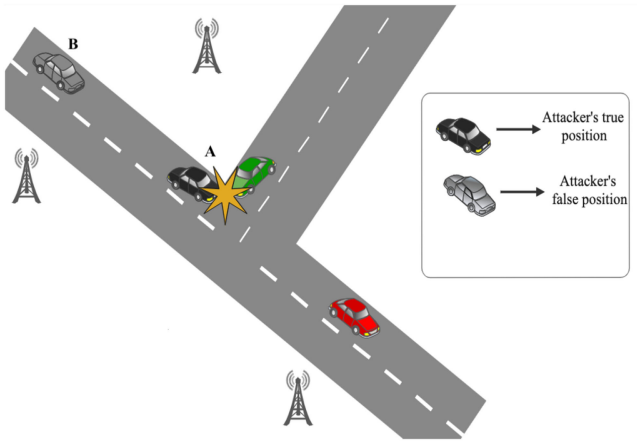



**FIGURE 2.** An example of Position Falsification Attack.

vehicles in the form of periodic beacons called *basic safety messages* (BSMs). BSMs contain important information about a vehicle, including its current position, speed, direction, as well as a timestamp indicating when the message was generated. However, BSM transmissions are vulnerable to various security attacks [6], [7] that can have severe negative consequences including personal injury and even death, if safety-critical applications are affected. Therefore, it is extremely important to ensure that the information contained in the BSMs are accurate, timely and unaltered and cryptographic techniques are employed to authenticate users and protect such communications [8].

Due to the time-sensitive nature of the information in a BSM and the requirements for fast processing, the BSM contents are typically not encrypted. But each BSM is digitally signed by the sender to ensure that the contents have not been altered and that the sender is a legitimate node in the network. However, such cryptographic techniques are insufficient to protect against *insider* attacks, where the misbehaving vehicle has valid credentials to access the network. This can occur, for example, if a valid vehicle is somehow compromised so an attacker is able to use its credentials.

A compromised vehicle, with valid credentials, can insert false information in a BSM and then digitally sign the packet. One example of this is a *position falsification attack*, where a malicious vehicle inserts incorrect position information in its BSM. As shown in Fig. 2, such attacks can lead to serious consequences including collisions. In Fig. 2, the *actual* position of a malicious vehicle $v_1$ is at location A, but its *reported* position in the BSM indicates it is at location B. This wrong position can lead to an accident with another vehicle $v_2$, since $v_2$ will mistakenly believe it is at a safe distance from $v_1$.

In this paper, we present a novel machine-learning (ML) [9] based approach for automatically detecting position falsification attacks in BSMs. Unlike existing approaches for detecting such attacks, which use features of a *single* BSM for training, we have created an augmented feature set by combining information from successive BSMs. We would like to note that the main contribution of the paper is not in designing new ML algorithms. Rather, it is the idea of using information from *consecutive* BSMs and showing that this allows more accurate attack detection compared to existing approaches, for the same dataset and even using some of the same ML algorithms. The main contributions of this paper are:

- A new ML-based approach to accurately detect different types of position falsification attacks.
- A hierarchical architecture, where the RSUs are responsible for misbehavior detection, rather than individual vehicle OBUs with limited space and computing resources.
- Evaluation and performance comparison for different classification algorithms such as K-Nearest Neighbor (KNN), Naive Bayes etc. [10]
- Improved performance compared to existing ML-based approaches for detecting position falsification attacks.

The remainder of the paper is organized as follows. In Section II, we briefly review VANET security and existing techniques for misbehavior detection, including machine learning based techniques. In Section III, we present our proposed architecture and misbehavior detection framework. We evaluate the proposed approach and compare its performance with existing techniques in Sections IV and V, and discuss our conclusions and some directions for future work in Section VI.

## II. OVERVIEW OF VANET MISBEHAVIOR DETECTION

VANET communications can be vulnerable to various types of attacks [7], [11] from different types of attackers. Based on the nature of the attack and motivation of the attacker, they are typically classified as [12]:

1) Insider vs. Outsider Attacker: Insider attackers are those who are authenticated members of the network, while outside attackers are those who are not authorized.

2) Active vs. Passive Attacker: Active attackers take part in the attack by directly participating in the attack, such as altering the message or destroying the message packet in the network. Passive attackers listen to the conversation in the network without interfering directly and may use the information for malicious purposes.
3) Malicious vs. Rational Attacker: The goal of a malicious attacker is often to cause damage to the network itself, while rational attackers trigger the attack for personal gain.

In this paper we focus on *active* attacks carried out by *insider* attackers, whose motivation may be malicious or rational. Accurate detection of different types of attacks in VANET [6] is a challenging task. Most intrusion detection approaches can be divided into *node-centric detection*, where the detection of misbehaviour depends on the credibility of the sender or *data-centric detection*, where detection is based on contents of the message [13].

In [14], the authors introduced a framework called Maat that uses subjective logic to build a fusion and data management system to determine the trustworthiness of data. The authors used four comparison checks for performance evaluation of the model, namely Acceptance Range Threshold (ART), Sudden Appearance Warning (SAW), Simple Speed Check (SSC), and Distance Moved Verifier (DMV). The framework was evaluated using the VeReMi dataset, which the authors generated through simulations. In [15], the authors proposed integrating plausibility checks and a machine learning framework for misbehaviour detection using the sender-receiver pair approach in the VeReMi dataset. They added six features, including two plausibility checks capable of detecting fake location and four quantitative metrics used to describe vehicle's behaviour in the network.

The authors of [16] proposed an intrusion detection method for vehicular networks based on the survival analysis model. The authors' main aim was to identify malicious (Controller Area Network) CAN messages and accurately detect the normality and abnormality of a vehicle network without semantic knowledge of the CAN ID function. According to the authors, a CAN ID with a longer cycle decreases detection accuracy, while the number of CAN IDs impacts detection speed.

In [17], Xue *et al.* proposed using a trusted neighbour table (TNT) to detect position spoofing attacks. It is a location verification scheme, where each vehicle maintains a TNT that contains its neighbouring nodes' latest location. The use of TNT is different from simple list of neighbours, as TNT contents are authenticated. In paper [18], authors proposed Intrusion Detection System which can efficiently identify a fake information attack using statistical techniques, as well as other forms of attacks without relying on trust or reputation scores. The algorithm is based on the idea that neighboring vehicles will also experience the similar levels of traffic flow. So, when a vehicle receives a flow value which does not match with the other vehicles, it is rejected, and vehicle ID is reported. The intrusion is easier to detect if the bogus data differs significantly from the computed data; otherwise, it is considerably more difficult to detect using this approach.

### A. ML-BASED MISBEHAVIOR DETECTION

Machine learning (ML) is a branch of Artificial Intelligence that has been used in diverse fields, such as healthcare, e-commerce, facial recognition etc., to improve the performance of specific tasks [19], [20] and can also help to improve the security of a highly dynamic vehicular network [21]. It is a data-centric approach that aims to optimize network performance by reducing the vulnerabilities of the network. One important application has been to correctly identify legitimate vehicles and misbehaving nodes, using supervised classification algorithms [22], such as *K-Nearest Neighbour* algorithm [23], *Decision Tree* algorithm [24], *Random Forest* algorithm [25], and *Naïve Bayes* classification algorithm [26].

There are two main types of classification:

- Binary classification, where relevant items from a dataset are classified as belonging to one of two possible classes. In this paper, the two classes in binary classification are legitimate vehicles and attacker vehicles.
- Multiclass classification, where classification involves classifying into more than two classes in a dataset. For example, we consider five different position falsification attacks and the ML model should not only identify a misbehaving vehicle as "attacker" but determine the type of attack being carried out.

In this section, we discuss some recent approaches for ML-based misbehavior detection in VANET. One of the earliest examples is [27], where Grover *et al.* introduced an ensemble learning-based approach for classifying honest and misbehaving vehicles. The authors used different classification algorithms, including Naïve Bayes, Instance-based learner, Random Forest, Decision Tree and AdaBoost and combined the results from different classifiers using majority rule to classify the vehicles individually.

In [28] the authors used ML algorithms to detect wormhole attacks, which is a type of routing attack, where packets received by a node are tunnelled to another node and then replayed in the network. The models were implemented using support vector machine (SVM) and KNN and showed promising results on a datatset generated by the authors.

A number of recent papers have used ensemble learning [29], a ML technique that combines several base models to produce an improved predictive model, for misbehavior detection in VANET. The work in [27] used ensemble learning to identify different types of misbehavior including identity spoofing, position forging, packet and packet replay. In [30], the authors proposed a hybrid context-aware misbehavior detection system (EHCA-MDS) that combined several non-parametric, unsupervised-based online statistical classifiers with a supervised classifier model. These classifiers worked together to detect the different types of misbehaving vehicles that share false mobility messages. The work in [31] implemented an on-demand collaborative intrusion detection system (MA-CIDS) for misbehavior detection in VANET, using

ensemble learning. Finally, [32] used stacking to to improve the classification accuracy for detecting position falsification attacks. Their model was trained and evaluated using the VeReMi dataset [14] and achieves an overall accuracy of 98%.

Khot *et al.* [33] proposed a machine learning framework to predict the next position of the vehicle in the network and compared the predicted values with the positions reported in the BSMs. If there was sufficient discrepancy between the calculated and reported positions, the sending vehicle was classified as an attacker. The authors considered several ML algorithms and found that Random Forest performed best compared to other algorithms. In paper [34], authors used SVM and Logistic Regression algorithms to detect position falsification attacks based on different combinations of features/predictors, such as the sender vehicle's position, speed, and any change in position or speed of sender vehicle.

The work in [37] used Support Vector Machines with Modified Fading Memory (SVM-MFM) to detect misbehavior in VANET messages. This approach provided a feasible solution that reduced high computational cost for RSUs. In [34], the authors analyzed safety messages from vehicles to detect incorrect position information inserted by misbehaving nodes. They used supervised learning algorithms including SVM and logistic regression to detect various position falsification attacks. The work in [35] applied machine learning techniques to test the received power coherency metric, which was used as a misbehaving detection metric along with the vehicle position. The authors combined trust value computation with KNN classifier to detect false position coordinates in vehicle messages.

In recent research by Gyawali *et al.* [36], the authors introduced a misbehaviour detection model for both false alert verification scheme and position falsification attack based on the sender-receiver pair approach. The false alerts could include hazard condition notification, emergency vehicle stopping warning or emergency braking of a vehicle. The receiver vehicle used the sender vehicle's speed, position, receiving distance and RSSI value as features in the dataset. Different machine learning algorithms, including K-Nearest Neighbor, Decision Tree and Random Forest were used to train and test the model using this dataset.

In [38], the authors proposed a hybrid Intrusion Detection System to improve the accuracy and performance using Artificial Neural Networks. The performance of the detection system was evaluated using two scenarios: misuse and anomaly. The proposed approach achieved higher accuracy and precision and lower false alarm rates when detecting malicious nodes. In [39], the authors used SVM and Naïve Bayes feature embedding as an intrusion detection framework. They implemented Naïve Bayes feature transformation technique on original features to obtain new high quality data and also implemented the framework on various datasets. The research in [40] proposed ReFIoV, a novel reputation framework for information-centric vehicular applications based on machine learning and the artificial immune system (AIS). According to the authors, ReFIoV outperformed state-of-the-art reputation systems by reducing the number of incorrectly labelled misbehaving nodes, and requiring lower overhead and detection time.

**TABLE I. Position Falsification Attack Detection Using Machine Learning by Other Researchers**

| Paper Title | Dataset used | Machine Learning Algorithm Used |
|---|---|---|
| Machine Learning Based Approach to Detect Position Falsification Attack in VANETs [34] | VeReMi dataset | SVM and Logistic Regression |
| A New Combination of Machine Learning Algorithms using Stacking Approach for Misbehavior Detection in VANETs [32] | VeReMi dataset | Random Forest |
| Detection of Position Falsification Attacks in VANETs Applying Trust Model and Machine Learning [35] | VeReMi dataset | Naive Bayes, Logistic regression, KNN, Random Forest |
| Integrating plausibility checks and machine learning for Misbehavior detection in VANET [15] | VeReMi dataset | KNN, SVM |
| Misbehavior Detection using Machine Learning in Vehicular Communication Networks [36] | VeReMi dataset | Decision Tree, KNN, Logistic Regression, Random Forest and Bagging |

Table I shows an overview of recent ML based approaches that have been proposed in the literature specifically to detect position falsification attacks in BSMs. Unlike existing approaches, the proposed methodology uses a vehicle-RSU pair approach with a combined feature set from consecutive BSMs for position falsification detection. Different machine learning algorithms are used to classify legitimate vehicles and attacker vehicles. A preliminary version of this work with initial results has been presented in [41]. In this paper, we have added a detailed description of our approach and introduced a new representation for attack type 16, which is able to improve detection accuracy. We have also added new results for different vehicle densities and comparisons with ML models that use only single BSMs.

## III. PROPOSED ML-BASED APPROACH FOR MISBEHAVIOUR DETECTION

In this section, we discuss our proposed misbehavior detection approach. In particular, we focus on position falsification attacks, where a legitimate sender (i.e. with valid credentials) inserts incorrect information about its position in the BSMs. Cryptograhic techniques alone may not be sufficient to detect such attacks; therefore, we present a data-centric approach that uses ML models to classify vehicles as legitimate or malicious, based on the contents of the BSMs sent by a vehicle. The proposed "2BSM approach" combines information from consecutive BSMs sent by a vehicle to detect position falsification attacks more accurately compared to using single BSMs only.

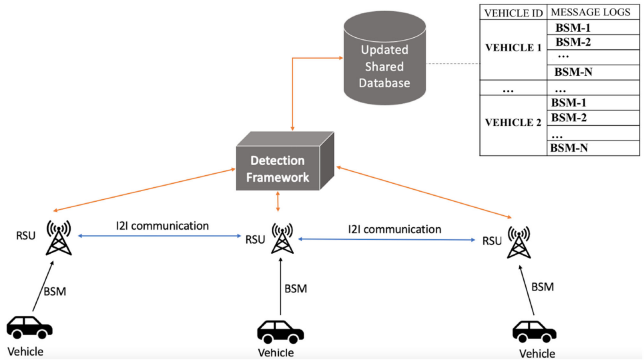



**FIGURE 3.** Proposed Architecture.

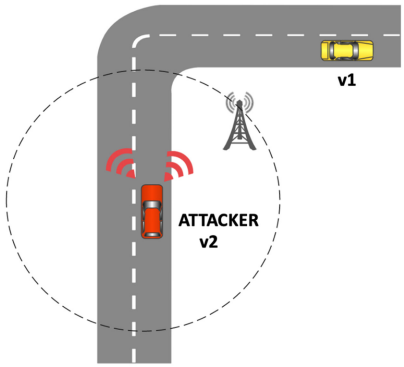



**FIGURE 4.** An example of attacker in the network.

### A. PROPOSED ARCHITECTURE

Fig. 3 shows the communication architecture for our proposed misbehavior detetection framework. In order to participate in the network, each vehicle must first register with the regional authorization party or certificate authority, which provides it with appropriate credentials, e.g., public-private key pair(s) for communication. Authorized vehicles periodically generate their own BSMs and digitally sign them before broadcasting them on the network. These BSMs are received by neighboring vehicles and RSUs within its communication range. RSUs in the network can also communicate with each other and other infrastructure nodes, through a wired backbone network. The BSMs received by a RSU are used to update a shared database, which can be accessed by the other RSUs as well as the misbehavior detection framework.

Unlike existing approaches, where individual vehicle OBUs are responsible for running the ML models and detecting misbehavior, in the proposed scheme the detection framework is installed at the RSUs. On receiving a new BSM from a vehicle, the RSU accesses the shared database to retrieve the last received BSM from the same sender. The proposed framework deployed in the RSU then combines the information from the 2 BSMs in the proper format and applies the ML models to classify the vehicle as legitimate or attacker. After classification, the latest BSM received by a vehicle is updated into the shared database if necessary. When a vehicle is classified as an "attacker" vehicle, the RSU informs the nearby vehicles and infrastructures about the misbehaving vehicle. Upon receiving such *alert* messages from RSUs, each vehicle adds this information to a local log of such flagged vehicles, which is maintained by the OBU. Any additional actions taken by the certificate authorities and other nodes, after receiving such notifications, will depend on the specific network policies and is out of the scope of this paper. Our focus is simply to detect the misbehavior.

RSUs have more computational resources available for training and attack detection, while vehicle OBUs are more resource constrained. Another advantage of the proposed approach is that a vehicle can be notified of a potential attacker, even before entering its communication range. For example, as shown in Fig. 4, the vehicle $v_1$ is not within the range of the attacker vehicle $v_2$. However, the RSU can classify the attacker based on the received BSMs and notify vehicle $v_1$ accordingly. This will allow $v_1$ to be forewarned and take appropriate actions regarding BSMs from $v_2$, if the two vehicles happen to come within communication range at a later time. The additional communication overhead for attack detection is negligible, as BSMs are broadcast by vehicles and will be received by nearby RSUs as well as all surrounding vehicles. If an attacker is detected, the RSU has to only broadcast a single "alert" message to nearby vehicles. The overhead for this is minimal, as most vehicles are expected to be legitimate, and only a single alert message is needed to notify all neighboring vehicles.

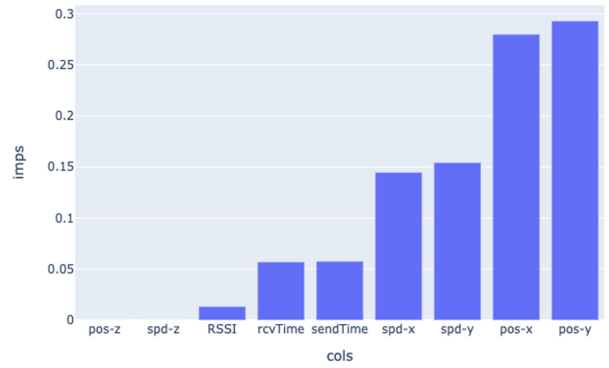
This proposed approach is more suitable for urban areas, as it assumes that there will be a RSU in the vicinity of each vehicle to store the BSMs and run the ML models. If this is not the case, vehicle OBUs will need to be responsible to run their own (possibly simpler, less robust) detection algorithms during any "gaps" in RSU coverage. We also assume that the shared database can be accessed in real time by all RSUs and that the infrastructure nodes are not compromised. We focus on misbehaving *vehicles*, rather than RSUs, as vehicles are more vulnerable and likely to be compromised. However, it is possible for RSUs and other infrastructure nodes to become compromised as well and additional mechanisms, such as those mentioned in [42] [43] can be used in such cases. In the following section, we discuss the steps in the proposed 2BSM approach in detail.

### B. PROCESSING STEPS FOR 2BSM APPROACH

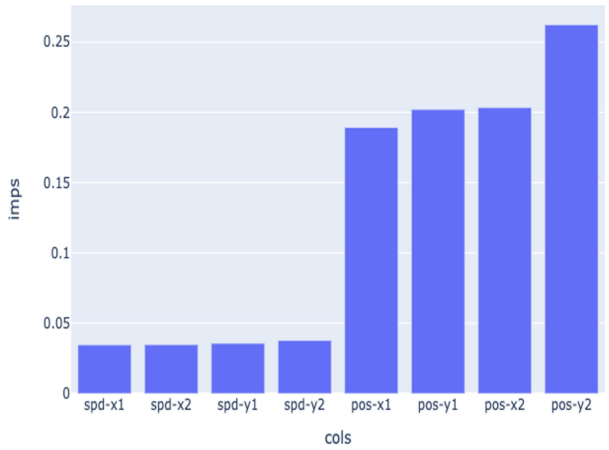
The implementation of the 2BSM approach consists of three main stages: dataset extraction, data preparation and classification.

#### 1) DATA EXTRACTION

In addition to the relevant status information, each BSM contains a timestamp indicating when it was sent, the ID of the sender, as well as a unique message ID for each BSM. We have used the labelled VeReMi dataset [14] to train and test our ML models. For each simulation, there is a single ground truth file and multiple individual log files, which record the




(a) Single BSM Dataset




(b) Two-consecutive BSM Dataset

**FIGURE 5.** **Feature importance graph.**

BSMs received by the individual vehicles. Thus, the number of log files is equal to the number of receivers. Since each BSM is received by multiple vehicles, it is recorded in many different log files. First, we process the combined log files to remove duplicate data. The ground truth file is then merged with the combined log files, and this merged file is used to generate a labelled dataset for each simulation.

#### 2) DATA PREPARATION

The labelled dataset created in the previous step contains many features; some of these contribute significantly to improve attack detection, while others are not helpful. Such non-contributing features can decrease model accuracy and efficiency. Therefore, we perform an analysis of feature importance to identify the useful features and filter out the non-contributing features. Fig. 5(a) shows the relative importance of different features for a single BSM. We see that the x and y coordinates of the vehicle position and speed are the most useful, while other features such as RSSI value, position and speed noise vectors, etc. provide less important information for the model to train and are removed. Similarly, Fig. 5(b) shows that two BSMs can yield more information of vehicle's behaviour than a single BSM as position coordinates and speeds from both BSMs can provide meaningful information for misbehaviour detection.

Based on feature importance, the the $x$ and $y$ coordinates of the vehicle speed and position information in each BSM were included in the format of the labelled data, while features such as position noise vector, speed noise vector, and message-id were removed. Table II, shows an example of individual items of the labelled dataset used in the 2BSM approach. In Table II, *pos1_x*, *pos1_y*, *spd1_x*, *spd1_y* are the position and speed coordinates of BSM 1 and *pos2_x*, *pos2_y*, *spd2_x*, *spd2_y* are the position and speed coordinates of BSM 2. Label 0 depicts a legitimate vehicle, and 1 depicts an attacker vehicle.

#### 3) CLASSIFICATION

After creating the needed labelled dataset, we used different ML algorithms to train the models to detect attacks. The VeReMi dataset contains five different types of position falsification attacks. We implemented both *binary* classification, to simply classify vehicles as legitimate or attacker and *multi-class* classification to identify the specific attack being carried out. There are many different classification algorithms that can be used to train and test the models. From these, we selected the following four classifiers: K-Nearest Neighbour, Random-Forest, Decision tree, and Naïve Bayes, as they yielded better results compared to the others during our initial simulations.

*Hyperparameter tuning:* Hyperparameters are certain values that control the learning of the model and can improve the accuracy and optimize the performance of the model, if adjusted properly. For example, in the K-Nearest Neighbour algorithm, the number of neighbours can be tuned, and the value which performs the best selected for classification. In this research, K(= number of nearest neighbours) was tuned for values in range $K = 3$ to $K = 20$, and the best results were obtained with $K = 3$. For the Random Forest classifier, number of estimators used to generate the results was kept at 20. No notable difference was seen in the results, by increasing the estimators; but for high value of estimators, the classifier took more time to train.

*Cross-validation:* K-fold cross-validation was performed on the dataset to prevent the model from overfitting and efficiently measuring its accuracy. The entire dataset was split into $k$ folds of *train* and *test* sets, where one split became the validation set and remaining *k-1* split used for training. The value of $k$ usually lies between 5-10, depending on the dataset. In this implementation, we used $k = 5$, 10, and both generated similar results.

### C. MODIFIED ATTACK TYPE 16

During the process for training the models, we found that all the classifiers had difficulty in detecting attack type 16, where the vehicle behaves normally for some time in the network and then transmits the same position repeatedly in the network as if it made an eventual stop. In this case, the VeReMi dataset labelled the vehicle as an attacker vehicle,

**TABLE II.** An Example of a Two-Consecutive BSM Dataset

| Vehicle No. | pos1_x | pos1_y | spd1_x | spd1_y | pos2_x | pos2_y | spd2_x | spd2_y | Label |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 3609.39 | 5446.80 | -3.53 | 30.62 | 3605.87 | 5477.34 | -3.53 | 30.62 | 0 |
| 2 | 3586.20 | 5707.55 | 0.19 | 0.45 | 3816.45 | 5245.45 | 1.10 | 2.37 | 1 |
| 3 | 3815.61 | 5243.85 | -5.72 | 36.70 | 3816.45 | 5245.45 | -5.71 | 36.64 | 0 |

even during the period when it was behaving normally. This created a degree of confusion in the network, and as a result, affected the detection accuracy of the models. So, we modified the dataset for attack type 16 and created another attack, which we designated as "modified attack type 16". In this attack, the attacker vehicle is labelled as an attacker only when it starts misbehaving in the network. This helped to improve the detection accuracy, as discussed in detail in Section IV-B

## IV. SIMULATION SETUP AND RESULTS

In order to validate the performance of the proposed approach and ensure fair comparisons, we have decided to use a well-known, publicly available dataset (VeReMi dataset) [14] to train and test our models. We have evaluated the proposed approach and compared its performance with existing approaches, using standard metrics on this dataset. In this section, we first briefly discuss the VeReMi dataset, including the different types of position falsification attacks that are considered as well as the metrics used to evaluate the proposed approach. Next we compare the performance of the different classification algorithms and finally we show how the proposed 2BSM approach achieves improved detection accuracy compared to existing techniques.

### A. VEREMI DATASET AND EVALUATION METRICS

VeReMi dataset [44] consists of 225 individual simulations with five different attacker types, three different traffic densities (low, medium and high), three different attacker densities (10%, 20% and 30%), and five repetitions of each parameter set with random seeds. The dataset is created using the Luxembourg traffic scenario (LuST) [45], which offers a wide-ranging scenario for evaluating VANET applications. For low vehicle density, the simulation consists of 35-39 vehicles, which generate 908-1144 individual BSMs. The medium and high density scenarios simulate up to 108 and 519 vehicles generating up to 4489 and 21878 individual messages respectively.

There are five specific types of position falsification attacks generated in the VEREMI dataset. These are:

1) Constant attack (Attack Type 1): Attacker vehicle transmits fixed position in the network.
2) Constant offset attack (Attack Type 2): Attacker vehicle transmits a position with a fixed offset added to the actual position.
3) Random attack (Attack Type 4): Attacker vehicle transmits random position from the playground.
4) Random offset attack (Attack Type 8): Attacker vehicle transmits a uniformly random position from a predefined rectangle around the vehicle.
5) Eventual stop attack (Attack Type 16): Attacker vehicle behaves like a legitimate vehicle for some time and then transmits a current position repeatedly in the network.

The VeReMi Dataset is an imbalanced dataset [46] containing information from both legitimate vehicles and attacker vehicles. Since accuracy alone is not an adequate metric for an imbalanced dataset, we use the following metrics, shown in eqns (1) – (3), to evaluate and compare the performance of the proposed approach. In our dataset, positive denotes attacker, and negative indicates legitimate vehicle.

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive} \tag{1}$$

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative} \tag{2}$$

$$F1\text{-}score = 2 * \frac{Precision\ *\ Recall}{Precision\ +\ Recall} \tag{3}$$

### B. COMPARISON OF DIFFERENT CLASSIFIERS

In this section, we consider four classifier algorithms (K-Nearest Neighbour, Random Forest, Naïve Bayes and Decision Tree) and compare their performance using the 2BSM approach. We also discuss the performance of the proposed 2BSM approach, compared to using data from a single BSM, for different attack types.

#### 1) BINARY CLASSIFICATION RESULTS

Tables III, IV, V show the results of the 2BSM models, using the four classification algorithms for low, medium and high vehicle density respectively. Most of the algorithms, except for Naive Bayes, performed well for the different attack types. The Naive Bayes model was able to detect Attack types 1 and 4 accurately but performed poorly for the other attacks. Although, there were some minor fluctuations, we did not observe any significant differences as the vehicle density was changed. The performance of our proposed 2BSM approach for each attack type is discussed below:

**Attack type 1:** This type of attack is easy to identify, as a vehicle constantly transmits a fixed location but not a fixed velocity. This is clear from the Tables, as all 4 classifieres were able to detect type 1 attacks with 99.9% accuracy or better for different vehicle densities. K-Nearest Neigbbour and Naïve Bayes algorithms showed 100% successful detection for all three densities, whereas Random Forest and Decision

**TABLE III.** Classification Results of Proposed model-LOW

| Algorithm: | Precision | Recall | F1-Score |
|---|---|---|---|
| **ATTACK 1** | | | |
| K Nearest Neighbor | 100 | 100 | 100 |
| Random Forest | 100 | 100 | 100 |
| Naive Bayes | 100 | 100 | 100 |
| Decision Tree | 100 | 100 | 100 |
| **ATTACK 2** | | | |
| K Nearest Neighbor | 100 | 100 | 100 |
| Random Forest | 100 | 99.7 | 99.8 |
| Naive Bayes | 22 | 16.6 | 20 |
| Decision Tree | 99.7 | 99.5 | 99.6 |
| **ATTACK 4** | | | |
| K Nearest Neighbor | 100 | 97.9 | 98.9 |
| Random Forest | 100 | 99.4 | 99.7 |
| Naive Bayes | 92.2 | 100 | 95.7 |
| Decision Tree | 99.7 | 96.5 | 98 |
| **ATTACK 8** | | | |
| K Nearest Neighbor | 100 | 90.2 | 94.6 |
| Random Forest | 99.2 | 96.7 | 97.9 |
| Naive Bayes | 48.9 | 9.1 | 15.3 |
| Decision Tree | 97.7 | 96.7 | 97.5 |
| **ATTACK 16** | | | |
| K Nearest Neighbor | 96.7 | 94.2 | 95 |
| Random Forest | 97.1 | 93.4 | 95.2 |
| Naive Bayes | 11.4 | 100 | 20.5 |
| Decision Tree | 95.3 | 92.4 | 94.1 |
| **MODIFIED ATTACK 16** | | | |
| K Nearest Neighbor | 100 | 99.6 | 99.8 |
| Random Forest | 98.3 | 95.6 | 96.9 |
| Naive Bayes | 10.6 | 99.3 | 19.4 |
| Decision Tree | 97.1 | 96.7 | 96.9 |

**TABLE IV.** Classification Results of Proposed model-MEDIUM

| Algorithm: | Precision | Recall | F1-Score |
|---|---|---|---|
| **ATTACK 1** | | | |
| K Nearest Neighbor | 100 | 100 | 100 |
| Random Forest | 100 | 100 | 100 |
| Naive Bayes | 100 | 100 | 100 |
| Decision Tree | 99.9 | 100 | 99.9 |
| **ATTACK 2** | | | |
| K Nearest Neighbor | 99.7 | 98.3 | 99 |
| Random Forest | 99.8 | 99 | 99.4 |
| Naive Bayes | 73.2 | 12.7 | 21.6 |
| Decision Tree | 98.9 | 98.6 | 98.7 |
| **ATTACK 4** | | | |
| K Nearest Neighbor | 100 | 99.5 | 99.8 |
| Random Forest | 100 | 99.8 | 99.9 |
| Naive Bayes | 100 | 99.2 | 99.6 |
| Decision Tree | 100 | 99.8 | 99.9 |
| **ATTACK 8** | | | |
| K Nearest Neighbor | 99.8 | 92.5 | 95.9 |
| Random Forest | 99.1 | 95.2 | 97.1 |
| Naive Bayes | 47.6 | 7.6 | 13.1 |
| Decision Tree | 97.8 | 95.9 | 96.8 |
| **ATTACK 16** | | | |
| K Nearest Neighbor | 96.5 | 95.4 | 95.7 |
| Random Forest | 96.5 | 95.5 | 96 |
| Naive Bayes | 40.8 | 21.2 | 27.9 |
| Decision Tree | 94.1 | 96.1 | 95.1 |
| **MODIFIED ATTACK 16** | | | |
| K Nearest Neighbor | 98.1 | 98.7 | 98.5 |
| Random Forest | 97.7 | 98 | 97.9 |
| Naive Bayes | 36.2 | 20 | 25.7 |
| Decision Tree | 96.6 | 97.8 | 97.2 |

Tree identified all the attacker vehicles, but 0.01% of honest vehicles were misclassified in the high-density dataset.

**Attack type 2:** Constant offset attack is harder to detect as the attacker modifies the position by adding a fixed offset to it, so position changes are often very similar to normal vehicle movements. The attack was classified with more than 99 percent precision and recall using K-Nearest Neigbbour, Random Forest, and Decision Tree in low and high-density data and similar results with more than 98 percent classification in medium density. The Naïve Bayes algorithm, however, performed quite poorly compared to the other classifiers and performance varied widely depending on vehicle density.

**Attack type 4:** In attack type 4, the vehicle sends a random position from the simulation playground. This attack can be easily detected using the 2BSM approach, as there will be little correlation between the two successive position coordinates from a vehicle. As expected, the results indicate that attack type 4 is detected with high precision and recall by all four algorithms in all three densities.

**Attack type 8:** Similar to attack type 4, this attack transmits random positions; however these positions are selected from a fixed area near the vehicle. Since the distance between the actual and reported (i.e. false) position is smaller, detecting this attack is more difficult. However, the proposed 2BSM model still performed well with Random Forest classifiers and Decision Tree classifiers in low and medium density. Although KNN was able to generate only 90% and 92% recall for low and medium density, it significantly improved the performance for high-density data and gave more than 99% precision and recall values. Naïve Bayes classifier did not perform well in classifying this attack.

**Attack type 16:** In this attack, the attacker transmits its correct positions for a certain period of time, and then starts to

**TABLE V. Classification Results of Proposed Model- HIGH**

| Algorithm: | Precision | Recall | F1-Score |
|---|---|---|---|
| **ATTACK 1** | | | |
| K Nearest Neighbor | 100 | 100 | 100 |
| Random Forest | 99.9 | 100 | 99.9 |
| Naive Bayes | 100 | 100 | 100 |
| Decision Tree | 99.9 | 100 | 99.9 |
| **ATTACK 2** | | | |
| K Nearest Neighbor | 99.8 | 99.7 | 99.8 |
| Random Forest | 99.8 | 99.6 | 99.7 |
| Naive Bayes | 54.8 | 41.8 | 47.4 |
| Decision Tree | 99.4 | 99.3 | 99.4 |
| **ATTACK 4** | | | |
| K Nearest Neighbor | 100 | 99.8 | 99.9 |
| Random Forest | 99.9 | 99.9 | 99.9 |
| Naive Bayes | 100 | 99.5 | 99.7 |
| Decision Tree | 99.9 | 99.9 | 99.9 |
| **ATTACK 8** | | | |
| K Nearest Neighbor | 99.9 | 97.2 | 98.5 |
| Random Forest | 99.4 | 97.5 | 98.6 |
| Naive Bayes | 68.6 | 14.9 | 24.4 |
| Decision Tree | 98.8 | 97.7 | 98.3 |
| **ATTACK 16** | | | |
| K Nearest Neighbor | 96.8 | 95.2 | 96 |
| Random Forest | 96.7 | 94.6 | 95.5 |
| Naive Bayes | 53.1 | 11 | 18.3 |
| Decision Tree | 94.1 | 94.6 | 94.3 |
| **MODIFIED ATTACK 16** | | | |
| K Nearest Neighbor | 98.7 | 98.5 | 98.7 |
| Random Forest | 98.4 | 97.3 | 97.9 |
| Naive Bayes | 58 | 9 | 15.3 |
| Decision Tree | 97.6 | 97.1 | 97.3 |

transmit the same location repeatedly in subsequent BSMs. In contrast to the other four attack types, the classification of attack type 16 yielded slightly lower precision and recall values in all three densities. The model showed no improvement in performance with an increase in the data density. This could be because the vehicle is labelled as an attacker, even during the time it is acting normally, confusing the machine learning model.

**Modified Attack type 16:** To address the lower detection accuracy for Attack type 16, we modified the dataset as follows: when the vehicle is sending correct positions in the BSMs, the corresponding label for that instance was changed from "attacker" to "legitimate". The "attacker" label is set only when the BSM contains false information. Using this modified dataset, we observed significant improvements in the

**TABLE VI. Comparison of 2BSM and 1BSM Approaches Using K-Nearest Neigbbour**

| Metric | Attack 1 | | Attack 2 | |
|---|---|---|---|---|
| | 1 BSM | 2 BSM | 1 BSM | 2BSM |
| Precision | 99.3 | 100 | 98.8 | 99.8 |
| Recall | 99.3 | 100 | 94.1 | 99.7 |
| F1-score | 99.3 | 100 | 96.4 | 99.8 |
| | Attack 4 | | Attack 8 | |
| Precision | 99.9 | 100 | 98.7 | 99.9 |
| Recall | 98.4 | 99.8 | 86.5 | 97.2 |
| F1-score | 99.2 | 99.9 | 92.2 | 98.5 |
| | Attack 16 | | Modified Attack 16 | |
| Precision | 97 | 96.8 | 96.4 | 98.7 |
| Recall | 93.9 | 95.2 | 93.6 | 98.5 |
| F1-Score | 95.4 | 96 | 94.9 | 98.6 |

classification results for K-Nearest Neigbbour, Random Forest, and Decision Tree. Naïve Bayes classifier tried to classify all the BSMs into "attacker," giving almost 100% recall value but extremely low precision in low-density data. For high and medium vehicle density, Naïve Bayes had better precision values, but recall value dropped; hence F1-scores were low in all three densities.

Based on the results reported in Tables III, IV, V, the K-Nearest Neigbbour algorithm generally yielded the best results overall for different vehicle densities. In Table VI, we compare the performance of the 2BSM model with the traditional approach using a single BSM (1BSM), for high vehicle density. For both 1BSM and 2BSM approaches, we used the K-Nearest Neighbour algorithm to train and test the models. The results for low and medium densities were very similar and have been omitted. The results clearly indicate the the proposed 2BSM approach is able consistently detect misbehavior more accurately for all attack types considered in the dataset.

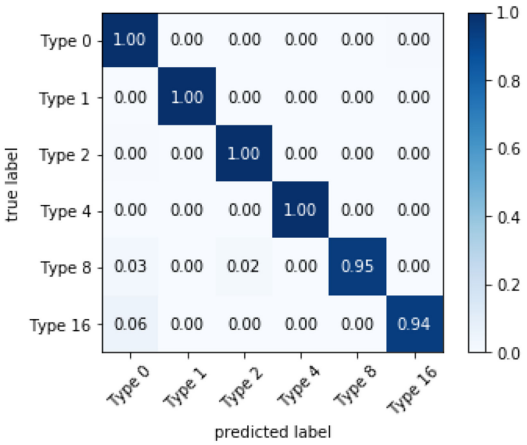
### 2) MULTICLASS CLASSIFICATION RESULTS

Multiclass classification is used to classify a dataset of more than two classes/labels. For this classification category, we created a dataset that includes all five attack types in a single dataset. Two versions of this combined dataset were created - the first with the original attack type 16 and the second with the modified attack type 16. Table VII depicts classification results obtained using the proposed 2BSM approach on both versions of the multi-class dataset. Compared with the other three classifiers, the K-Nearest Neighbour classifier achieved better results in both the datasets, while the Naive Bayes classifier showed unsatisfactory performance. A slight improvement in classification results is seen with modified attack type 16.

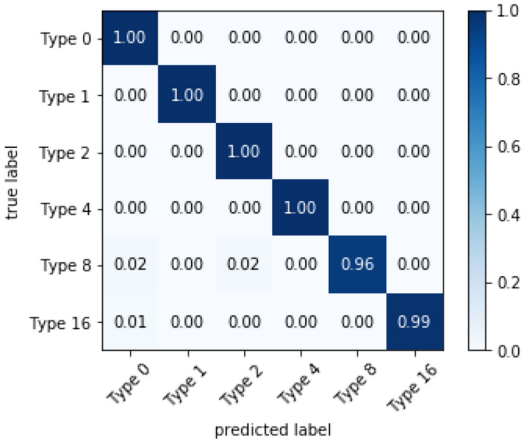
Fig. 6 shows a normalized confusion matrix, generated using a K-Nearest Neighbour classifier, to depict the way in

**TABLE VII. Classification Results of Multi-Class Classification**

| Classification Algorithm | Precision | Recall | F1-score | Precision | Recall | F1-score |
|---|---|---|---|---|---|---|
| With: | ATTACK 16 | | | MODIFIED ATTACK 16 | | |
| K Nearest-Neighbour | 98.8 | 98.1 | 98.5 | 99.2 | 98.8 | 99 |
| Random Forest | 98.7 | 97.8 | 98.3 | 99 | 98.3 | 98.7 |
| Naïve Bayes | 64.5 | 54.9 | 59.1 | 64.4 | 54.8 | 59.2 |
| Decision Tree | 97.9 | 97.8 | 97.8 | 98.6 | 98.4 | 98.5 |




(a) With Attack type 16




(b) With Modified Attack type 16

**FIGURE 6. Confusion matrix of Multi-class classification.**

which different attack types were misclassified. In this confusion matrix, the "Type 0" denotes legitimate BSMs from a vehicle. The results show that only attack types 8 and 16 were misclassified, while the other attacks were correctly identified. Attack type 8 was misclassified as either "Type 0" or "Type 2". Fig. 6(a), which uses the *original* attack type 16, shows that 94% of attack type 16 was classified correctly and 6% was misclassified as "Type 0". But in Fig. 6(b), using *modified* attack type 16, misclassification is reduced to only 1%.

### 3) VISUALIZING THE RESULTS

For visualizing the results obtained, we used a precision-recall curve [47]. The precision-recall curve is most commonly used for situations involving imbalanced datasets, and it is used for evaluating the performance of binary classification. The precision-recall curve demonstrates the trade-off between precision value and recall value. A larger area under the curve implies both recall and precision have a high value. High precision denotes a low false positive rate, and high recall means a low false-negative rate.

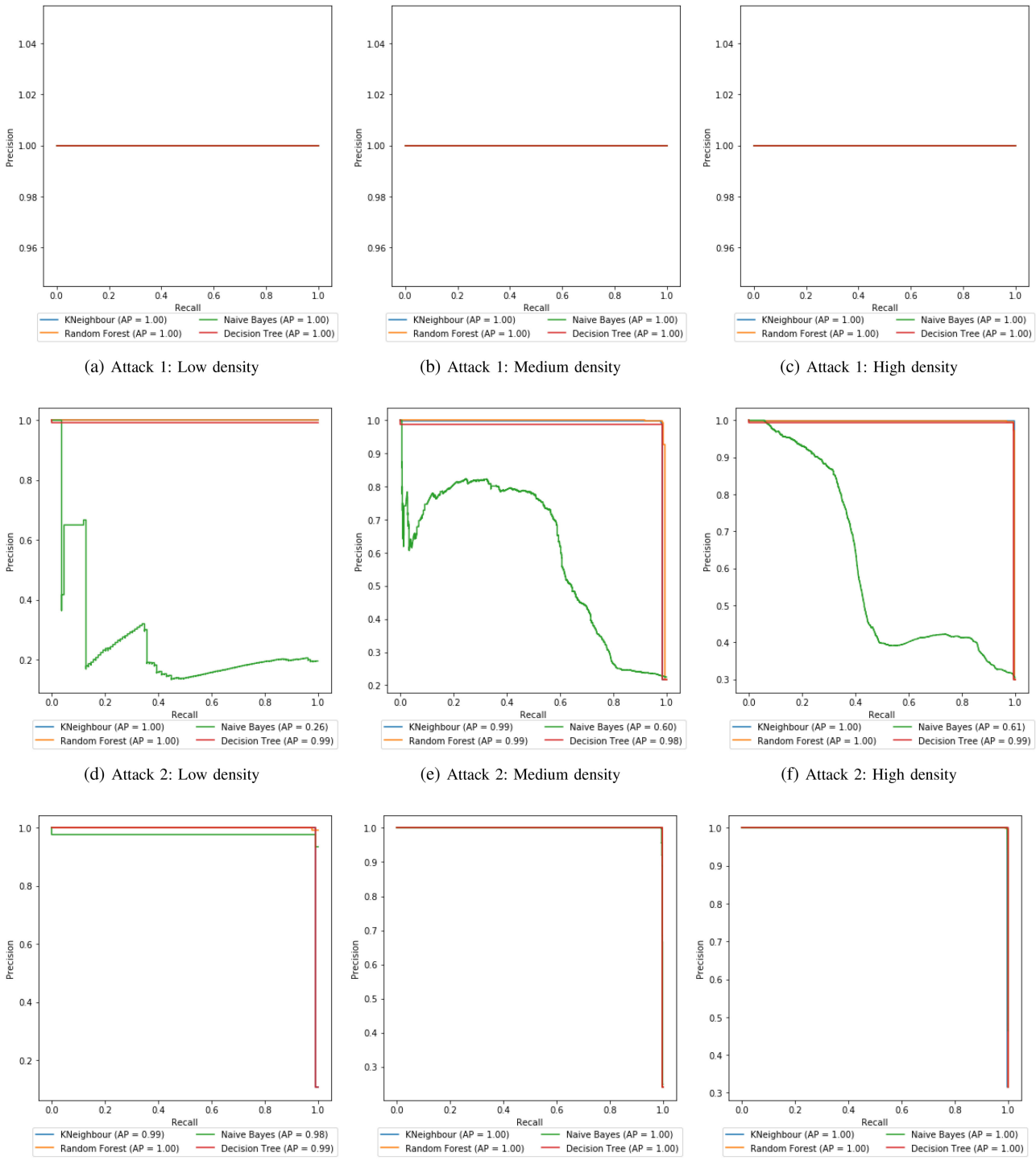
Figs. 7 and 8 show the precision-recall curves for the different attack types for low, medium and high traffic densities. Attack type 1 has zero false positives and perfectly separates the area into two areas. Similarly, for attack type 4, both precision and recall are very high as well. This means that these two attacks can be detected accurately, by all the algorithms. For the remaining attacks, K-Nearest Neighbour, Random Forest, and Decision Tree classifiers perform well, with Decision Tree performing slightly less than the other two classifiers. In contrast, Naïve Bayes showed poor results with a noisy graph (indicated by the zig-zag curves), with much lower area under the curve.

## V. COMPARISON WITH EXISTING APPROACHES

Based on the results in the previous section, the K-Nearest Neighbour classifier had the overall best performance using the 2BSM approach. In this section, we compare the performance of the proposed approach, using K-Nearest Neighbour, with some existing techniques that also used the VeReMi dataset for detecting position falsification. The proposed model was also compared to a raw dataset consisting of single BSM data from a vehicle, without any feature selection. As expected, the raw dataset performed poorly for almost all attacks. The only exception was a high precision score for attack type 4.

Table VIII compares the precision and recall values obtained using the 2BSM approach with techniques reported in some recent papers. Paper 1 [14] and Paper 3 [36] performed similarly to the proposed model for attack types 1 and 4 generating high precision and recall values; however, Paper 2 [15] showed comparatively less precision and recall value.

The proposed model showed the best performance, compared to existing techniques, for Attack type 2, which was classified with more than 99% precision and recall using the




(a) Attack 1: Low density (b) Attack 1: Medium density (c) Attack 1: High density

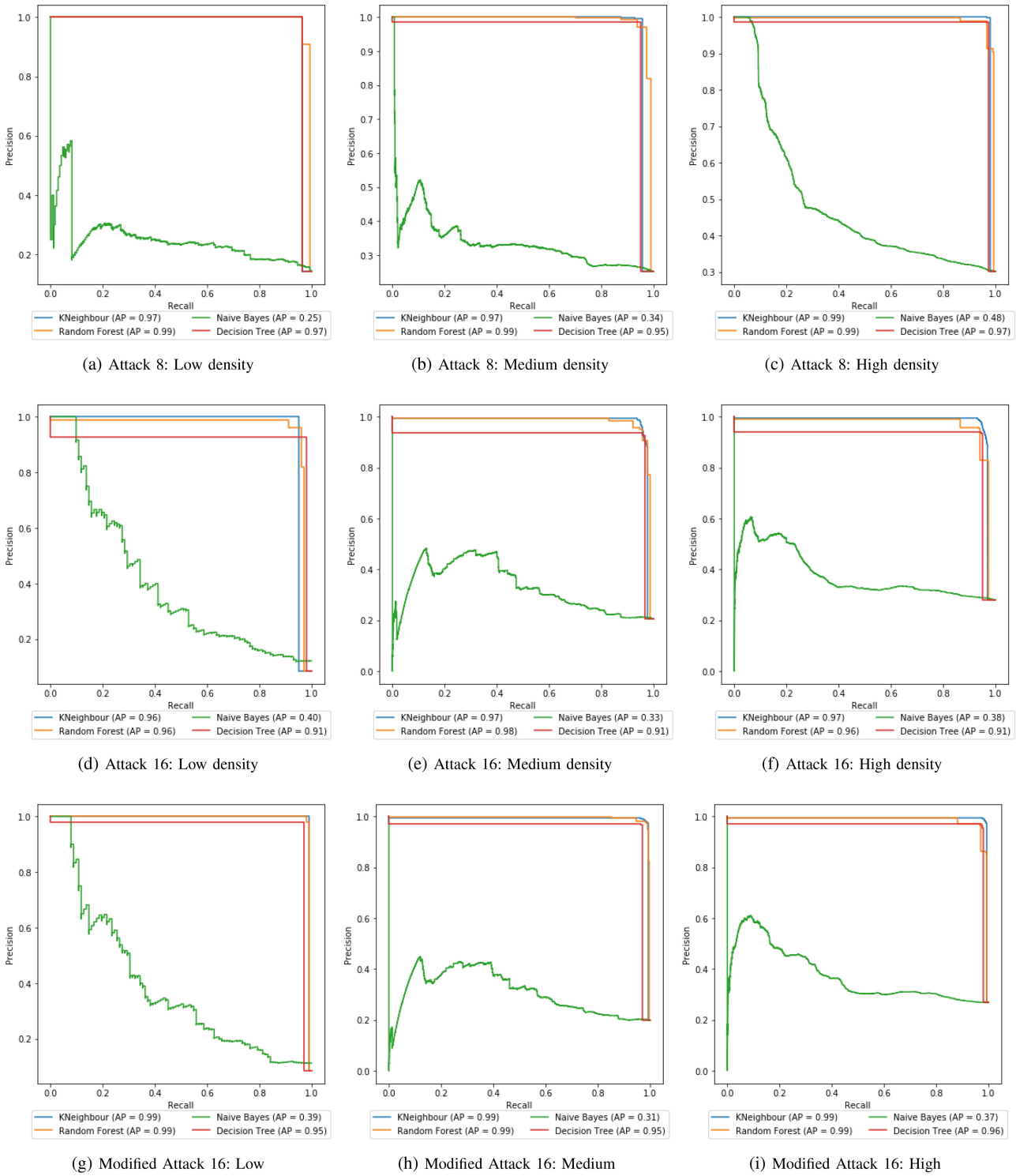
(d) Attack 2: Low density (e) Attack 2: Medium density (f) Attack 2: High density

(g) Attack 4: Low density (h) Attack 4: Medium density (i) Attack 4: High density

**FIGURE 7. Precision-recall curve of attack types 1,2 and 4 in low, medium and high density.**

proposed model, whereas the existing approaches showed varied results. Paper 1 obtained a 100% recall value, but the precision value was very low. Paper 3 showed the highest results out of the three existing approaches, with Paper 2 not showing satisfactory results. For attack type 8, the 2BSM model again had the best performance, with Paper 3, also giving high values for both precision and recall. Overall, Attack types 2 and 8 are more challenging to detect, and our proposed model achieved higher precision and recall than the other existing techniques.

In the case of attack type 16, our model showed promising results in maintaining a balance between precision and




(a) Attack 8: Low density (b) Attack 8: Medium density (c) Attack 8: High density

(d) Attack 16: Low density (e) Attack 16: Medium density (f) Attack 16: High density

(g) Modified Attack 16: Low (h) Modified Attack 16: Medium (i) Modified Attack 16: High

**FIGURE 8.** **Precision-recall curve of attack types 8, 16 and modified attack type 16 in low, medium and high density.**

recall, with results similar to Paper 3. The performances for Papers 1 and 2, were significantly lower. Using the *modified* attack type 16 dataset, improves the performance of the proposed approach even further, compared to existing techniques.

### *A. MULTICLASS CLASSIFICATION*

Of the existing techniques we used for comparison only Paper 2 [15] reported results for multiclass classification. Table IX compares the multiclass precision and recall values of the proposed approach with that in Paper 2. The 2BSM model is able

**TABLE VIII. Comparison of Proposed Model With Existing Approaches**

| Results from: | Attack 1 | | Attack 2 | | Attack 4 | | Attack 8 | | Attack 16 | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Precision | Recall | Precision | Recall | Precision | Recall | Precision | Recall | Precision | Recall |
| **Raw Dataset** | 57.4 | 67.2 | 34.9 | 18.8 | 99.8 | 68.9 | 29 | 14.7 | 31 | 16 |
| **Paper 1: [14]** | 100 | 100 | 40 | 100 | 100 | 99 | 70 | 95 | 80 | 90 |
| **Paper 2: [15]** | 95.2 | 83.2 | 56.1 | 19.3 | 95 | 83.6 | 96.2 | 82.5 | 71.4 | 42.5 |
| **Paper 3: [36]** | 100 | 99 | 94 | 80 | 100 | 99 | 97 | 95 | 98 | 93 |
| **Proposed 2BSM Model** | 100 | 100 | 99.8 | 99.7 | 100 | 99.8 | 99.9 | 97.2 | 96.8 | 95.2 |

**TABLE IX. Comparison of Multi-Class Classification**

| Metric | Precision | Recall |
|---|---|---|
| Proposed 2BSM Model | 98.1 | 98.5 |
| Proposed 2 BSM Model (With modified Attack 16) | 98.8 | 99 |
| Paper 2 [15] | 88.7 | 61.6 |

to correctly classify over 98% of attackers, using both original and modified attack type 16; while the precision reported in Paper 2 is only 88%. The recall for the approach in Paper 2 is even lower (61.6%), indicating a significantly higher number of attackers are not being recognized properly.

## VI. CONCLUSION

This paper proposes a novel Machine Learning-based approach for classifying position falsification attacks in VANET. The proposed approach shifts the computational overhead from vehicles (OBUs), implementing the misbehavior detection framework in the RSUs, which can share this information widely with other RSUs and vehicles. Unlike existing techniques, the proposed scheme uses 2 consecutive BSMs from the same vehicle to create an augmented dataset, which is used to train the proposed model using different machine learning algorithms. Comparing different ML algorithms, it was observed that K-Nearest Neighbour and Random Forest classifiers yield the best results. The performance of the proposed model was also compared with the recent results reported in the literature for existing ML-based techniques. The obtained results indicate that the proposed approach consistently outperforms the existing methods across different attack types, in terms of both precision and recall.

In this work, the proposed models were trained using the five specific attack types given in the VeReMi dataset only, which does not represent all possible position falsification attacks in VANETs. For secure VANET operation, it is necessary to develop robust models capable of detecting incorrect information in other BSM parameters (e.g., speed, acceleration, heading etc), as well as recognizing unknown attacks. We are currently investigating deep learning based approaches that can discover hidden patterns from the data to defend against both known and emerging threats.

## REFERENCES


[1] "Global status report on road safety 2018," Accessed: Jan. 5, 2021. [Online]. Available: http://apps.who.int/iris/bitstream/handle/10665/277370/WHO-NMH-NVI-18.20-eng.pdf?ua=1Re

[2] S. Al-Sultan, M. M. Al-Doori, A. H. Al-Bayatti, and H. Zedan, "A comprehensive survey on vehicular Ad Hoc network," *J. Netw. Comput. Appl.*, vol. 37, pp. 380–392, 2014.

[3] S.-h. An, B.-H. Lee, and D.-R. Shin, "A survey of intelligent transportation systems," in *Proc. 3rd Int. Conf. Comput. Intell., Commun. Syst. Netw.*, 2011, pp. 332–337.

[4] S. Zeadally, R. Hunt, Y.-S. Chen, A. Irwin, and A. Hassan, "Vehicular ad hoc networks (VANETs): Status, results, and challenges," *Telecommun. Syst.*, vol. 50, no. 4, pp. 217–241, 2012.

[5] F. Bai, H. Hartenstein, M. Gruteser, R. Kravets, T. Zhang, and D. D. Stancil, "Special section on vehicular networks and communication systems: From laboratory into reality," *IEEE Trans. Veh. Technol.*, vol. 62, no. 9, pp. 4146–4149, Nov. 2013.

[6] H. Hasrouny, A. E. Samhat, C. Bassil, and A. Laouiti, "VANET security challenges and solutions: A survey," *Veh. Commun.*, vol. 7, pp. 7–20, 2017.

[7] M. Arif, G. Wang, M. Z. A. Bhuiyan, T. Wang, and J. Chen, "A survey on security attacks in VANETs: Communication, applications and challenges," *Veh. Commun.*, vol. 19, 2019, Art. no. 100179.

[8] S. S. Manvi and S. Tangade, "A survey on authentication schemes in VANETs for secured communication," *Veh. Commun.*, vol. 9, pp. 19–30, 2017.

[9] G. Bonaccorso, *Machine Learning Algorithms*. Birmingham, U.K.: Packt Publishing Ltd., 2017.

[10] M. Binkhonain and L. Zhao, "A review of machine learning algorithms for identification and classification of non-functional requirements," *Expert Syst. With Appl.: X*, vol. 1, 2019, Art. no. 100001.

[11] M. A. Hezam *et al.*, "Classification of security attacks in VANET: A review of requirements and perspectives," in *Proc. MATEC Web Conf 150 06038*, 2018, doi: 10.1051/matecconf/201815006038.

[12] P. Tyagi and D. Dembla, "A taxonomy of security attacks and issues in vehicular Ad-Hoc networks (VANETs)," *Int. J. Comput. Appl.*, vol. 91, no. 7, pp. 22–29, 2014.

[13] R. W. van der Heijden, S. Dietzel, and F. Kargl, "Misbehavior detection in vehicular Ad-hoc networks," in *Proc. 1st Inter- Veh. Commun. Conf. (FG-IVC 2013)*, 2013, pp. 23–25.

[14] R. W. van der Heijden, T. Lukaseder, and F. Kargl, "VeReMi: A dataset for comparable evaluation of misbehavior detection in VANETs," in *Security and Privacy in Communication Networks*, R. Beyah *et al.* Eds., New York, NY, USA: Springer, 2018, pp. 318–337.

[15] S. So, P. Sharma, and J. Petit, "Integrating plausibility checks and machine learning for misbehavior detection in VANET," in *Proc. 17th IEEE Int. Conf. Mach. Learn. Appl.*, 2018, pp. 564–571.

[16] M. L. Han, B. I. Kwak, and H. K. Kim, "Anomaly intrusion detection method for vehicular networks based on survival analysis," *Veh. Commun.*, vol. 14, pp. 52–63, 2018.

[17] X. Xue, N. Lin, J. Ding, and Y. Ji, "A trusted neighbor table based location verification for VANET routing," in *IET 3rd Int. Conf. Wireless, Mobile Multimedia Netw.*, 2010, pp. 1–5, doi: 10.1049/cp.2010.0603.

[18] K. Zaidi, M. B. Milojevic, V. Rakocevic, A. Nallanathan, and M. Rajarajan, "Host-based intrusion detection for VANETs: A statistical approach to rogue node detection," *IEEE Trans. Veh. Technol.*, vol. 65, no. 8, pp. 6703–6714, Aug. 2016.

[19] M. Mohammed, M. B. Khan, and E. B. M. Bashier, *Machine Learning: Algorithms and Applications*. Boca Raton, FL, USA: CRC Press, 2016.

[20] V. Chaoji, R. Rastogi, and G. Roy, "Machine learning in the real world," in *Proc. VLDB Endowment*, vol. 9, no. 13, pp. 1597–1600, 2016.
[21] L. Liang, H. Ye, and G. Y. Li, "Toward intelligent vehicular networks: A machine learning framework," *IEEE Internet Things J.*, vol. 6, no. 1, pp. 124–135, Feb. 2 2019.
[22] P. C. Sen, M. Hajra, and M. Ghosh, "Supervised classification algorithms in machine learning: A survey and review," *Emerging Technology in Modelling and Graphics*. Singapore: Springer, 2020, pp. 99–111.
[23] A. Mucherino, P. J. Papajorgji, and P. M. Pardalos, "K-nearest neighbor classification," in *Data Mining in Agriculture*. New York, NY, USA: Springer, 2009, pp. 83–106.
[24] A. Priyam, G. Abhijeeta, A. Rathee, and S. Srivastava, "Comparative analysis of decision tree classification algorithms," *Int. J. Curr. Eng. Technol.*, vol. 3, no. 2, pp. 334–337, 2013.
[25] A. Liaw *et al.*, "Classification and regression by randomforest," *R News*, vol. 2, no. 3, pp. 18–22, 2002.
[26] K. M. Leung, "Naive bayesian classifier," *Polytech. Univ. Dept. Comput. Sci./Finance Risk Eng.*, vol. 2007, pp. 123–156, 2007.
[27] J. Grover, V. Laxmi, and M. S. Gaur, "Misbehavior detection based on ensemble learning in VANET," in *Proc. Int. Conf. Adv. Comput., Netw. Secur.* Berlin, Heidelberg: Springer, 2011, pp. 602–611.
[28] P. K. Singh, R. R. Gupta, S. K. Nandi, and S. Nandi, "Machine learning based approach to detect wormhole attack in VANETs," in *Proc. Workshops Int. Conf. Adv. Inf. Netw. Appl.* Cham: Springer, 2019, pp. 651–661.
[29] T. G. Dietterich, "Ensemble methods in machine learning," in *Proc. Int. Workshop Multiple Classifier Syst.* Berlin, Heidelberg: Springer, 2000, pp. 1–15.
[30] F. A. Ghaleb, M. A. Maarof, A. Zainal, B. A. S. Al-rimy, A. Alsaeedi, and W. Boulila, "Ensemble-based hybrid context-aware misbehavior detection model for vehicular ad hoc network," *Remote Sens.*, vol. 11, no. 23, 2019, Art. no. 2852.
[31] F. A. Ghaleb *et al.*, "Misbehavior-aware on-demand collaborative intrusion detection system using distributed ensemble learning for VANET," *Electronics*, vol. 9, no. 9, 2020, Art. no. 1411.
[32] A. Sonker and R. Gupta, "A new procedure for misbehavior detection in vehicular Ad-Hoc networks using machine learning." *Int. J. Elect. Comput. Eng. (2088–8708)*, vol. 11, no. 3, pp. 2535–2547, 2021.
[33] A. Khot and M. Dave, "Position falsification misbehavior detection in VANETs," in *Mobile Radio Communications and 5G Networks*. Singapore: Springer, 2020, pp. 487–499.
[34] P. K. Singh, S. Gupta, R. Vashistha, S. K. Nandi, and S. Nandi, "Machine learning based approach to detect position falsification attack in VANETs," in *Proc. Int. Conf. Secur. Privacy*. Singapore: Springer, 2019, pp. 166–178.
[35] J. Montenegro, C. Iza, and M. Aguilar Igartua, "Detection of position falsification attacks in VANETs applying trust model and machine learning," in *Proc. 17th ACM Symp. Perform. Eval. Wireless Ad Hoc, Sensor, Ubiquitous Netw.*, 2020, pp. 9–16.
[36] S. Gyawali and Y. Qian, "Misbehavior detection using machine learning in vehicular communication networks," in *Proc. ICC IEEE Int. Conf. Commun.*, 2019, pp. 1–6.
[37] S. Sharanya and S. Karthikeyan, "Classifying malicious nodes in VANETs using support vector machines with modified fading memory," *ARPN J. Eng. Appl. Sci.*, vol. 12, no. 1, pp. 171–176, 2017.
[38] M. J. S. Aneja, T. Bhatia, G. Sharma, and G. Shrivastava, "Artificial intelligence based intrusion detection system to detect flooding attack in VANETs," in *Handbook of Research on Network Forensics and Analysis Techniques*. Pennsylvania, PA, USA: IGI Global, 2018, pp. 87–100.
[39] J. Gu and S. Lu, "An effective intrusion detection approach using SVM with naïve bayes feature embedding," *Comput. Secur.*, vol. 103, 2021, Art. no. 102158.
[40] N. Magaia and Z. Sheng, "ReFioV: A novel reputation framework for information-centric vehicular applications," *IEEE Trans. Veh. Technol.*, vol. 68, no. 2, pp. 1810–1823, Feb. 2019.
[41] A. Sharma and A. Jaekel, "Machine learning approach for detecting location spoofing in VANET," in *Proc. Int. Conf. Comput. Commun. Netw.*, 2021, pp. 1–6.
[42] A. Haydari and Y. Yilmaz, "Real-time detection and mitigation of DDoS attacks in intelligent transportation systems," in *Proc. 21st Int. Conf. Intell. Transp. Syst.*, 2018, pp. 157–163.
[43] V. Raghuwanshi and S. Jain, "Denial of service attack in VANET: A survey," *Int. J. Eng. Trends Technol.*, vol. 28, no. 1, pp. 15–20, 2015.
[44] "VeReMi dataset," Accessed: Jan. 5, 2021. [Online]: Available: https://veremi-dataset.github.io/
[45] L. Codecá, R. Frank, S. Faye, and T. Engel, "Luxembourg sumo traffic (LUsT) scenario: Traffic demand evaluation," *IEEE Intell. Transp. Syst. Mag.*, vol. 9, no. 2, pp. 52–63, Summer 2017.
[46] J. Brownlee, "A gentle introduction to imbalanced classification," *Mach. Learn. Mastery*, vol. 22, 2019.
[47] K. Boyd, K. H. Eng, and C. D. Page, "Area under the precision-recall curve: Point estimates and confidence intervals," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Berlin, Heidelberg: Springer, 2013, pp. 451–466.

**AEKTA SHARMA** received the bachelor's degree in computer science from Maharshi Dayanand University, Rohtak, India, in 2017, and the master's degree from the University of Windsor, Windsor, ON, Canada, in April 2021. Along with her studies, she was a Graduate Assistant with the University of Windsor, where she mentors and facilitates lab for undergraduate students. She was a Developer in a Organization called AlayaCare after completing the master's degree. Her main research interest include machine learning, artificial intelligence, amazon web services, and software development.

**ARUNITA JAEKEL** (Member, IEEE) received the B. Eng. degree in electronics and telecommunications engineering from Jadavpur University, Kolkata, India, and the M.A.Sc. and Ph.D. degrees in electrical engineering from the University of Windsor, Windsor, ON, Canada.

Since 1995, she has been a Faculty Member with the School of Computer Science, University of Windsor, where she is currently a tenured Professor. Her research is supported by grants from the Natural Sciences and Engineering Research Council (NSERC), Canada. She has authored or coauthored more than 100 refereed articles in the areas of her current research interests. Her current research interests include vehicle-to-vehicle communication, design of reliable wireless sensor networks, and optical networks.