Journal of
Information Systems Engineering
and Business Intelligence

Vol.9, No.2, October 2023
Available online at: http://e-journal.unair.ac.id/index.php/JISEBI

Ensemble Learning Based Malicious Node Detection in SDN-Based VANETs

Kunal Vermani[1)], **Amandeep Noliya**[2)], **Sunil Kumar**[3)] [*], **Kamlesh Dutta**[4)]

[1)2)3)] *Guru Jambheshwar University of Science and Technology, Haryana, India*
[1)]vermanikunal@gmail.com, [2)]am.noliya@gmail.com, [3)]sunilkaushik27@gmail.com

[4)] *National Institute of Technology Hamirpur, Himachal Pradesh, India*
[4)] kd@nith.ac.in

Abstract

Background: The architecture of Software Defined Networking (SDN) integrated with Vehicular Ad-hoc Networks (VANETs) is considered a practical method for handling large-scale, dynamic, heterogeneous vehicular networks, since it offers flexibility, programmability, scalability, and a global understanding. However, the integration with VANETs introduces additional security vulnerabilities due to the deployment of a logically centralized control mechanism. These security attacks are classified as internal and external based on the nature of the attacker. The method adopted in this work facilitated the detection of internal position falsification attacks.
Objective: This study aimed to investigate the performance of k-NN, SVM, Naïve Bayes, Logistic Regression, and Random Forest machine learning (ML) algorithms in detecting position falsification attacks using the Vehicular Reference Misbehavior (VeReMi) dataset. It also aimed to conduct a comparative analysis of two ensemble classification models, namely voting and stacking for final decision-making. These ensemble classification methods used the ML algorithms cooperatively to achieve improved classification.
Methods: The simulations and evaluations were conducted using the Python programming language. VeReMi dataset was selected since it was an application-specific dataset for VANETs environment. Performance evaluation metrics, such as accuracy, precision, recall, F-measure, and prediction time were also used in the comparative studies.
Results: This experimental study showed that Random Forest ML algorithm provided the best performance in detecting attacks among the ML algorithms. Voting and stacking were both used to enhance classification accuracy and reduce time required to identify an attack through predictions generated by k-NN, SVM, Naïve Bayes, Logistic Regression, and Random Forest classifiers.
Conclusion: In terms of attack detection accuracy, both methods (voting and stacking) achieved the same level of accuracy as Random Forest. However, the detection of attack using stacking could be achieved in roughly less than half the time required by voting ensemble.

Keywords: Machine learning methods, Majority voting ensemble, SDN-based VANETs, Security attacks, Stacking ensemble classifiers, VANETs,

Article history: Received 9 May 2023, first decision 13 July 2023, accepted 18 August 2023, available online 28 October 2023

I. Introduction

Vehicular Ad-hoc networks (VANETs) are reliable communication network that alert drivers about potential collision risks and accidents on the road. VANETs are similar to Mobile Ad-hoc Networks (MANETs), where the nodes are vehicles instead of mobile phones. However, they differ in terms of high node mobility due to the continuous rapid movement of vehicles, frequent changes in network topology, and high bandwidth requirements [1]. VANETs offer a range of advantages, including accident reduction, traffic management, congestion notification, fuel station location, and parking spot suggestions. Alongside these benefits, the performance constraints are due to frequent disconnections among vehicles caused by high mobility, dynamic connectivity, and security attacks [2,3]. Software Defined Network (SDN) integrated with VANETs provide enhanced flexibility and adaptability as the data plane is regulated and controlled by a separate centralized programmable controller. Its architecture significantly reduces the dependence on hardware setup, as previously, even when a small modification in the vehicular network requires hardware upgrades.

[*] Corresponding author

As shown in Fig. 1, SDN-based VANETs architecture comprises three layers, namely the data (forwarding), control (logical), and application (services) planes. The data and control planes are separated using Southbound APIs, while control and application are separated using Northbound. Data is a forwarding plane consisting of moving vehicles and fixed roadside units (RSUs) that communicate wirelessly among vehicles and infrastructure. Control, on the other hand, comprises a software-defined controller (SDNC), serving as the logical brain of the vehicular system. It uses flow tables for packet transmission and decision-making. The application layer is a collection of network-based application services, such as external memory using cloud computing, quality services, routing, and more. It also manages the high mobility and heterogeneous behavior of the network in a programmed and systematic way. However, SDN-based VANETs have a dual nature, and may introduce security risks or new vulnerabilities to the system due to their centralized characteristics [4].

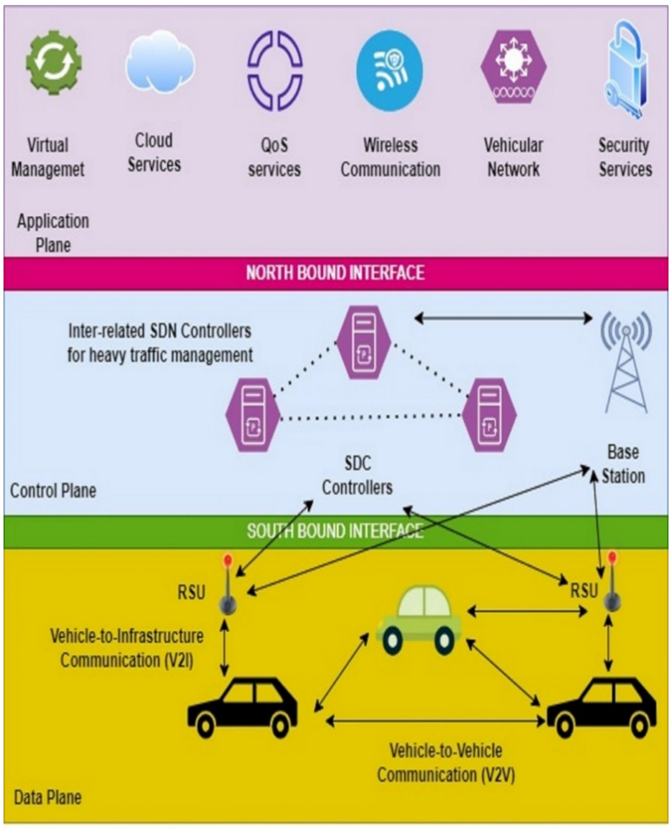

Fig. 1 SDN-based VANETs architecture

Security is a major concern in SDN-based vehicular networking architecture, as the networking system is vulnerable to various attacks, either external or internal. External attacks are carried out by unauthorized individuals and aim to impact the security and communication of the network. They may include phishing, DDoS attacks, and spamming, which can be mitigated using cryptography, a first line of defense. On the other hand, internal attacks are executed by authorized malicious nodes (member nodes of the network) within the data plane. These nodes possess legitimate sources of information, making them challenging to detect using public key-based cryptography method ques. Internal attacks may include position falsification, wrong alert messages, location spoofing, packets dropping, and more. The current integration of machine learning (ML) with VANETs networking is a potential method to analyze the data and protect against these types of attacks.

Khatri *et al.* [5] discussed the challenges of traffic, communication, and safety in VANET systems, as well as the potential solutions offered by ML methods. Several previous investigations presented the use of ML integrated security solutions to enhance the accuracy of attack detection in VANETs [6–8]. Supervised learning classifiers, such as k-NN, SVM, logistic regression, and decision trees are indispensable for identifying anomalous candidates within typical data sets. These methods also play a significant role in intrusion detection for vehicular systems [7-9]. Another popular classification strategy entails using multiple ML classifiers in an "ensemble" to boost performance. This combines the strength of various classifiers and bases the conclusions about the future on their predictions. Voting, stacking, bagging, and boosting are some of the recognized ensemble methods exhibiting higher accuracy rates in classification. Internal attacks pose a greater threat to the functioning of the VANETs system, as authorized users may act maliciously, triggering scenarios of fake alert messages, jamming, collisions, and altered routing suggestions. Ensemble learning algorithms are recommended to effectively address such attacks.

In this motivation, an experimental comparative study was conducted to detect internal attacks in SDN-based VANETs using k-NN, SVM, Naïve Bayes, Logistic Regression, and Random Forest ML classifiers. Subsequently, ensemble learning methods, namely voting and stacking, were used for detection of internal attacks, leveraging the predictions generated from ML classifiers. With the specific aim of detecting the position falsification attacks in SDN-based VANETs, this study used open-source Vehicular Reference Misbehavior (VeReMi) dataset as it is an application-specific dataset for the VANETs environment. All simulations were carried out using the Python programming language. The results showed that the detection of attacks using "voting ensemble" could be achieved in roughly half the time required by "stacking ensemble".

The following are the key contributions of this study: (1) Evaluated the performance of k-NN, SVM, Naïve Bayes, Logistic Regression, and Random Forest ML algorithms in the detection of position falsification attack using VeReMi dataset and (2) Conducted comparative analysis of two ensemble classification models namely voting and stacking, for final decision-making. These methods cooperatively leveraged ML algorithms to improve decision-making.

The subsequent sections a classified as follows: Section II explained the prevailing methods used for attack detection in SDN-based VANETs. Section III presented the proposed method for attack detection. Section IV provided details of the results, evaluation was discussed in Section V, while Section VI concluded the study.

II. Literature Review

The classification of various types of attacks using ML methods in different communication systems has been extensively explored in recent years. With the growth of wireless communication and services, the study community has shown significant interest in designing security solutions for vehicular networks.

Sangwan *et al.* [10] reviewed and categorized various misbehavior assaults in VANETs systems based on architecture, method, node-centricity, and data-centricity. It also discussed the significance of ML methods in the context of misbehavior detection. Ghosh *et al.* [11] presented a comprehensive survey on misbehaving node detection attacks in VANETs, with an emphasis on the adoption of hybrid methods to achieve improved performance in terms of precision and accuracy. Singh *et al.* [12] investigated the effect of normalization using various ML classifiers for detecting position falsification attacks through VeReMi dataset. The experimental study showed that normalized SVM outperformed logistic regression, both with and without normalization. Sonker *et al.* [13] also introduced a method for selecting the optimal algorithm to detect malicious node attacks, with Random Forest achieving a higher accuracy rate of 97.62%.

The extraction of features is a crucial step in dataset processing, as it significantly influences the accuracy of the prediction model. So *et al.* [14] presented a plausibility check score with ML-based scheme for misbehavior detection in VANET, and showed that k-NN and SVM classifiers performed better. Similarly, Gyawali *et al.* [15] devised a new MDS (misbehavior detection system)-based ML algorithm for detecting position falsification attacks [15]. Grover *et al.* [16] proposed ML classifier-based malicious node detection method, using various feature attributes, such as geographical position, speed deviation, acceptance range of RSU, and received signal strength (RSS). The results showed that Random Forest and J48 outperformed other classifiers, including Nave Base, IBK, and Adaboost1. Montenegro *et al.* [17] designed a trust-based model for position falsification attacks based on the k-NN classifier model. In this model, position and received power coherency were used in detecting misbehaving VANETs nodes. The results showed a maximum accuracy rate of 95.8% for the validation model and 84.4% for the random offset position attack. Sultana *et al.* [18] proposed ML-based scheme with local and global detection levels for Constant attack Type 1. Kang *et al.* [19] investigated the role of a deep neural network (DNN)-based intrusion detection scheme, with special emphasis on the in-vehicle system over inter-vehicle communication. Bangui *et al.* [20] proposed hybrid methods comprising Random Forest classifier and clustering-based algorithm to detect both known and unknown attacks. In comparison to conventional ML classifiers, the hybrid method substantially improved detection efficiency.

Multiple ensemble classifiers have also been evaluated for the detection of different types of attacks in VANETs environment. Ercan *et al.* [21] emphasized the significance of the stacking method for validating newly designed features under various vehicle densities. Ghaleb *et al.* [22] introduced a collaborative intrusion detection system (MA-CIDS) that used ensemble learning to enhance attack detection efficacy in VANETs models. Using the network security laboratory-knowledge discovery data mining (NSL-KDD) dataset, locally trained and weighted classifiers were thoroughly evaluated. The results showed a false positive rate of 4% and F1 score of 97%. Khan *et al.* [23]

developed an ensemble-based voting classifier for intrusion detection that incorporated multiple base classifiers, showing a 96% accuracy rate for GPS detection compared to standard ML algorithm. Similarly, Azam *et al.* [24] used majority voting to detect sybil attacks in the VANET system. Standard ML classifiers, including k-NN, Nave Bayes, Decision tree, SVM, and Logical Regression, were used within the majority voting framework, with the proposed scheme achieving a 95% degree of accuracy. Sonker *et al.* [25] designed an algorithm that combined stacking and the bagging algorithms Random Forest and Xgboost. The proposed scheme attained a 98.44% detection rate of misbehaving nodes. The study showed that ensemble classifier could significantly enhance the scheme performance for detecting misbehaving nodes. A comprehensive framework for detecting internal intrusions is presented in the next section.

III. METHOD

This section presents the proposed framework for detecting malicious nodes and identifying position falsification attacks using VeReMi dataset. Fig. 2 presents the proposed malicious node detection framework, comprising Levels 1, 2, and 3, using voting and stacking ensemble learning methods.

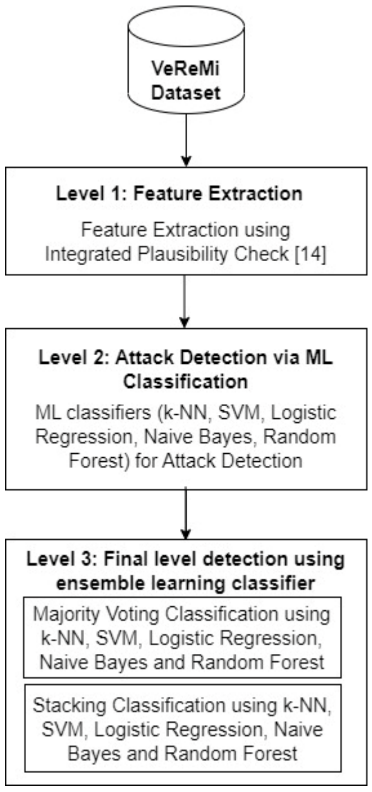
VeReMi Dataset
Level 1: Feature Extraction
Feature Extraction using Integrated Plausibility Check [14]
Level 2: Attack Detection via ML Classification
ML classifiers (k-NN, SVM, Logistic Regression, Naive Bayes, Random Forest) for Attack Detection
Level 3: Final level detection using ensemble learning classifier
Majority Voting Classification using k-NN, SVM, Logistic Regression, Naive Bayes and Random Forest
Stacking Classification using k-NN, SVM, Logistic Regression, Naive Bayes and Random Forest

Fig. 2 Proposed Framework

Level 1: Feature extraction with Plausibility check, *Level 2:* Attack detection using k-NN, SVM, Naïve Bayes, Logistic Regression, and Random Forest ML classifiers, *Level 3:* Ensemble learning (voting and stacking) methods of the classifiers were deployed to improve the classification accuracy and attack detection times. The proposed algorithm was inspired by an extensive literature review and experiments conducted by [14,24,25,27].

A. Dataset Preparation and Analysis

The experimental simulations were conducted using the open-source VeReMi dataset, which comprised 5 position falsification attack scenarios, each repeated 5 times. These repetitions provided a comprehensive representation of various real-time scenarios of misbehaving vehicular nodes [26]. The dataset is presented in Table 1.

TABLE 1
DATASET DESCRIPTION

Attack ID	Attack Type	Parameters
Type 1	Constant Attack	x=5560, y=5820
Type 2	Constant Offset Attack	Δ x = 250, Δ y = -150
Type 4	Random Attack	Uniformly Random
Type 8	Random Offset Attack	Δ x = -300, Δ y = 300
Type 16	Eventual Stop Attack	Stop probability + = 0.025 each position update

B. *Feature Extraction*

The raw VeReMi dataset was processed for the implementation and evaluation of ML algorithm using the algorithm proposed in [14]. The vehicles communicated by broadcasting their details every 100 ms, including speed in the x-y direction, current location, acceleration, velocities, and more, through Basic Safety Messages (BSM). The type of attack was labeled from 0 to 5, where '0' represented a legitimate vehicle, and '1-5' corresponded to different attack types. In the final dataset, the first two features pertained to the accuracy of the sender data and the movement of the sending vehicles, while the remaining four described their behaviors. The location plausibility check was based on previous velocity in the BSM message, GPS location, and average accelerations, using (1) and movement.

$$\text{predicted}_{(x,95)} = x_i + \Delta t\left(v_{(x,i)} + a_{(x,95)} * \Delta t\right) \quad (1)$$

The movement plausibility check determined whether the speed fell below predefined speed threshold. This was calculated using the total displacement between the previous and current locations, velocity, and time obtained from BSM messages. When the total displacement is 0 but the average velocity is not, the value of the score would be 1, otherwise it would be set at 0. The plausibility score fell within the range of [0,4], and represented the total sum of plausibility scores for the x and y coordinates, indicating the number of misbehaving vehicles. These scores were calculated for the x and y coordinates within the pre-defined confidence interval of 95% and 99%. To estimate the behavior, the displacement, magnitude and velocity of vehicles in x and y directions were computed. Features 3 and 4 represented the differences between the calculated average velocities based on total displacement and time, as well as the predicted average velocities based on the reported velocity and time in the *x* and *y* directions, respectively. Feature 5 represented the magnitude of Features 3 and 4, while Feature 6 was the total displacement between the calculated distance and the predicted total displacement based on average velocity. For advanced attack detection, these extracted features were used to classify legitimate from misbehaving vehicles using various learning-based ML classifiers.

C. *Machine Learning (ML) based Attack Classification*

In the attack detection level of the proposed method, the data were cleaned and parsed for training and testing. Multiple classifiers, including k-NN, SVM, Naïve Bayes, Logistic Regression, and Random Forest, were comparatively applied for internal attack detection.

1) *k-Nearest Neighbor (k-NN) Classifier*

k-NN is a well-known supervised learning algorithm for detecting attacker vehicle using Euclidean distance measurement. The Euclidean distance between two vehicles at position (x_i, y_i) and (x_j, y_j) is calculated using (2).

$$d = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}. \quad (2)$$

2) *Logistic Regression Classifier*

Logistic Regression is a classifier that uses a sigmoid function to separate data points. The decision boundary, where the sigmoid function f(x) equals 0.5, defines the border between the classes. The equation is given in (3).

$$f(x) = \frac{1}{1+e^{-y}}. \quad (3)$$

3) *SVM Classifier*

Support Vector Machine (SVM) is a supervised learning-based classifier that uses multidimensional hyperplanes to segregate data points. The decision boundary relies on various support vectors that define the extreme maximums and minimums.

4) *Naïve Bayes Classifier*

The Naïve Bayes Classifier is based on Bayes theorem and predicts data based on the probability of the object. It calculates probabilities using the following equation, where P(X) is the independent probability of X, P(Y) is the independent probability of Y, P(X/Y) that is seen on (4) is the probability of X when Y is true, and P(Y/X) is the probability of Y when X is true.

$$P(\frac{Y}{X}) = \frac{P(\frac{X}{Y}) * P(Y)}{P(X)}. \quad (4)$$

5) *Random Forest Classifier*

Random Forest Classifier is method based on decision trees. To classify the attacks, the dataset was divided into smaller sets, and each decision tree provided a class for input data. The classifier subsequently processed the information and selected the most voted prediction as the output. It is capable of handling large datasets with high dimensionalities and less prone to overfitting. Therefore, to evaluate its impact on the attack detection rate, the RF classifier was applied to the processed features extracted from VeReMi dataset. Random Forest classifier can be seen in Fig. 3.

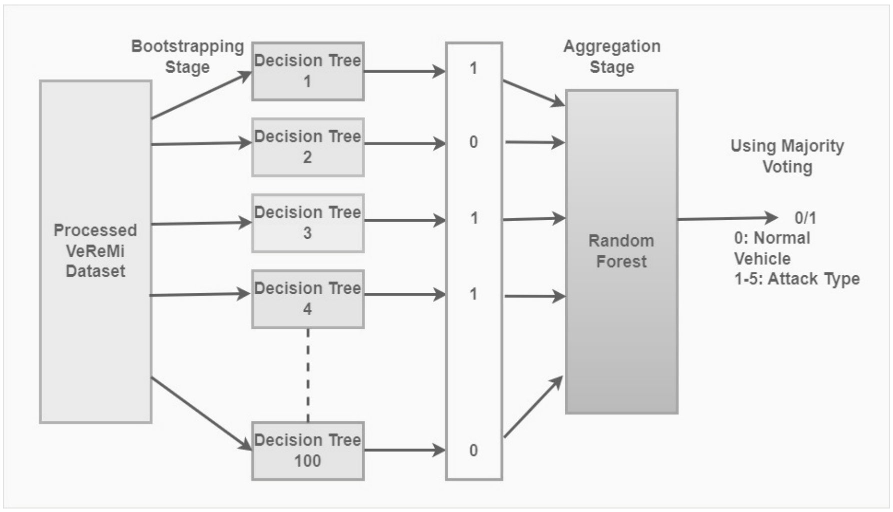

Fig. 3 Random Forest classifier

D. *Final Level Detection*

Two different methods were compared at this level, based on prediction time. The predictions of the classifiers previously mentioned were used to design both majority voting and stacking classifiers. The majority voting is an ensemble classifier that uses the predictions of different classifiers based on assigned weightage. Meanwhile, the stacking ensemble generates a new meta-classifier using the input classifiers and incorporates their characteristics. The entire concept is summarized in the Algorithm 1 and Fig. 4.

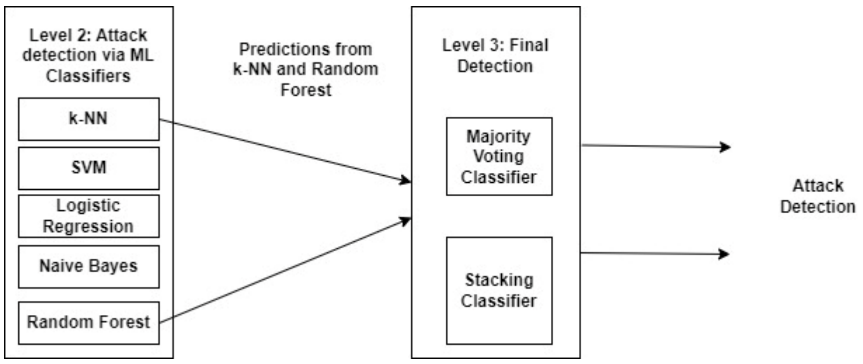
Level 2: Attack detection via ML Classifiers
k-NN
SVM
Logistic Regression
Naive Bayes
Random Forest
Predictions from k-NN and Random Forest
Level 3: Final Detection
Majority Voting Classifier
Stacking Classifier
Attack Detection

Fig. 4 Final level detection

Algorithm 1
Ensemble Learning based malicious node detection in SDN based VANETs

```
function plausibility and consistency check ()
        get each vehicle (Vm € V) where m= 1,2,3…, n
        for each vehicle do feature extraction
                perform data cleaning on the obtained PCR
                        train multiple classifiers including k-NN, SVM,
                              Logistic Regression, Naïve bayes, Random Forest
                              for attack detection
        end for

             perform testing and metric measurement
                        get results of each classifier
  for each classifier
     train Majority Voting and Stacking classifiers
  end for
        perform testing and metric evaluation
return final decision for attack detection
```

The results were evaluated in terms of parametric metrics, such as precision, recall value, F1-score, accuracy, and prediction time. The precision showed the number of true positives predicted positive attack (TP and FP) predictions. The precision value ranged between 0 and 1, representing the specificity of the model. The equation used to calculate precision can be seen on (5).

$$\text{Precision} = \frac{\text{TP}}{\text{TP+FP}}. \tag{5}$$

Recall represents the sensitivity of the model and indicates the true positives that are correctly identified. The formula used to calculate recall is shown in (6). Similar to precision, it also ranges from 0 to 1, with higher values indicating better performance.

$$\text{Recall} = \frac{\text{TP}}{\text{TP+FN}}. \tag{6}$$

The F-Measures shown in (7) that is also known as F1-Score, defines the relationship between precision and recall, with the aim of striking a balance.

$$\text{F} - \text{Measure} = \frac{2\times\text{Precision}\times\text{Recall}}{\text{Recall+Precision}} \tag{7}$$

Accuracy, also called classification rate, measures the number of correct predictions over the total predictions in the dataset, and is calculated as the ratio of the total sum of TP and TN over the total predictions. The formula of accuracy is shown in (8).

$$\text{Accuracy} = \frac{\text{TP+TN}}{\text{TP+TN+FP+FN}} \tag{8}$$

IV. Results

This section presents the result and analysis of the proposed scheme with respect to detection of different attack types in VeReMi dataset. The performance metrics precision, recall, and F1-score of different classifiers across all specified attacks are shown in Tables 2-4.

TABLE 2
Comparative Analysis in Terms of Precision

Attack ID	Naïve Bayes	SVM	Logistic Regression	k-NN	Random Forest	Majority Voting	Stacking Classifier
Type 1	0.72	0.82	0.74	0.96	0.99	0.99	0.99
Type 2	0.53	0.57	0.47	0.63	0.65	0.65	0.70
Type 4	0.76	0.78	0.76	0.86	1.00	1.00	1.00
Type 8	0.60	0.87	0.83	0.90	0.95	0.95	0.95
Type 16	0.62	0.76	0.58	0.78	0.98	0.98	0.99

Table 2 shows that Random Forest performs best in the detection of all attack types with precision values of 0.99, 0.65, 1.0000, 0.95, and 0.98, respectively. While both the majority voting and stacking classification yielded similar results for Random Forest, there was a slight improvement in precision values for types 2 and 16 when using stacking.

TABLE 3
COMPARATIVE ANALYSIS IN TERMS OF RECALL

Attack ID	Naïve Bayes	SVM	Logistic Regression	k-NN	Random Forest	Majority Voting	Stacking Classifier
Type 1	0.78	0.77	0.75	0.96	0.98	0.98	0.99
Type 2	0.51	0.50	0.50	0.55	0.60	0.60	0.66
Type 4	0.82	0.84	0.80	0.82	1.00	1.00	1.00
Type 8	0.52	0.78	0.51	0.84	0.95	0.95	0.96
Type 16	0.66	0.78	0.50	0.77	0.98	0.98	0.99

In Table 3, recall value, representing the efficiency of detecting the positive cases, indicated that the stacking-based method outperformed other traditional classifiers. However, all the classifiers exhibited significantly low recall values in detecting Constant Offset Attacks (Type 2). The lower recall rate of these classifiers signified a higher False Negative Rate (FNR), making it challenging to trace malicious nodes with offset errors. Interestingly, the F1-Score for types 1, 4, 8, and 16 were 0.99, 1.00, 0.95, and 0.98, respectively. Similar to recall value, stacking had a lower F1-Score of 0.65, as shown in Table 4.

TABLE 4
COMPARATIVE ANALYSIS IN TERMS OF F1-SCORE

Attack ID	Naïve Bayes	SVM	Logistic Regression	k-NN	Random Forest	Majority Voting	Stacking Classifier
Type 1	0.72	0.79	0.74	0.97	0.99	0.99	0.99
Type 2	0.46	0.53	0.48	0.54	0.61	0.61	0.65
Type 4	0.73	0.80	0.77	0.83	1.00	1.00	1.00
Type 8	0.25	0.82	0.45	0.86	0.95	0.95	0.95
Type 16	0.62	0.76	0.54	0.74	0.98	0.98	0.98

Fig. 5 shows the general performance of the classifiers in terms of accuracy. The stacking classification yielded similar results as Random Forest, outperforming other classifiers with accuracies of 99.16%, 75.73%, 99.78%, 96.48%, and 95.54% for Types 1, 2, 4, 8, and 16, respectively.

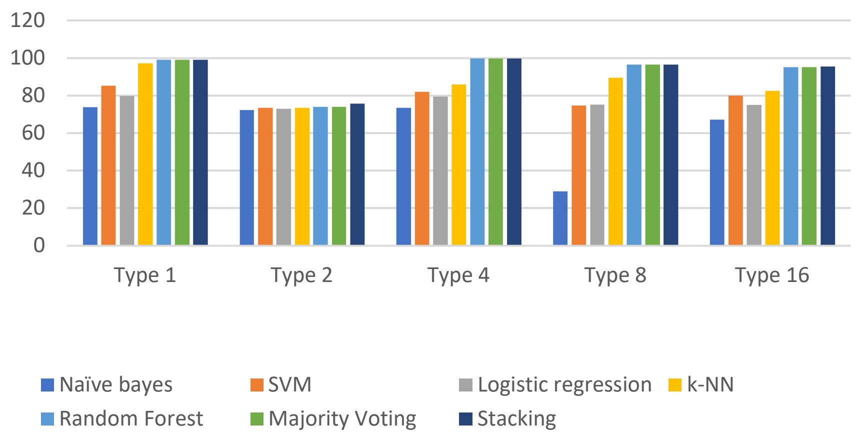

Fig. 5 Accuracy of k-NN, Random Forest, majority voting and stacking classifiers

The performance of majority voting ensemble and stacking classifier was compared at the third level. The best accuracy and other performance metrics were achieved through stacking of multiple classifiers. For majority voting, various classifiers were assigned weight scores, such as [1,1,1,1,5] for Naïve Bayes, SVM, logistic regression, k-NN, and Random Forest, respectively. While both the methods improved the performance matrices, the prediction time of Stacking Classifier was significantly shorter than the prediction time of the majority voting classification method. Table 5 shows the results for different attack type detection.

TABLE 5
DETECTION OF ATTACKS IN REAL TIME (MILLI SECONDS)

Attack ID	Attack Type	Majority Voting	Stacking Classifier
Type 1	Constant Attack	1.06	0.07
Type 2	Constant Offset Attack	3.92	0.24
Type 4	Random Attack	4.26	0.20
Type 8	Random Offset Attack	4.21	0.16
Type 16	Eventual Stop Attack	4.13	0.17

The ROC curve of the final stage of Stacking Classification is presented in Fig. 6, with an area under curve score (AUC) of 0.985. Therefore, the detection rate of the proposed method was significantly high.

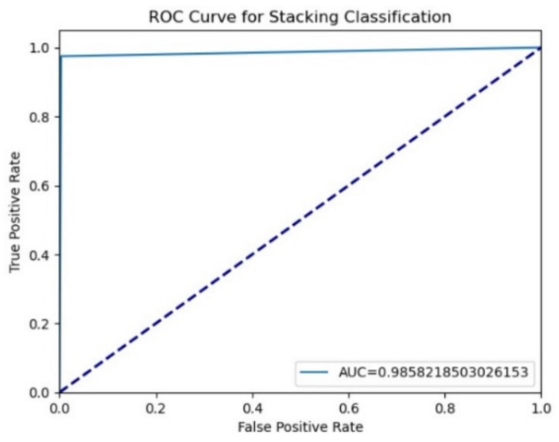

Fig. 6 ROC Curve for stacking classifier

V. DISCUSSION

A multi-level method was proposed to detect the position falsification attack in VeReMi dataset. In the first part of the proposed method, VeReMi dataset was preprocessed, and feature extraction was performed to enhance prediction using plausibility check features [14]. The dataset was cleaned and parsed to train and test various ML classifiers. The results obtained were compared to [14, 15, 17] in terms of precision and recall values, as shown in Table 6. The proposed method showed better performance compared to [14] for all attack types. In comparison to [15,17], the results were significantly comparable, and the work performed better for type 2, incorporating the method used in [15].

TABLE 6
COMPARISON OF DETECTION METHODS

Attack Type	[14]		[15]		[17]		Our Results	
	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall
Constant Attack	0.95	0.83	1.00	0.99	0.86	1.00	0.99	0.99
Constant Offset Attack	0.56	0.19	0.94	0.80	0.91	0.74	0.70	0.66
Random Attack	0.95	0.83	1.00	0.99	0.91	0.88	1.00	1.00
Random Offset Attack	0.96	0.82	0.97	0.95	0.84	0.86	0.95	0.96
Eventual Stop Attack	0.71	0.42	0.98	0.93	0.96	0.77	0.99	0.99

The predictions of the best classifiers were fed into two different ensemble classifiers, namely, majority voting and stacking classifier, to enhance detection and assurance levels in the second part of the algorithm. The accuracy rate of the two classifiers was similar to Random Forest, with a slight improvement in stacking classification. Furthermore, both were evaluated in terms of prediction, with stacking predicting attacks significantly faster than the majority voting classifier for all attack types. This difference was mainly because the majority voting decision simply depended on the feedback from the input classifiers majority voting, while stacking was a weighted voting scheme where the vehicle score was used as a weight for decision-making.

VI. Conclusions

In conclusion, SDN-based VANETs were vulnerable to internal attacks that could significantly impact VANETs services and threaten human lives. This study proposed an ensemble learning-based framework for detecting malicious nodes in the network. In the first part, the performance of various ML classifiers, including Naïve Bayes, SVM, logistic regression, k-NN, and Random Forest, was evaluated for detecting various attack scenarios (Attack Types 1, 2, 4, 8, and 16) in VeReMi dataset. The results showed that Random Forest outperformed all other classifiers in terms of performance metrics, such as precision, recall, F-score, and accuracy. For the final level of detection, ensemble learning-based classifiers, namely "majority voting" and "stacking," were comparatively studied. The results showed that stacking classification improved the performance metrics with lower prediction times compared to majority voting classification. Although these results were significantly promising, the proposed algorithm exhibited relatively low performance metrics for type 2. Therefore, future studies were recommended to design a more adaptive framework for its detection.

Author Contributions: *Kunal Vermani, Amandeep Noliya, Sunil Kumar and Kamlesh Dutta*: Methodology and Conceptualization. *Kunal Vermani*: Implementation and Writing – original draft. *Kamlesh Dutta*: Editing.

All authors have read and agreed to the published version of the manuscript.

Funding: This research received no specific grant from any funding agency.

Acknowledgments: We deeply acknowledge Prof. Dharmender Kumar, Chairperson, Department of Computer Science & Engineering, Guru Jambheshwar University of Science & Technology, Hisar, Haryana (INDIA) for providing the necessary facilities to carry out research work in this department and for his kind support.

Conflicts of Interest: The authors declare no conflict of interest.

Data Availability: The work is the part of a major study; hence the data source is restricted.

Informed Consent: There were no human subjects.

Animal Subjects: There were no animal subjects.

ORCID:
Kunal Vermani: https://orcid.org/0000-0002-9840-9650
Amandeep Noliya: N/A
Sunil Kumar: N/A
Kamlesh Dutta: N/A

References

[1] H. Hartenstein and K.P. Laberteaux, "A tutorial survey on vehicular ad hoc networks," *IEEE Commun. Mag.*, vol. 46, no. 6, pp. 164-171, 2008, doi: 10.1109/MCOM.2008.4539481.

[2] 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, doi: 10.1007/s11235-010-9400-5.

[3] S.K. Bhoi, and P.M. Khilar, "Vehicular communication: a survey," *IET Networks*, vol. 3, no. 3, pp. 204-217, 2014, doi: 10.1049/iet-net.2013.0065.

[4] R. Swami, M. Dave, and V. Ranga, "Software-defined Networking-based DDoS Defense Mechanisms," *ACM Comput Surv*, vol. 52, no. 2, pp. 1-36, 2020, doi: 10.1145/3301614.

[5] S. Khatri, H. Vachhani, S. Shah, J. Bhatia, M. Chaturvedi, S. Tanwar, and N. Kumar, "Machine learning models and techniques for VANET based traffic management: Implementation issues and challenges," *Peer Peer Netw Appl*, vol. 14, no. 3, pp. 1778-1805, 2021, doi: 10.1007/s12083-020-00993-4.

[6] A. di Maio, M.R. Palattella, R. Soua, L. Lamorte, X. Vilajosana, J.A. Zarate, and T. Engel, "Enabling SDN in VANETs: what is the impact on security?" *Sensors (Switzerland)*, vol. 16, no. 12, 2077, 2016, doi: 10.3390/s16122077.

[7] M. Mousa, A.M. Bahaa-Eldin, and M. Sobh, "Software Defined Networking concepts and challenges," in *11th International Conference on Computer Engineering & Systems (ICCES)*, pp. 79–90, 2016, doi: 10.1109/ICCES.2016.7821979.

[8] K. Adhikary, S. Bhushan, S. Kumar, K. Dutta, "Hybrid algorithm to detect DDOS attacks in VANETs", *Wireless Personal Communications,* vol. 114, pp. 3613-3634, 2022, doi: 10.1007/s11277-020-07549-y.

[9] P.C. Sen, M. Hajra, and M. Ghosh, "Supervised classification algorithms in machine learning: A survey and review" in *Emerging Technology in Modelling and Graphics*, pp. 99-111, 2020.

[10] A. Sangwan, A. Sangwan, R.P. Singh, “A classification of misbehavior detection schemes for VANETs: A Survey”, *Wireless Personal Communications*, 2022, doi:10.1007/s11277-022-10098-1.

[11] R. Barnwal, and S. Ghosh, *Detection of misbehaving nodes in vehicular ad hoc network*, 2014.

[12] 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: Communications in Computer and Information Science, pp. 166-178, 2019, https://doi.org/10.1007/978-981-13-7561-3_13

[13] A. Sonker, and R.K. Gupta, “A new procedure for misbehavior detection in vehicular ad-hoc networks using machine learning,” *Int. J. Electr. Comput. Eng.,* vol. 11, no. 3, pp: 2535-2547, 2021.

[14] S. So, P. Sharma, and J. Petit, “Integrating plausibility checks and machine learning for misbehavior detection in VANET,” in *Proceedings 17th IEEE International Conference on Machine Learning and Applications,* pp. 564–571, 2019, doi: 10.1109/ICMLA.2018.00091.

[15] S. Gyawali and Y. Qian, "Misbehavior detection using machine learning in vehicular communication networks," in *IEEE International Conference on Communications (ICC), Shanghai, China*, pp. 1-6, 2019, doi: 10.1109/ICC.2019.8761300.

[16] J. Grover, N. K. Prajapati, V. Laxmi, and M. S. Gaur, “Machine learning approach for multiple misbehavior detection in VANET,” *Communications in Computer and Information Science*, vol. 192, no. 3, pp. 644–653, 2011, doi: 10.1007/978-3-642-22720-2_68.

[17] J. Montenegro, C. Iza, and M.A. Igartua, “Detection of position falsification attacks in VANETs applying trust model and machine learning,” in *PE-WASUN 2020 - Proceedings of the 17th ACM Symposium on Performance Evaluation of Wireless Ad Hoc, Sensor, and Ubiquitous Networks*, pp. 9-16, 2020, doi: 10.1145/3416011.3424757.

[18] R. Sultana, J. Grover, and M. Tripathi, “A novel framework for misbehavior detection in SDN-based VANET,” in *International Symposium on Advanced Networks and Telecommunication Systems,* pp. 1-6, 2020, doi: 10.1109/ANTS50601.2020.9342778.

[19] M. J. Kang, and J.W. Kang, “Intrusion detection system using deep neural network for in-vehicle network security”, *PloS One*, vol. 11, no. 6, e0155781, 2016, doi: https://doi.org/10.1371/journal.pone.0155781.

[20] H. Bangui, M. Ge, and B. Buhnova, “A hybrid machine learning model for intrusion detection in VANET,” *Computing*, vol. 104, pp. 503-531, 2022, doi: 10.1007/s00607-021-01001-0.

[21] S. Ercan, M. Ayaida, and N. Messai, “Misbehavior Detection for Position Falsification Attacks in VANETs Using Machine Learning”, *IEEE Access*, vol. 10, pp. 1893-1904, 2022, doi: 10.1109/ACCESS.2021.3136706

[22] F.A. Ghaleb, F. Saeed, M. Al-Sarem, B.A.S. Al-rimy, W. Boulila, A.E.M. Eljialy, K. Aloufi, and M. Alazab, “Misbehavior-Aware On-Demand Collaborative Intrusion Detection System Using Distributed Ensemble Learning for VANET,” *Electronics*, vol. 9, no. 9, 1411, 2020, https://doi.org/10.3390/electronics9091411

[23] M.A. Khan, M.A. Khan, K.S. Latif, A.A. Shah, M.U. Rehman, W. Boulila, M. Driss, and J. Ahmad. "Voting classifier-based intrusion detection for IOT networks," in *Advances on Smart and Soft Computing: Proceedings of ICACIn*, pp. 313-328, 2021, https://doi.org/10.48550/arXiv.2104.10015.

[24] S. Azam, M. Bibi, R. Riaz, S.S. Rizvi, and S.J. Kwon, “Collaborative learning based sybil attack detection in vehicular AD-HOC networks (VANETS),” *Sensors*, vol. 22, no. 18, 6934, 2022, doi: 10.3390/s22186934.

[25] A. Sonker, R.K. Gupta, “A new combination of machine learning algorithms using stacking approach for misbehavior detection in VANETs,” *International Journal of Computer Science and Network Security*, vol. 20, no. 10, pp. 94-100, 2020, doi.org/10.22937/IJCSNS.2020.20.10.13.

[26] R. W. van der Heijden, T. Lukaseder, and F. Kargl, “VeReMi: A dataset for comparable evaluation of misbehavior detection in VANETs,” Apr. 2018, [Online]. Available: http://arxiv.org/abs/1804.06701