Inteligencia Artificial 25(70), 138-154
doi: 10.4114/intartif.vol25iss70pp138-154

INTELIGENCIA ARTIFICIAL

http://journal.iberamia.org/

# Towards Misbehavior Intelligent Detection Using Guided Machine Learning in Vehicular Ad-hoc Networks (VANET)

Omessaad Slama[1,A] Bechir Alaya[2,C], Salah Zidi[1,B]

[1]Hatem Bettaher Laboratory (IRESCOMATH, University of Gabes, Tunisia)

[A]slama.oumsaad@gmail.com

[B]salah_zidi@yahoo.fr

[2]Department of Management Information Systems and Production Management, College of Business and Economics, Qassim University, Saudi Arabia

[C]b.alaya@qu.edu.sa

**Abstract** C-ITS (Cooperative Intelligent Transport Systems) is a new technology that aids in the reduction of traffic accidents and the enhancement of road safety. VANETs (Vehicular Ad hoc Networks) are an ITS system based on inter-vehicle communication through the transmission of basic safety messages (BSM), which are vulnerable to a variety of misbehaviors. To resolve this challenge, we developed in this paper an automated learning approach-based misbehavior detection system (MDS) to identify and categorize misbehaving messages delivered by a vehicle on VANETs using VeReMi extension database. This study examines different types of classification: in binary classification, all sorts of misbehavior were grouped into one "misbehavior" category; however, in multi-class classification for three classes, misbehavior was divided into two classes: attacks and faults. The classifier has substantial issues when learning from unbalanced data when working with multi-class issues, this gets considerably more complex. The relations between categories are no longer well-defined, and it is easy to lose effectiveness in one class while improving in another. As a result, the findings are uncovinient in the Classic Learning Approach for Multi-class Classification when classifying misbehaviors into different types of misbehaving classes. To solve this issue, we developed a novel and powerful approach named "Guided Learning Approach for Multi-class Classification" to reduce the number of classes by combining comparable misbehaviors into one. According to the results, the Random Forest classifier outperforms other classifiers.

**Keywords**: Misbehavior Detection, Machine Learning, VANETs, Binary Classification, Multi-class classification, Imbalanced data.

**Nomenclature**

MDS Misbehavior Detection System

VeReMi Vehicular Misbehavior Dataset

C-ITS Cooperative Intelligent Transport Systems

RF Random Forest

KNN K-Nearest Neighbors

NB Naive Bayes

LR Logistics Regression

DT Decision Tree

MLP Multi Layer Perceptron

ANN Artificial Neural Network

CNN Convolutional Neural Network

**ISSN: 1137-3601 (print), 1988-3064 (on-line)**

LSTM Long Short-Term Memory

RFE Recursive Feature Elimination

ANOVA One-Way Analysis of Variance

BCA Binary classification approach

MCATC Multi-class Classification Approach for Three Classes

C-LAMC Classic Learning Approach for Multi-class classification

G-LAMC Guided Learning Approach for Multi-class Classification

# 1 Introduction

During these years, the increased number of automobiles on the road made driving dangerous and risky. In addition, damaged roads affect driver and passenger comfort, making driving more challenging [1]. Furthermore, excessive speeding might result in catastrophic accidents. According to the National Inter ministerial Road Safety Observatory, 253 people died on the roads in February 2019, and 5,021 people were injured in traffic accidents in metropolitan France [2].

To increase driving safety and prevent traffic accidents., the automobile manufacturing industry and public authorities have tried to develop Intelligent Transportation Systems (ITS) [3,4]. The Intelligent Transportation System (ITS) has introduced a novel technology named VANETs.

VANETs [5,6] are composed by different infrastructures: a vehicle which equipped by an OnBoard Unit (OBU) that regularly broadcasts their status to its neighbors, a Road-side Unit (RSU) [7] is an infrastructure installed on the side of the road that communicates with nearby communicating OBUs by broadcasting traffic information, weather conditions, and helping OBUs connect to the Internet. Another infrastructure called Central Authority (CA) providing services such as registration and management of all network entities (OBU and RSU) [8] as well as the revocation of certificates in the event of misbehavior [9]. There are different forms of communication in vehicular networks such as Vehicle-to-vehicle (V2V), vehicle-to-infrastructure (V2I), and RSU-to-RSU (I2I) communication [10,11,12] as shown in Fig. 1. The interaction between vehicles (V2V) is ensured by broadcasting every 100 ms a Basic Safety Messages (BSMs) [13] , according to the SAE J2735 standard, containing the current position, acceleration, speed, heading, and other details of each vehicle. The basic purpose of vehicular networks is to ensure passenger safety, conductors, and vehicles through the exchange of private accident and traffic information among vehicles [14]. However, malicious vehicle using VANET might disrupt vehicle communication by broadcasting incorrect information and fake alarms [15,16].

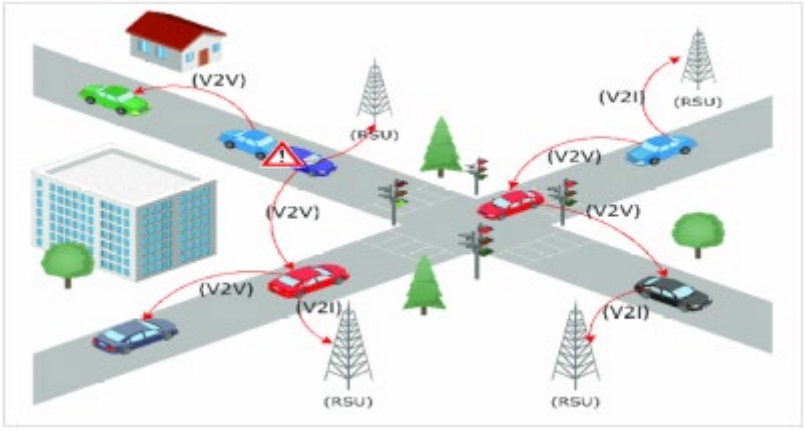


**Figure 1**: Types of communication in VANETs

To address this issue, a public key infrastructure (PKI) delivers certificates, as described by the IEEE 1609.2 protocol [17,18] to vehicles which used to sign and validate each message transmission to assure authenticity, integrity, and privacy. Nevertheless, despite the PKI's protection, the security risk still exists.

Hence, there is a larger requirement to provide an effective misbehavior detection system (MDS) to discover and categorize misbehavior in vehicular networks.

The automatic learning approach has yielded promising results in the area of anomaly detection. Because ML qualified classifiers can learn complicated patterns and behaviors, they are a good solution for identifying data that departs from behavioral

patterns. This research examine the possibility of a model based on supervised Machine Learning that uses Basic Safety Message information to determine if the activity is genuine or misbehaving and to categorize between various types of misbehaviors.

During this experimental study, we used the basic steps: we choosed a new database named VeReMi extension [19]. The database was thoroughly analyzed by removing the flow identification features to eliminate overfitting. The VeReMi extension dataset also has a variety of issues, including difficulty with imbalanced data, categorical attributes, and inconsistent data. For challenges involving the VeReMi extension network database, we provide a hybrid version. The expertises are assessed using Random Forest, Naive Bayes, K-Nearest Neighbors, Decision Tree, Multi-Layer Perceptron, Logistic Regression machine learning classifiers, Artificial Neural Network, Long Short Term Memory and Convolutional Neural Network Deep Learning classifiers, then tested by VeReMi extension database.

Our paper's primary contributions are as follows: First, we suggest a Misbehavior Detection System (MDS) based on ML for VANETs that includes assessment criteria that are comparable to prior research. A realistic VeReMi extension dataset is used to test the efficacy of the suggested model. As assessment measures, precision, recall, and F1-score are employed. Then, we address issues with the VeReMi extension dataset, like missing data, category unbalance, and inappropriate attributes that have an impact on the MDS model's efficiency. The best attributes of the database were chosen using the RFE and ANOVA methods. Then, while working with multi- class data, we develop a novel approach to solving the problem of learning from imbalanced data. Finally, we test multiple machine learning classifiers to distinguish between benign and malicious occurrences. The following is how the rest of the article is organized: Some related works adapted on misbehavior detection systems in VANETs and our proposed ML and DL approaches are included in Section 2 In section 3, we describe the database utilized and the proposed methodology for detecting misbehavior developed in this study as well also we explains diverse features as well as the feature selection methods, which are utilized in the categorization of misbehaviors in the vehicular environment. The outcomes of experiments are discussed in Section 4.The paper comes to a close with Section 5.

## 2 Related Works

In this section, we describe several related studies based on misbehavior detection systems in VANETs, as well as the suggested ML and DL approaches

### 2.1 ML adapted to VANET

In this part, we provide a short overview of machine learning based MDS approaches in the context of Cooperative Intelligent Transport Systems. A study titled Detecting Malicious Nodes in VANETs was published by [20], they presented a new approach named Detection of Malicious Nodes. They used a network simulator to conduct the experiment [21] applied Naive Bayes, IBK, AdaBoost1, and J-48 RF to identify misbehavior in vehicular ad hoc networks. The greatest outcomes were from random forest and J-48. There were 3101 genuine samples and 1427 attacker samples in the dataset. The findings were based on measures with high TPR and TNR values of 0.93 and 0.99, respectively, and low FPR (0.005) and FNR (0.06) values. [22] introduced a method of misbehaving identification based on ensemble learning in vehicular networks, once again in 2012. Naive Bayes, IBK, AdaBoost1, J-48, RF, and ensemble based learning were among the algorithms utilized. Using ensemble based learning, the most accurate were TPR (0.95), FPR (0.01), and TNR (0.99), FNR (0.03) [23] introduced research on detecting misbehaving vehicles in Vehicular environnement via trust management. They present some misbehavior prevention studies in location privacy-enhanced VANETs. They want to optimize the detection rate of the suggested system in the future, as well as analyze the efficiency of the suggested method with various node densities and average speeds.

Barnwal and Ghosh published a research on detecting misbehaving nodes in VANETs, concluding that hybrid-based approaches for misbehavior detection should be used [24]. Sedjelmaci et al. proposed a study [25] on predicting and preventing unruly attackers in heterogeneous vehicular networks (HetVNet) to avoid the occurrence of the most serious assaults that target HetVNet. They used NS-3 to assess the efficiency of the suggested method, which demonstrated a high accuracy prediction rate, a short detection time, and a small communication overhead.

A survey of misbehaving vehicle identification in VANETs was published by [26], when compared to detection approaches based on SVM, Dempster-Shafer, and averaging, the accuracy of the SVM algorithm is the best.

Tiwari and Gupta studied [27] the use of hash functions to improve the security of misbehaving vehicles in VANETs; they tested J-48, RF, IBK, Naive Bayes, and AdaBoost classifiers. However, J-48 and RF produced the greatest results. In [28], Yan et al. introduced a model for identifying location-related misbehaviors called position verification. Steven et al. published a paper [29] that used KNN and SVM ML classifiers to detect and categorize location spoofing attacks on the VeReMi dataset, with satisfactory results. They developed a framework that identifies and categorizes misbehavior. Another study [30] proposed by Singh et al. looked at a ML approach for detecting position falsification attacks in VANETs. They investigated various feature extractions that gave variable accuracy outcomes using Logistic Regression and SVM machine learning algorithms. Their results show that SVM exceeds Logistic Regression. In their paper [31], Waleed et al. presented Optimized Node Clustering in VANETs utilizing Meta- Heuristic methods.

### 2.2 Overview of Machine Learning/Deep Learning

The VeReMi extension dataset has been used to test a variety of ML approaches. During the feature selection step of the misbehavior detection phase, the chosen algorithms are utilized to train and test with various parameters. Precision, recall, and F1-score were employed to assess the various algorithms. Since these techniques are widely utilized in the field of cybersecurity, they were evaluated. The methodologies employed have demonstrated good performance in the production of MDS. We look at

the RF, KNN, Naive Bayes, Logistic Regression, Decision Tree, MLP classifiers ML qnd ANN, CNN, LSTM classifiers DL. This is an overview of these techniques:

- **K-Nearest Neighbors (KNN):** KNN is a supervised machine learning classifier that assigns unlabeled observations to the category of the most identical labeled instances using the Euclidean distance to compute their distance from the neighbors as shown in equation (1). Fig. 2 explains the KNN classifiers to classify new unlabeled observations. The blue circles represent the normal traffic class and the green squares depict the misbehaving class. Every unlabeled observation (orange hexagon) will be categorised according to the number of the two classes' closest distant neighbors. KNN is widely used since it is simple to construct and can handle multiclass scenarios [32].

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

Where d (x, y) is the Euclidean distance between the two sampling, $x_i$ is the first observation, $y_i$ is the second sample of the data, and n is the number of observations.

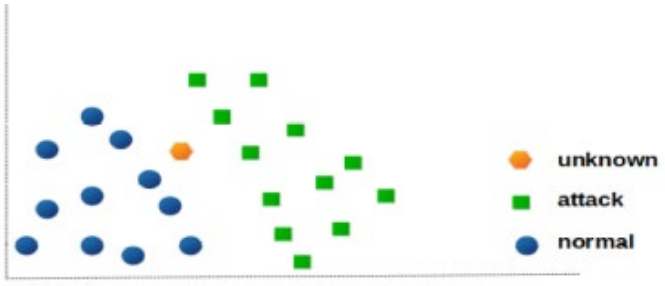


**Figure 2:** K-Nearest Neighbors Algorithm

- **Random Forest (RF):** It is a supervised ML classifier that includes many decision trees. To classify a novel sample, it's necessary to assemble the classifications of every decision tree model and then select the most polled prediction as a result. Random Forest produces a strong and precise outcome that requires less input, does not require the feature extraction process [33] is not susceptible to over-fitting. This algorithm, according to some academics, outperforms SVM, ANN, and KNN in specific sorts of attack detection, and it's also applicable to MDS. The Random Forest has the benefit of being able to be used in both regression and classification situations.

- **Decision Tree (DT):** It is a supervised ML classifier [34] employed for categorization and regression on a database by applying ensembles of decisions. It has a tree structure with each node representing a feature, each branch representing a decision, and each leaf representing a class.
To avoid over-fitting, the classifier chooses the best features then deletes irrelevant branches from the tree. This approach requires less work for data preparation in pre- processing.

- **Logistic Regression (LR)** It is a supervised ML approach used to solve multivariate classification problems [35] like misbehavior detection, although it can predict the likelihood of an observation being assigned to a particular category. If the estimated probability is greater than the stated threshold, the algorithm may predict that the occurrence will match the misbehavior. Moreover, it will assume the occurrence is a genuine category. Logistic regression determines the probability using equation (2).

$$h_\theta(x) = \sigma(\theta^T X) \quad (2)$$

Where $h_\theta$ is the hypothesis, x is the given input vector,
θ is the Logistic Regression settings, and σ (r is the threshold definition using a sigmoid function. The sigmoid is described as follows as shown in equation (3):

$$\sigma(r) = \frac{1}{1+e^{-r}} \quad (3)$$

The output is somewhere in the middle (0:1), with r being the term ($\theta^T$) from the preceding equation).

- **Naïve Bayes (NB)** It's a supervised machine learning method that predicts the probability of a phenomenon appearing based on previous observations of similar phenomena. NB is a basic, easy-to-implement classifier that is employed in a variety of MDSs and is widely used in machine learning to categorize normal and attacker nodes [36] in large databases. Naïve Bayes demands small examples for training [37], having the ability to classify binary and multi-label sorting. The

interdependencies between attributes for classification aren't taken into consideration because they influence the performance of the model [38].

- **Multi Layer Perceptron (MLP)** MLP [39] is a simple deep learning algorithm, as well famed as feed-forward neural networks, having one or more hidden layers and one of the feed-forward ANN. MLP contains three or more layers (input, output, and hidden layers) where every node is connected with some weight $W_{ij}$. In MLP, the full neurons of one layer are totally connected to all other neurons of the following layer. If the MLP classifier is without any hidden layers, so it's amounting to LG.

- **Artificial Neural Network (ANN)** ANNs [40] are computer programs that mimic the learning process of the human brain. ANNs, like people, learn by experience with appropriate learning instances rather than through programming. They're computer representations of human brains.

- **Convolutional Neural Network (CNN)** It is [41] a deep learning classifier. CNN is the most complex and widely used for reducing the number of data inputs. It is made up of three layers: input (convolutional layer), hidden (pooling layer), and output (activation unit) presented in Fig. 3 ; each neuron in the layer is linked to a small zone of neurons in the input data. The input layers utilize diverse kernels to convoluting data inputs. The hidden layers minimize dimension examples, hence reducing the sizes of the next layers.
  The output layers launch an activation function on each feature in the feature ensemble in a non- linear fashion [42]. CNN, on the other hand, requires a lot of computer capacity to train and extract features from raw data in a short amount of time.

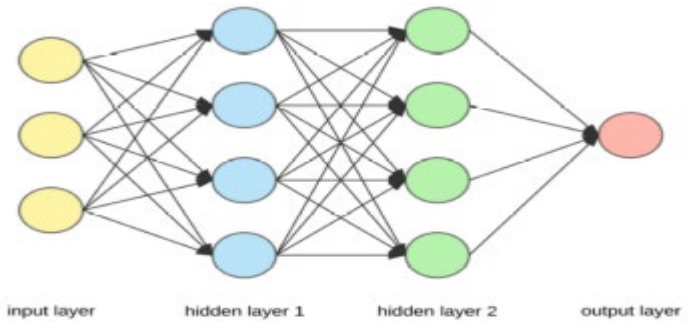


**Figure 3:** Structure of the CNN model

- **Long Short-Term Memory (LSTM)** LSTMs are a complex area of deep learning and are a type of recurrent neural network (RNN) [43] that has been proven to be highly efficient at identifying time series data because of the feedback loops in their design that can recall temporal data [44]. As a result, they're useful in situations like intrusion detection and voice recognition that need temporal sequence categorization.

# 3 Proposed Methodology for Detecting Misbehavior

In this part, we detail the database used in this study, as well as the suggested approach for detecting misbehavior. We also explain the diverse features and feature selection methods used to classify misbehaviors in the vehicular environment.

## 3.1 VeReMi extension database

The database utilized in this investigation is the VeReMi extension. The dataset includes a set of Basic Ssfety Messages (BSM) in a network with varied traffic situations.
Ground truth and log files in the type of JSON files are included in every simulation. In a simulation, just one ground truth file exists that comprises a vehicle's real network behavior, a misbehavior type, which distinguishes genuine nodes from misbehaving nodes. In a simulation, the number of log files equals the number of network nodes. Every node in vehicular environment produces a log file containing all of the Basic Safety Messages it receives from the other devices during its entire journey.

In one simulation, all JSON files were transformed to CSV format before being merged. The misbehavior kind and receiver identity features were appended to the generated merged CSV file. The "misbehavior kind," which labels misbehaviors between 0 and 19, where 0 represents a genuine vehicle and 1-19 represents various forms of misbehavior, was transferred from the initial Ground File to a merged CSV file. The "Receiver Identity" feature was created expressly to create the source and destination pair.

We distinguish between malfunctions (faults) and attacks in the "misbehavior kind" attribute: the fault is caused by a defective OBU or vehicle sensors producing the wrong location, velocity, acceleration, and heading values, whereas the attack is generated by the vehicle sending fake information.
The various faults are: (constant, random, constant offset, random offset) position, (constant, random, constant offset, and random offset) velocity, and Delayed Messages.

The various types of cyberattacks are: Denial of Service (DoS), DoS random, DoS disruptive, DoS random sybil, DoS disruptive sybil, data replay, disruptive, data replay sybil and traffic congestion sybil. More details are in [45].

There are 30 features, 20 classes, and 3194808 samples in the database, including 1900539 legitimate samples and 1294269 misbehavior samples.
Fig. 4 and Fig. 5 demonstrate the legitimate and misbehavior statistics for networking data gathered in the train and test VeReMi extension database. Unbalanced data poses a difficulty during classification, which becomes much more challenging when dealing with multi-class issues.

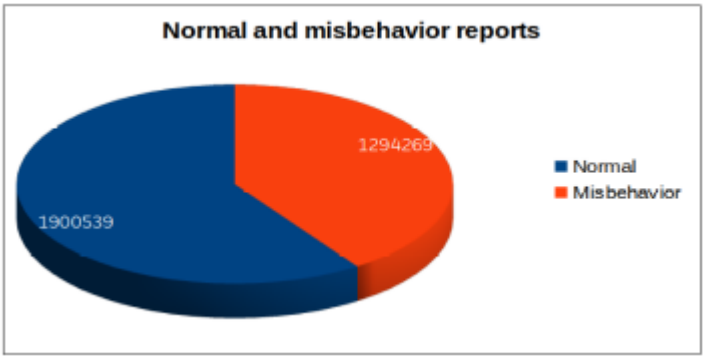


**Figure 4:** Statistics reports of the network VeReMi extension database

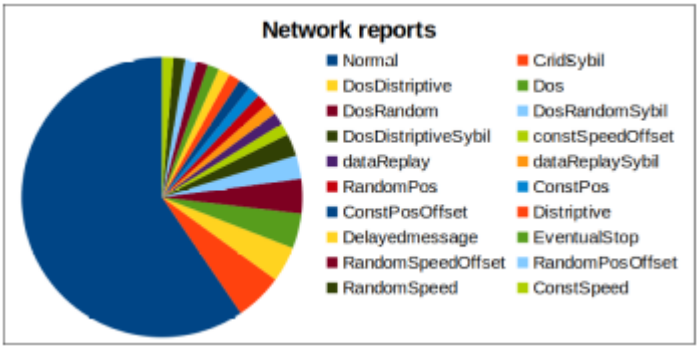


**Figure 5:** Statistics reports of the network VeReMi extension database

The classification task is among the most difficult problems in data mining, especially when learning from unbalanced data. When working on multi-class situations, it'll become considerably more complex. Relations across categories are no longer well-defined, and it's simple to lose efficiency in one while improving in another. Although the ML community has been more interested in this area in recent years, as well as a requirement to create innovative and effective methods to deal with this issue. We provide a novel method in this study called "Guided Learning Approach for Multi-class Classification" (G-LAMC) to deal with multi-class unbalanced data challenges.

The suggested methodology in this study seeks to identify and classify misbehaviors by examining binary and multi-class classification on Vehicular networks with ML classifiers. Every 100 ms, any vehicle diffuses BSMs including its location, velocity, direction, and some other pertinent data to the neighboring infrastructure and vehicles. The information contained in these BSMs can aid in identifying the features and activities of an assault vehicle in VANET.
We use the processes described in Fig. 6 to develop an intelligent Misbehavior Detection System able to categorize vehicle as benign or a attacker.

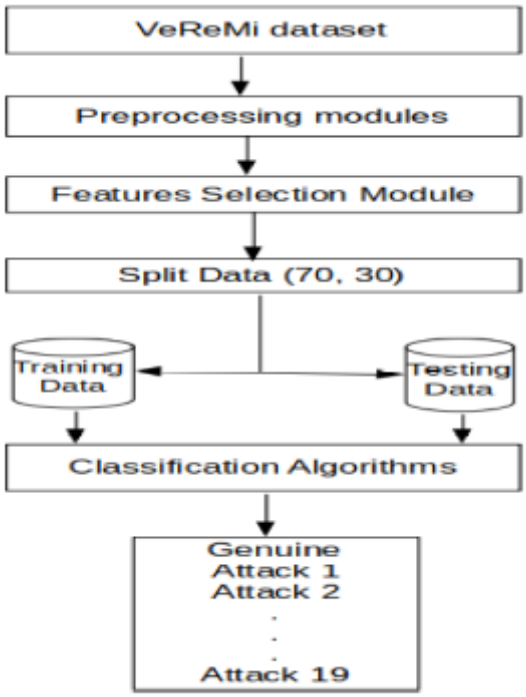


**Figure 6:** Steps in the proposed MDS approach

## 3.2 Feature selection methods

Choosing the relevant attributes subset with informative attributes is referred to as feature selection. That's the most crucial phase in the ML process since it helps to minimize overfitting and improve model generalizability [46]. For feature selection, we propose three approaches: method 1 (which studies the influence of each feature on the labeled class), method 2 (Anova), and method 3 (wrapper).
The findings of feature selection approaches will be compared against the efficiency.

Table 1 lists the algorithms and feature selection techniques used in this study. Every algorithm will be evaluated against each feature selection approaches, generating a total of 27 of unique combinaisons detailed in table 1.

**Table 1** Feature Selection approachs and Algorithms studied

| | |
|---|---|
| Algorithms | RF<br>KNN<br>DT<br>NB<br>LR<br>MLP<br>ANN<br>CNN<br>LSTM |
| Method 1 | Study the effect of every feature on the labeled class |
| Method 2 | f-test (ANOVA) |
| Method 3 | Wrapper (RFE) |

Since VeReMi dataset comprises twenty nine features ('type', 'sendTime', 'sender', 'senderPseudo', 'messageID','posx', 'posy', 'posz', 'posx_n', 'posy_n', 'posz_n', 'spdx', 'spdy', 'spdz', 'spdx_n', 'spdy_n', 'spdz_n', 'aclx', 'acly','aclz', 'aclx_n', 'acly_n', 'aclz_n', 'hedx', 'hedy', 'hedz', 'hedx_n', 'hedy_n', 'hedz_n'), we apply the three distinct methodologies of feature selection to eliminate unnecessary features and delete redundant records from the database [47] that can negatively affect the effectiveness of the model in the detection of malicious activity.
The three approaches for selecting feature subsets are described below:

- **Method 1** In this model we study the effect of every feature on the labeled class attribute to distinguish if a vehicle is normal or misbehavior. Fig. 7 and Fig. 8 represent the impact of location (posx', 'posy') and sender time ('sendTime') features on the labeled class attribute, then we analyze the relevance of each feature to eliminate the irrelevant attribute.

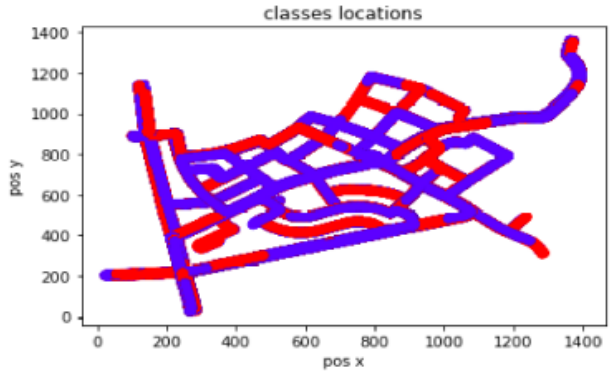


**Figure 7:** The Position effect on class attributes

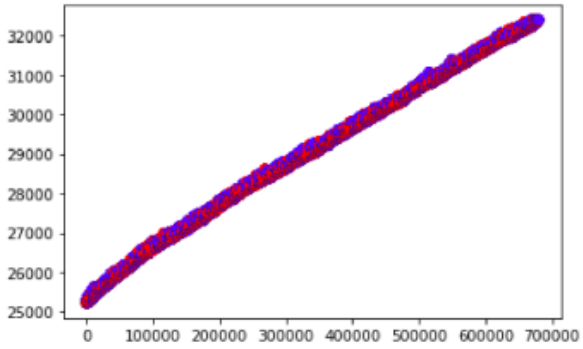


**Figure 8:** Sender Time effect on class attributes

We notice from these figures that the posx' and 'posy' attributes are relevant to differentiate between the normal and malicious vehicle, on the contrary 'sendTime' is an irrelevant attribute.
We may pick only twenty-four features from the VeReMi extension database using this feature selection approach ('posx', 'posy', 'posz', 'posx_n', 'posy_n', 'posz_n', 'spdx', 'spdy', 'spdz', 'spdx_n', 'spdy_n', 'spdz_n', 'aclx', 'acly','aclz', 'aclx_n', 'acly_n', 'aclz_n', 'hedx', 'hedy', 'hedz', 'hedx_n', 'hedy_n', 'hedz_n')}.

- **Method 2** In this method, we have employed the f-test in one-way Analysis of Variance (ANOVA), which is a type of F-statistic that donates an f-test by computing the ratio of the mean square of a gifted feature and the mean squared error [48]. Greater F-test values show that a feature is more profitable in distinguishing between groups. In this method, the features are ranked based on greater values of the f-test.
  Only twelve features of the VeReMi extension database may be selected using this approach ('posx', 'posy', 'posx_n', 'posy_n', 'spdx', 'spdy', 'spdx_n', 'aclx_n', 'hedx', 'hedy', 'hedx_n', 'hedy_n').

- **Method 3** In this method, we used Recursive Feature Elimination (RFE) [48] shown in Fig. 9, which is a Wrapper method. The procedure RFE is to attribute weights to features by an estimator in order to choose features through recursively envisaging less and less ensembles of features: the initial ensemble of features was trained by the estimator and with any determined attribute or callable, we get the relevance of every feature.

Then, the fewest important attributes are pruned to an existing ensemble of attributes. This procedure is recursively refined on the pruned ensemble pending the wanted number of features to choose is finally attained.
Only twelve features of the VeReMi extension database may be chosen using this approach (‘posx’, ‘posy’, ‘spdx’, ‘spdy’, ‘spdx_n’, ‘spdy_n’, ‘aclx’, ‘acly’, ‘hedx’, ‘hedy’, ‘hedx_n’, ‘hedy_n’).

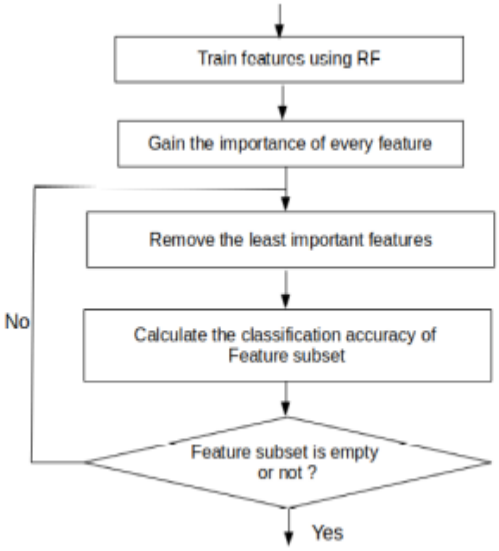


Figure 9: Recursive Feature Elimination algorithm

## 3.3 Classification

At this stage, Machine Learning classifiers are employed to determine if a node is normal or a malicious then determine the specific kind of misbehaving. With the presented features selection methods. In our research, we employed nine classification approaches for overall misbehavior detection: KNN, Random Forest, Decision Tree, Naive Bayes, MLP, ANN, CNN, and LSTM. These classifiers use a training set to train the model to distinguish the data as genuine or malicious.

- **A Binary Classification Approach (BCA)** In this technique, all kinds of misbehaviors are deemed to be part of a single "misbehavior," and the normal vehicle was labeled as 0, while the attacks and faults were grouped into one class (attacks+faults) and labeled as 1.

- **A Multi-class classification approachs** In this section, we will study three types of multi-class classification.

  *A Multi-class Classification Approach for Three Classes (MCATC):* In this approach, we distinguished between attacks and faults by separating the data into three classes: normal data was labeled as 0, attacker vehicle was labeled as 1, and fault behavior was labeled as 2.

  *A Classic Learning Approach for Multi-class classification (C-LAMC)*: allows you to identify and classify specific kinds of misbehavior into different misbehaving classes. In this approach, we use the twenty classes defined in the Extension VeReMi dataset numbered from 0 to 19.

  *A Guided Learning Approach for Multi-class Classification(G-LAMC):* Multi-class imbalance issues are far more difficult to solve because there are so many classes to consider and the interactions between them are so complex. To solve the problem of unbalanced data learning, we developed a new approach named "Guided Learning Approach for Multi-class Classification" (G-LAMC).

VeReMi extension dataset comprises nineteen types of misbehavior. We grouped similar attacks, which make similar malicious tasks, into one category in this new approach. So, we obtained nine classes presented below lastly we applied our classifiers to these primary categories:

**Normal (class 1):** This class represents the genuine vehicle.
**DosDistruptive attack (class 2):** It is a type of attack in which previously received data is replayed from random neighbors. It might also be a flood-the-network method to prevent legitimate communications from being broadcast, with a vehicle transmitting messages at a frequency greater than the limit defined by the appropriate IEEE or ETSI standards.
**Delayed Messages fault (Class 3) :** A high network overhead or a low-cost or slower on-board processing unit might be the cause. These communications contain all of the necessary data and information, but they are transmitted with a time delay from reality. Alternatively, the node communicates its data with a delay in comparison to reality.

**Random Position/speed offset fault (Class 4):** The node transmits its true position or speed with a random offset restricted to a maximum value.
**Eventual stop attack (Class 5):** By setting the speed to zero and establishing the location, the attacker simulated an abrupt stop.
**Constant position/speed fault (class 6):** It is a class when the attacker broadcasts a constant position or speed with each beacon.
**Constant offset Position/speed fault (Class 7):** In this class, the attacker broadcasts its true position or speed with a predefined offset.
**Random position/speed fault (Class 8):** The malicious vehicle broadcasts a random position from the vehicular area without taking into account its actual location. The attacker, on the other hand, transmits a random speed with an upper limit.
**Data Replay attack (class 9):** This includes forwarding already messages from a particular target neighbor. The information that is replayed is signed using the attacker's certificate. It might be carried out in Sybil mode, with the attacker changing identities for each new chosen target to prevent detection.

# 4 Results and Discussion

In this part, we present the performance metrics and discuss the results of the experiments.

## 4.1 Performance Metrics

It's critical to test the efficiency of ML and DL classifiers for classification in Misbehavior Detection System. The evaluation metrics [50] were documented in a confusion matrix that included information on the Predicted and Actual classifications where positive represents misbehavior, whereas negative implies a genuine vehicle in our dataset.
**True Positive (TP)** The algorithm rightly predicts the malicious vehicle.
**False Negative (FN)** The classifier misclassified the misbehaving node as genuine.
**False Positive (FP)** The model misclassified the data instances wrongly as a malicious.
**True Negative (TN)** The classifiers predict correctly the genuine vehicle.

| | | *Predicted class* | |
|---|---|---|---|
| | | **Negative** | **Positive** |
| *True class* | **Negative** | True Negative (TN) | False Positive (FP) |
| | **Positive** | False Negative (FN) | True Positive (TP) |

**Figure 10:** Confusion matrix

Our VeReMi dataset is imbalanced, because the numbers of normal vehicles are more than misbehaving.
We utilise precision, recall, and F1-score to evaluate models in cyber security, since accuracy does not provide a correct detection for the imbalance dataset.
The confusion matrix in Fig. 10 may be used to determine the detection metrics Precision (P), Recall (R), and F1-score (F1).

**Precision** demonstrates the system's competency to distinguish between legitimate and misbehaving nodes; low precision indicates that the classifiers produce a lot of FP as shown in equation (4).

$$Precision = \frac{TP}{TP+FP} \tag{4}$$

**Recall** determines the classifier's ability to identify a malicious vehicle; a low recall indicates that misbehavior is more hard to discover as shown in equation (5).

$$Recall = \frac{TP}{TP+FN} \tag{5}$$

**F1-score** is the harmonic meaning between precision and recall. It establishes a trade-off between accuracy and recall, such that an elevated F1-score indicates elevated precision and recall values, implying that the model's efficiency will be better as shown in equation (6).

$$F1 - Score = \frac{2*(Precision*Recall)}{(Precision+Recall)} \tag{6}$$

## 4.2 Performance of proposed ML/DL algorithms

In this part, with the three proposed feature selection methods, we discuss four classification algorithms for overall misbehavior detection. In the suggested detection framework, we used nine classifiers K-Nearest Neighbors (KNN), Random Forest (RF), Naive Bayes (NB), Logistic Regression (LR), Decision Tree (DT), and Multi-Layer Perceptron (MLP) in Machine Learning,

Artificial Neural Network (ANN), Convolutional Neural Network (CNN), and Long Short Term Memory (LSTM) in Deep Learning. As demonstrated in the tables below, tables 2, 3, 4 and 5 offer tabular representations of these classifiers classification results utilizing precision, recall, and F1-score as assessment metrics. The bold values represent the best results and the italic values represent the second best results.

The algorithms were trained using 1330377 genuine and 905988 misbehaving instances using diverse attributes. The performance of binary class and multi-class categorization utilizing features of genuine and malicious vehicles is shown in the tables below. When compared to binary categorization, multi-class categorization seems to be more difficult.

- **Binary Classification Approach (BCA)** The binary classification approach is used to lump all kinds of misbehavior in an one "misbehavior" category. The results of nine ML/DL classifiers using three feature extraction approaches are presented in Table 2 and (Fig. 11 a, b, c).

**Table 2** Evaluation metrics for Binary Classification Approach (BCA)

| | Method 1 | | | Method 2 | | | Method 3 | | |
|---|---|---|---|---|---|---|---|---|---|
| Classifiers | Precision | Recall | F1-Score | Precision | Recall | F1-Score | Precision | Recall | F1-Score |
| Random Forest | **0.9415** | **0.9353** | **0.9381** | **0.9408** | **0.9347** | **0.9375** | **0.9431** | **0.9379** | **0.9403** |
| KNN | 0.8663 | 0.8489 | 0.8553 | 0.8532 | 0.8329 | 0.8399 | 0.8749 | 0.8582 | 0.8645 |
| Naive Bayes | 0.6907 | 0.6275 | 0.6210 | 0.6905 | 0.6266 | 0.6197 | 0.6830 | 0.6188 | 0.6100 |
| Logistic Regression | 0.7887 | 0.6579 | 0.6502 | 0.7001 | 0.6269 | 0.6184 | 0.7988 | 0.6602 | 0.6524 |
| Decision Tree | 0.9330 | 0.9311 | 0.9320 | *0.9281* | *0.9268* | *0.9274* | *0.9342* | *0.9323* | *0.9332* |
| MLP | 0.8653 | 0.7690 | 0.7815 | 0.8579 | 0.7690 | 0.7812 | 0.8685 | 0.7758 | 0.7888 |
| ANN | 0.8896 | 0.8568 | 0.8670 | 0.8801 | 0.8314 | 0.8438 | 0.8065 | 0.8065 | 0.8207 |
| CNN | 0.8767 | 0.7724 | 0.7853 | 0.8820 | 0.7990 | 0.8133 | 0.8549 | 0.7628 | 0.7745 |
| LSTM | *0.9391* | *0.9287* | *0.9332* | 0.9113 | 0.8866 | 0.8953 | 0.9002 | 0.8669 | 0.8774 |

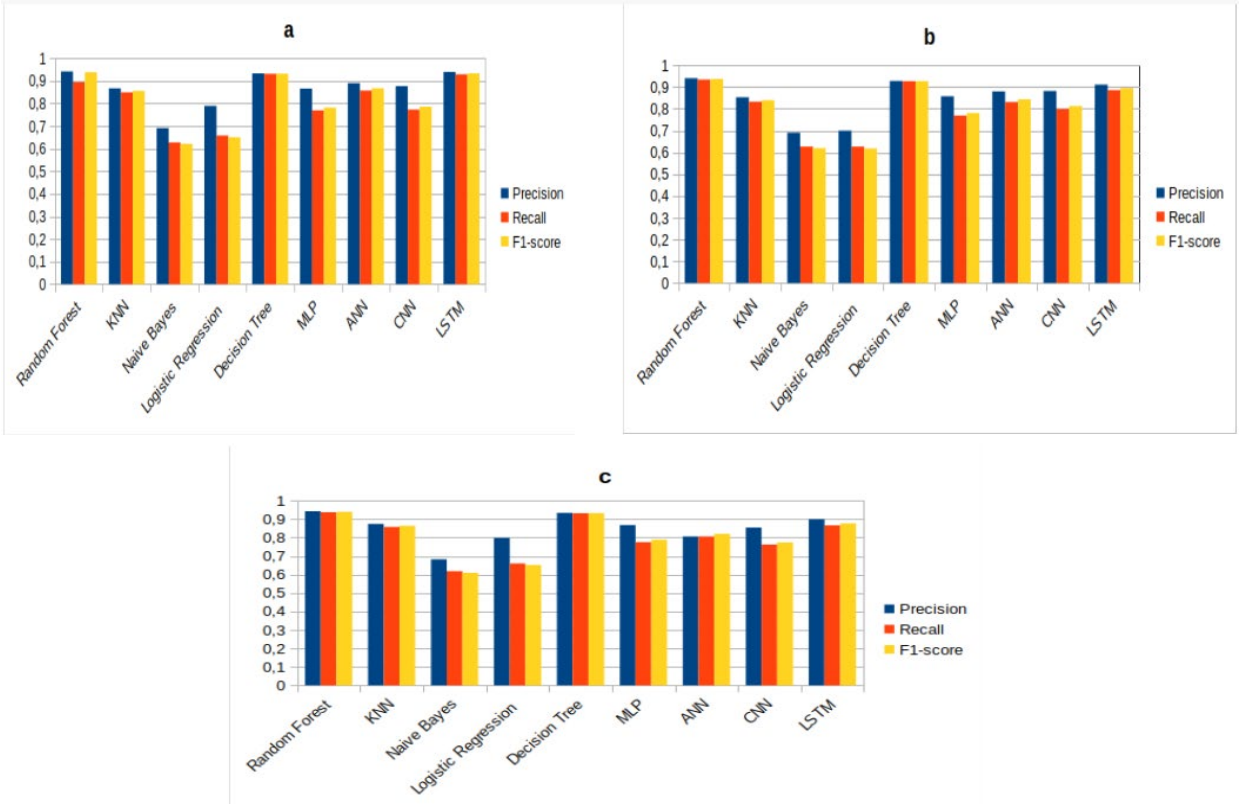


**Figure 11:** Binary classification approach using (a) Method 1 (b) Method 2 (c ) Method 3

BCA demonstrates:

In method 1, RF achieves excellent results with a precision of 0.9415, a recall of 0.9353, and an F1-score of 0.9381. LSTM as Deep Learning comes in second place with a precision of 0.9391, a recall of 0.9287, and an F1-score of 0.9332, while DT

comes in third place. The Naive Bayes classifier has the weakest metrics, with a precision of 0.6907, a recall of 0.6275 and an F1-score of 0.6210.

In method 2, the results of RF produce the best results, with a precision of 0.9408, a recall of 0.9347, and an F1-score of 0.9375. The second-best algorithm was DT, which had a precision of 0.928, a recall of 0.9268, and an F1-score of 0.9274. Naive Bayes is the weakest classifier, with a precision of 0.6905, a recall of 0.6266 and an F1-score of 0.6197.

Another method of testing based on the RFE approach (method 3) was used. When compared to other commonly used machine learning techniques, RF produces significant results with a precision of 0.9379, a recall of 0.9379, and an F1-score of 0.9403.

The second-best approach was DT, with a precision of 0.9342, a recall of 0.9323, and an F1-score of 0.9332 in DT. When compared to other commonly used ML approaches, Naive Bayes produces the poorest results with 0.6830, 0.6188, and 0.61 in precision, recall, and F1-score, respectively.

Lastly, we conclude, according to table 2 that Method 3 (RFE) outperforms Method 1 and Method 2, and that Random Forest classifiers outperform all other ML approaches to discriminate between genuine and misbehaving vehicles.

- **Multi-class classification approaches**

*Multi-class Classification Approach for Three Classes (MCATC)*

In MCATC, we distinguished between attacks and faults by separating the data into three classes. Table 3 and (Fig. 12 a, b, c) shows the findings of ML/DL classifiers employing various feature selection methods.

**Table 3** Evaluation metrics of Multi-class Classification Approach for Three Classes (MCATC)

| | Method 1 | | | Method 2 | | | Method 3 | | |
|---|---|---|---|---|---|---|---|---|---|
| Classifiers | Precision | Recall | F1-Score | Precision | Recall | F1-Score | Precision | Recall | F1-Score |
| Random Forest | *0.9399* | *0.9144* | *0.9265* | **0.9385** | **0.919** | **0.9284** | **0.9408** | **0.9224** | **0.9312** |
| KNN | 0.8788 | 0.8125 | 0.84 | 0.8678 | 0.7991 | 0.8276 | 0.8868 | 0.8315 | 0.8554 |
| Naive Bayes | 0.6734 | 0.6155 | 0.6117 | 0.6654 | 0.6061 | 0.5997 | 0.6733 | 0.6154 | 0.6019 |
| Logistic Regression | 0.7387 | 0.6161 | 0.6403 | 0.7121 | 0.6287 | 0.6124 | 0.7716 | 0.6432 | 0.6469 |
| Decision Tree | 0.9269 | 0.9140 | 0.9202 | *0.9232* | *0.9102* | *0.9165* | *0.9299* | *0.9164* | *0.9229* |
| MLP | 0.8708 | 0.6896 | 0.7347 | 0.8520 | 0.6768 | 0.7220 | 0.8756 | 0.6964 | 0.7413 |
| ANN | 0.8962 | 0.8625 | 0.8757 | 0.8901 | 0.8340 | 0.8511 | 0.8850 | 0.7841 | 0.8062 |
| CNN | 0.9225 | 0.8998 | 0.9092 | 0.9027 | 0.8657 | 0.8782 | 0.8969 | 0.8397 | 0.8559 |
| LSTM | **0.9312** | **0.9246** | **0.9273** | 0.9239 | 0.9057 | 0.9130 | 0.8914 | 0.8064 | 0.8210 |

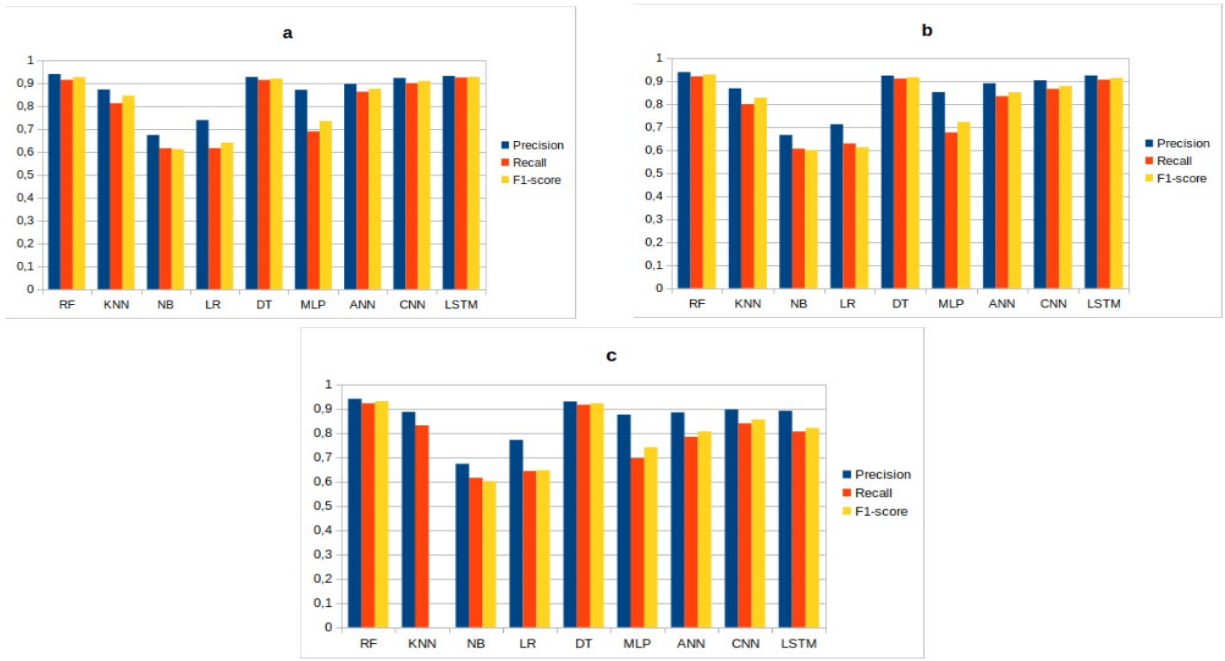


**Figure 12:** MCATC using (a) Method 1 (b) Method 2 (c) Method 3

MCATC demonstrates:

In method 1, LSTM classifiers perform well, with a precision of 0.9312, a recall of 0.9246, and an F1-score of 0.9273. In second place, we find that RF and CNN produce a good result with a precision of 0.9399, a recall of 0.9144 and an F1-score of 0.9265 in RF. However, Naive Bayes had a poor result with a precision of 0.6734, a recall of 0.6155, and an F1-score of 0.6117. We note that Method 1 works well in DL classifiers to classify faults and attacks.

In method 2, the results of RF produce the best results, with a precision of 0.9385, a recall of 0.919 , and an F1-score of 0.9284. The second-best algorithm was DT, which had a precision of 0.9232, a recall of 0.9102, and an F1-score of 0.9165. Naive Bayes is the weakest classifier, with a precision of 0.6654, a recall of 0.6061 and an F1-score of 0.5997.

In method 3, RF produces the best results with a precision of 0.9408, a recall of 0.9224, and an F1-score of 0.9312. In second place we find Decision Tree. The weakest results are in Naive Bayes with 0.6733, 0.6154, and 0.6019 in precision, recall, and F1-score respectively.

Among the two approaches, the BCA classifier appears to produce better results in machine learning methods when the classifiers can categorize attacks and faults in the same class;

The MCATC classifier appears to perform better in deep learning methods, where models can better categorize faults and attacks separately. As a result, this classification appears to be a better strategy because the performance is greater and we can categorize attacks and faults individually. This technique can help applications that require the classification of faults and attacks.

*Classic Learning Approach for Multi-class classification (C-LAMC)*

In this section, we will classify misbehaviors into particular misbehaving classes employing the C-LAMC shown in table 4 and (Fig. 13 a, b, c).

**Table 4** Evaluation metrics of Classic Learning Approach for Multi-class classification (C-LAMC)

| | Method 1 | | | Method 2 | | | Method 3 | | |
|---|---|---|---|---|---|---|---|---|---|
| Classifier | Precision | Recall | F1-Score | Precision | Recall | F1-Score | Precision | Recall | F1-Score |
| Random Forest | **0.699** | **0.675** | **0.682** | **0.663** | **0.643** | **0.648** | **0.71** | **0.691** | **0.696** |
| KNN | 0.609 | 0.569 | 0.58 | 0.535 | 0.499 | 0.507 | 0.608 | 0.572 | 0.582 |
| Naive Bayes | 0.122 | 0.054 | 0.053 | 0.101 | 0.103 | 0.108 | 0.101 | 0.102 | 0.101 |
| Logistic Regression | 0.103 | 0.024 | 0.026 | 0.103 | 0.01 | 0.01 | 0.077 | 0.002 | 0.02 |
| Decision Tree | 0.603 | 0.660 | 0.666 | 0.718 | 0.667 | 0.607 | 0.733 | 0.671 | 0.614 |
| MLP | 0.463 | 0.203 | 0.255 | 0.442 | 0.165 | 0.244 | 0.433 | 0.123 | 0.244 |
| ANN | 0.442 | 0.438 | 0.409 | 0.416 | 0.396 | 0.366 | 0.437 | 0.43 | 0.403 |
| CNN | 0.645 | 0.634 | 0.665 | 0.633 | 0.434 | 0.523 | 0.633 | 0.523 | 0.544 |
| LSTM | 0.285 | 0.306 | 0.26 | 0.302 | 0.307 | 0.265 | 0.297 | 0.312 | 0.278 |

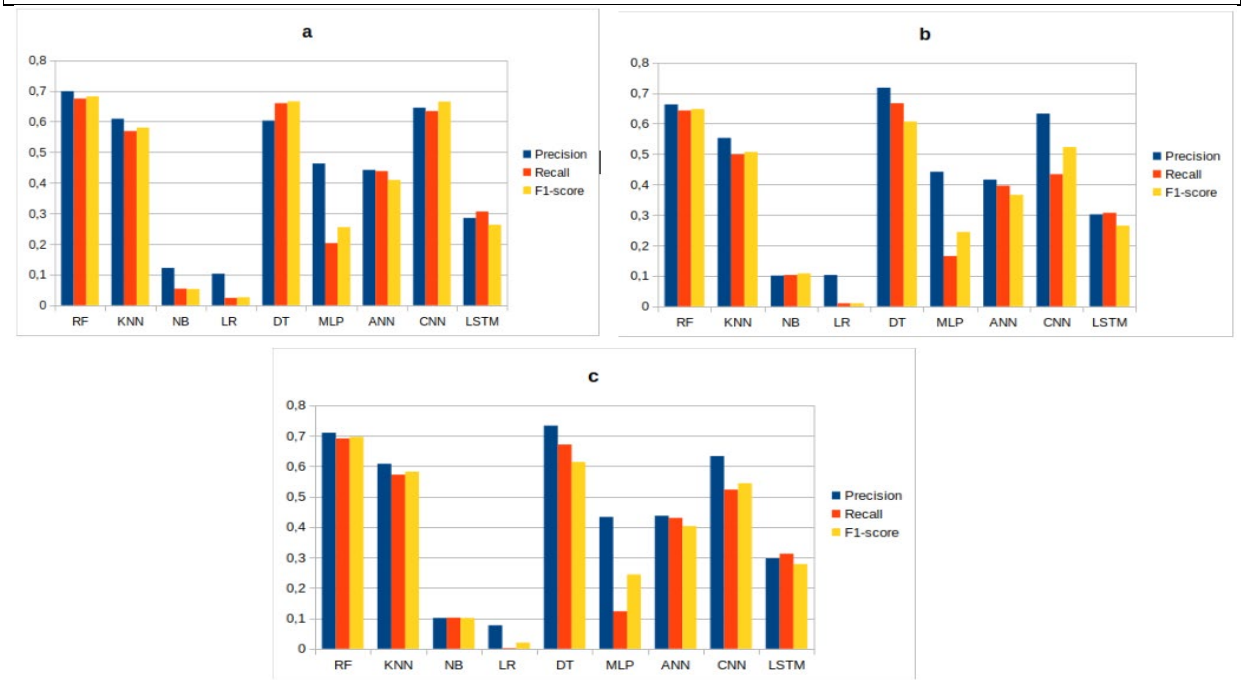


**Figure 13:** CLAMC using (a) Method 1 (b) Method 2 (c ) Method 3

We notice that method 3 gives good results than method 1 and method 2 and RF outperforms the others classifiers with a precision of 0.71, a recall of 0.691 and F1-score of 0.696 but these results are unsatisfactory due to learning from imbalanced data. So, to avoid this problem we attempted to implement a novel Guided Learning Approach for Multi-class classification (G-LAMC) presented in next section.

*Guided Learning Approach for Multi-class Classification(G-LAMC)*

We'll present the findings of our new method in this part to categorizing misbehaviors into specific classes; Table 5, (Fig. 14 a, b, c) summarizes the Guided Learning Approach for Multi-class Classification (G-LAMC) results.

**Table 5** Evaluation metrics of Guided Learning Approach for Multi-class Classification (G-LAMC)

| | Method 1 | | | Method 2 | | | Method 3 | | |
|---|---|---|---|---|---|---|---|---|---|
| Classifiers | Precision | Recall | F1-Score | Precision | Recall | F1-Score | Precision | Recall | F1-Score |
| Random Forest | **0.909** | **0.893** | **0.900** | *0.899* | *0.887* | *0.896* | **0.908** | **0.894** | **0.901** |
| KNN | 0.838 | 0.685 | 0.746 | 0.816 | 0.627 | 0.695 | 0.839 | 0.699 | 0.734 |
| Naive Bayes | 0.239 | 0.151 | 0.155 | 0.206 | 0.150 | 0.158 | 0.221 | 0.152 | 0.142 |
| Logistic Regression | 0.209 | 0.054 | 0.056 | 0.173 | 0.03 | 0.03 | 0.106 | 0.006 | 0.021 |
| Decision Tree | *0.923* | *0.880* | *0.899* | **0.916** | **0.879** | **0.897** | 0.923 | 0.880 | 0.899 |
| MLP | 0.626 | 0.343 | 0.423 | 0.604 | 0.331 | 0.403 | 0.64 | 0.376 | 0.449 |
| ANN | 0.783 | 0.576 | 0.612 | 0.497 | 0.694 | 0.519 | 0.758 | 0.529 | 0.574 |
| CNN | 0.866 | 0.802 | 0.822 | 0.825 | 0.659 | 0.702 | 0.844 | 0.737 | 0.769 |
| LSTM | 0.819 | 0.639 | 0.690 | 0.799 | 0.619 | 0.661 | 0.806 | 0.622 | 0.686 |

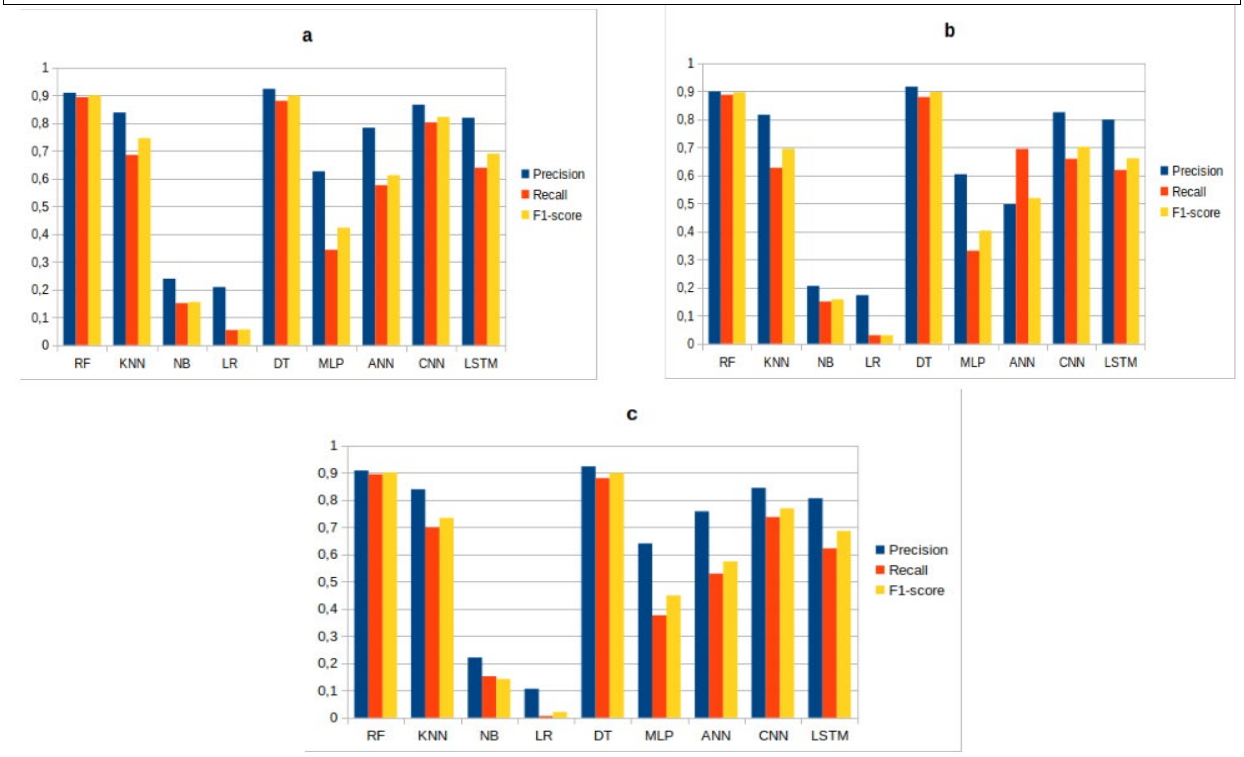


**Figure 14:** G-LAMC using (a) Method 1 (b) Method 2 (c ) Method 3

When compared to other ML approaches in *method 1,* we notice that RF achieves excellent results with an improvement of 22% when compared to the Classic Learning Approach for Multi-class classification (C-LAMC) where precision is 0.909, recall is 0.893, and the F1-score is 0.9 in the new *Guided Learning Approach for Multi-class Classification(G-LAMC)* .

Decision Tree comes in second place with an improvement of 21% in precision, recall and F1- score: 0.923, 0.880, and 0.899, respectively. When compared to all other ML approaches, the Logistic Regression and Naive Bayes classifiers have the weakest metrics, with a precision of 0.209, a recall of 0.054 and an F1-score of 0.056 in Logistic Regression.

We use the ANOVA (*method 2*) feature selection method for multi-class classification issues after evaluating all the specified ML algorithms in the entire dataset. The findings of DT in new approach *(G-LAMC)* are significant with an improvement of 19%, 21% and 27% in precision, recall and F1-score respectively when compared to classic approach (C-LAMC) where precision is 0.916, recall is 0.879, and F1-score is 0.879 in *(G-LAMC)*.

The second best algorithm was RF with an increase of 23%, 24% and 25% in precision, recall and F1-score respectively when compared to classic approach (C-LAMC), which had a precision of 0.899, a recall rate of 0.887 and F1-score of 0.896 in new approach *(G-LAMC)* . Logistic Regression and Naive Bayes stay the weakest classifiers despite the improvement in G-LAMC.

Another testing approach, based on the RFE (*method 3*) method, was used. When compared to other commonly used machine learning techniques, RF produces the best results with an increase of 20% when compared to classic approach (C-LAMC) . The precision is 0.908, the recall is 0.894, and the F1-score is 0.901 in new approach *(G-LAMC)*. The second best approach was DT with an improvement of 19%, 21% and 29% in precision, recall and F1-score respectively when compared to classic approach (C-LAMC)

Naive Bayes and Logistic Regression stay producing the poorest results.

Tables 5 shows that *Method 3* (RFE) outperforms *Models 1* and *Model 2*, and that Random Forest classifiers outperform all other ML approaches.

Based on the results presented in tables 4 and 5, we conclude that our new Guided Learning Approach for Multi-class Classification (GLAMC) produces better results than the Classic Learning Approach for Multi-class Classification (CLAMC) in terms of classifying misbehaviors into specific misbehaving classes.

## 5 Conclusion and Future Work

In this research, we suggested a ML and a DL based classification algorithms for detecting and categorizing malicious vehicles in VANETs. We've developed four classification approaches: binary, multi-class classification for three classes, Classic Learning Approach for Multiclass Classification, and a novel approach named "Guided Learning Approach for Multiclass Classification" to differentiate between (genuine and all different kinds of misbehavior), (genuine and misbehavior types are divided into attacks and faults), (genuine and a specific kind of misbehavior), and (genuine and grouping similar misbehaviors into one class) respectively.

Based on the results of the studies, we found that binary classification is better to multi-class classification for three classes in terms of classifying misbehaviors into specific misbehaving classes, we notice that our novel Guided Learning Approach for Multiclass Classification outperforms the Classic Learning Approach for Multiclass Classification, and ML classifiers, particularly Random Forest, are more efficient and promising than DL classifiers in classifying misbehavior.

Also we notice that the choice of features selection methods has an effect on the performance of the model, our experimental study shows that recursive feature elimination performs better. As part of our future work, we would like to build a new technique to improve the detection and multi-class classification of misbehaviors in VANET

## References

[1] N. Silva, J. Soares, V. Shah, M. Santos, and H. Rodrigues, “Anomaly detection in roads with a data mining approach,” *Procedia Computer Science*, vol. 121, pp. 415–422, 2017.

[2] O. national interministériel de la sécurité routière Onisr, “Bilan 2019 de la sécurité routière.” 2019.

[3] B. Alaya, R. Khan, and T. Moulahi, “Study on QoS Management for Video Streaming in Vehicular Ad Hoc Network (VANET),” *Wirel Pers Commun*, vol. 118, no. 2175–2207, 2021, doi: https://doi.org/10.1007/s11277-021-08118-7.

[4] W. Liang, Z. Li, H. Zhang, S. Wang, and R. Bie, “Vehicular ad hoc networks: architectures, research issues, methodologies, challenges, and trends,” *International Journal of Distributed Sensor Networks*, vol. 11, no. 8, Art. no. 8, 2015, doi: https://doi.org/10.1155/2015/745303.

[5] S. Al-sultan, M. Al-doori, A. Al-bayatti, and H. Zedan, “A comprehensive survey on vehicular ad hoc network,” *Journal of network and computer applications*, vol. 37, pp. 380–392, 2014, doi: http://dx.doi.org/10.1016/j.jnca.2013.02.036i.

[6] S. Sepasgozar and S. Pierre, “An Intelligent Network Traffic Prediction Model Considering Road Traffic Parameters Using Artificial Intelligence Methods in VANET,” *IEEE Access*, 2022, doi: 10.1109/ACCESS.2022.3144112.

[7] P. Tyagi and D. Dembla, “Investigating the security threats in vehicular ad hoc networks (VANETs): towards security engineering for safer on-road transportation,” *International conference on advances in computing, communications and informatics (ICACCI), IEEE*, pp. 2084–2090, 2014, doi: 978-1-4799-3080-7/14.

[8] B. Yelure and S. Sonavane, “SARP: secure routing protocol using anonymous authentication in vehicular Ad-hoc networks,” *Journal of Ambient Intelligence and Humanized Computing*, vol. 72, no. 2, Art. no. 2, 2021, doi: https://doi.org/10.1007/s12652-021-03486-1.

[9] Z. Sherali, H. Ray, and C. Yuh-Shyan, “Vehicular ad hoc networks (VANETS): status, results, and challenges,” *Telecommunication Systems*, vol. 50, no. 4, Art. no. 4, 2012, doi: 10.1007/s11235-010-9400-5.

[10] M. Jain and R. Saxena, “VANET: security attacks, solution and simulatione,” in *Proceedings of the Second International Conference on Computational Intelligence and Informatics*, Jul. 2018, vol. 712. doi: https://doi.org/10.1007/978-981-10-8228-3_42.

[11] R. Kolandaisamy, R. Noor, and I. Kolandaisamy, “A stream position performance analysis model based on DDoS attack detection for cluster-based routing in VANET,” *Journal of Ambient Intelligence and Humanized Computing*, vol. 12, no. 6, Art. no. 6, 2021, doi: https://doi.org/10.1007/s12652-020-02279-2.

[12] S. Iqbal, P. Ball, M. Kamarudin, and A. Bradley, "Simulating Malicious Attacks on VANETs for Connected and Autonomous Vehicle Cybersecurity: A Machine Learning Dataset," 2022.
[13] M. Zongo, J. Nlong II, R. Ndoundam, and A. Foerster, "DSRC Applicability in Cameroon Road Traffic: A Simulation Study," *EAI Endorsed Transactions on Mobile Communications and Applications*, vol. 6, no. 19, Art. no. 19, 2021, doi: 10.4108/eai.11-6-2021.170233.
[14] A. Gruebler, M. Donald, D. Klaus, Alheeti, and M. Khattab, "An intrusion detection system against black hole attacks on the communication network of self-driving cars," presented at the Sixth International Conference on Emerging Security Technologies, 2015. doi: 10.1109/EST.2015.10.
[15] A. Alsarhan, M. Alauthmen, and E. Alshdaifet, "Machine Learning-driven optimization for SVM-based intrusion detection system in vehicular ad hoc networks," *Journal of Ambient Intelligence and Humanized Computing*, pp. 1–10, 2021, doi: https://doi.org/10.1007/s12652-021-02963-x.
[16] N. Chaubey and D. Yadav, "Detection of Sybil attack in vehicular ad hoc networks by analyzing network performance," *International Journal of Electrical & Computer Engineering*, vol. 12, no. 2, Art. no. 2, 2022.
[17] J. Kim and S. Hong, "Study on the Privacy-Preserving Vehicular PKI in Autonomous Driving Environments," *Contemporary Engineering Sciences*, vol. 9, no. 13, Art. no. 13, 2016, doi: http://dx.doi.org/10.12988/ces.2016.6449.
[18] I. Ali, Y. CHen, M. Faisal, and L. Meng, *Efficient and Provably Secure Schemes for Vehicular Ad-Hoc Networks*. Springer, 2022.
[19] J. Kamel, M. Wolf, R. van der Hei, A. Kaiser, and P. Urien, "Veremi extension: A dataset for comparable evaluation of misbehavior detection in vanets." 2020.
[20] U. Khan, S. Agrawal, and S. Silakari, "Detection of Malicious Nodes (DMN) in Vehicular Ad-Hoc Networks," *Procedia computer science,* vol. 46, pp. 965–972, 2015, doi: 10.1016/j.procs.2015.01.006.
[21] J. Grover, N. Prajapati, V. Laxmi, and M. Gaur, "Machine Learning Approach for Multiple Misbehavior Detection in VANET," presented at the International conference on advances in computing and communications, 2011, vol. 192, pp. 644–653. doi: https://doi.org/10.1007/978-3-642-22720-2_68.
[22] J. Grover, V. Laxmi, and M. Gaur, "Misbehavior Detection based on Ensemble Learning in VANET," presented at the International Conference on Advanced Computing, Networking and Security, 2012, vol. 7135, pp. 602–611. doi: https://doi.org/10.1007/978-3-642-29280-4_70.
[23] S. Muthukumar and R. Karthick, "Identifying the Misbehavior Nodes Using Trust Management in VANETs," *International Journal of Advanced Research in Computer Science & technology (IJARCT 2014)*, vol. 2, no. 1, Art. no. 1, 2014.
[24] R. Barnwal and S. Ghosh, "Detection of Misbehaving Nodes in Vehicular Ad-hoc Network," *Security for Multihop Wireless Networks*, p. 125, 2014.
[25] H. Sedjelmaci and et al., "Predict and Prevent From Misbehaving Intruders in Heterogeneous Vehicular Networks," *Vehicular Communications, Elsevier*, vol. 10, pp. 74–83, 2017.
[26] Z. Soltani, "Misbehavior Node Detection in Vehicular ad-hoc Networks: A survey, With Special Emphasis on Multihop Broadcast Protocols," *Researcher*, vol. 9, no. 1, Art. no. 1, 2017.
[27] P. Tiwari and M. Gupta, "Security Enhancement of Misbehavior Nodes in Vehicular Ad-Hoc Networks Using Hash Function: A Survey," *International Journal of Engineering and Technical Research (IJETR)*, vol. 8, no. 6, Art. no. 6, 2018.
[28] G. Yan, S. Olariu, and M. Weigle, "Providing VANET Security Through Active Position Detection," *Comput. Commun.*, vol. 31, no. 12, Art. no. 12, 2008, doi: https://doi.org/10.1016/j.comcom.2008.01.009.
[29] S. Steven, P. Sharma, and J. Petit, "Integrating plausibility checks and machine learning for misbehavior detection in VANET," presented at the 17th IEEE International Conference on Machine Learning and Applications, 2018, pp. 564–571. doi: 11109/ICMLA.2018.00091.
[30] P. Singh, S. Gupta, R. Vashistha, and S. Nandi, "Machine learning based approach to detect position falsification attack in vanets," vol. 939, pp. 166–178, 2019, doi: https://doi.org/10.1007/978-981-13-7561-3_13.
[31] A. Waleed, M. FK, F. A, Muazzam M, S. A, and Y. N. et al, "Optimized node clustering in VANETs by using meta-heuristic algorithms," *Electronics*, vol. 9, no. 3, Art. no. 3, 2020.
[32] K. Ismo and et al., "Outlier detection using k-nearest neighbour graph." 2004.
[33] A. Buczak and E. Guven, "A survey of data mining and machine learning methods for cyber security intrusion detection," *IEEE Communications surveys & tutorials*, vol. 18, no. 2, Art. no. 2, 2015.
[34] S. Himani and K. Sunil, "A survey on decision tree algorithms of classification in data mining," *International Journal of Science and Research (IJSR)*, vol. 5, no. 4, Art. no. 4, 2016.
[35] S. Tausifa and M. Chishti, "Exploring the applications of machine learning in healthcare," *International Journal of Sensors Wireless Communications and Control*, vol. 10, no. 4, Art. no. 4, 2020, doi: https://doi.org/10.2174/2210327910666191220103417.
[36] M. Swarnkar and N. Hubballi, "OCPAD: One class Naive Bayes classifier for payload based anomaly detection," *Expert Systems with Applications*, vol. 64, no. 2016, Art. no. 2016, 2016, doi: https://doi.org/10.1016/j.eswa.2016.07.036.
[37] G. Box and G. Tiao, *Bayesian inference in statistical analysis*, vol. 40. John Wiley & Sons, 2011.
[38] N. Andrew and M. Jordan, "On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes," *Advances in neural information processing systems*, vol. 14, 2001.
[39] M. Madhiarasan and S. Deepa, "Comparative analysis on hidden neurons estimation in multi layer perceptron neural networks for wind speed forecasting," *Artificial Intelligence Review*, vol. 48, no. 4, Art. no. 4, 2017, doi: https://doi.org/10.1007/s10462-016-9506-6.
[40] J. Goldarag, Y. MohammadZadeh, and A. Ardakani, "A Fire risk assessment using neural network and logistic regression," *J. Indian Soc. Remote Sens*, vol. 44, pp. 885–89420, 2016, doi: https://doi.org/10.1007/s12524-016-0557-6.

[41] A. Borovykh, S. Bohte, and W. C. Oosterlee, "Conditional time series forecasting with convolutional neural networks," *arXiv preprint arXiv:1703.04691*, 2017.
[42] J. Asharf, N. Moustafa, H. Khurshid, E. Debie, W. Haider, and A. Wahab, "A review of intrusion detection systems using machine and deep learning in internet of things: challenges, solutions and future directions," *Electronics*, vol. 9, no. 7, Art. no. 7, 2020, doi: 10.3390/electronics9071177.
[43] R. Barzegar, M. Aalami, and J. Adamowski, "Short-term water quality variable prediction using a hybrid CNN–LSTM deep learning model," *Stochastic Environmental Research and Risk Assessment*, vol. 34, no. 2, Art. no. 2, 2020, doi: https://doi.org/10.1007/s00477-020-01776-2.
[44] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, Art. no. 8, 1997.
[45] J. Kamel, M. Wolf, R. van der Hei, A. Kaiser, p Urien, and P. Kargl, "VeReMi Extension." 2020. [Online]. Available: https://github.com/josephkamel/VeReMi-Dataset
[46] J. Tang, S. Alelyani, and H. Liu, "Feature selection for classifcation: A review," *CRC Press*, pp. 37–64, 2014.
[47] L. Ladha and T. Deepa, "Feature selection methods and algorithms," *International journal on computer science and engineering*, vol. 3, no. 5, Art. no. 5, 2011.
[48] J. Habbema and J. Hermans, "Selection of variables in discriminant analysis by F-statistic and error rate," *Technometrics*, vol. 19, no. 4, Art. no. 4, 1977, doi: https://doi.org/10.2307/1267890.
[49] X. Chen and J. Jeong, "Enhanced recursive feature elimination," in *Sixth International Conference on Machine Learning and Applications (ICMLA 2007)*, 2007, pp. 429–435. doi: 10.1109/ICMLA.2007.35.
[50] N. Bhandari, S. Khare, R. Walambe, and K. Kotecha, "Comparison of machine learning and deep learning techniques in promoter prediction across diverse species," *PeerJ Computer Science*, vol. 7, 2021, doi: http://doi.org/10.7717/peerj-cs.365.