Fall 2021

# Neural Network Based Approach for Detecting Location Spoofing in Vehicular Communication

Smarth Kukreja
*University of Windsor*

# Neural Network based Approach for Detecting Location Spoofing in Vehicular Communication

By

**Smarth Kukreja**

A Thesis
Submitted to the Faculty of Graduate Studies
Through the School of Computer Science
In Partial Fulfillment of the Requirements for
The Degree of Master of Science
At the University of Windsor

Windsor, Ontario, Canada

2021

# Neural Network based Approach for Detecting Location Spoofing in Vehicular Communication

by

Smarth Kukreja

APPROVED BY:

---

A. Sarker

Department of Mathematics and Statistics

---

K. Selvarajah

School of Computer Science

---

A. Jaekel, Advisor

School of Computer Science

November 11, 2021

# Declaration of Originality

I hereby certify that I am the sole author of this thesis and that no part of this thesis has been published or submitted for publication.

I certify that, to the best of my knowledge, my thesis does not infringe upon anyone's copyright nor violate any proprietary rights and that any ideas, techniques, quotations, or any other material from the work of other people included in my thesis, published or otherwise, are fully acknowledged in accordance with the standard referencing practices. Furthermore, to the extent that I have included copyrighted material that surpasses the bounds of fair dealing within the meaning of the Canada Copyright Act, I certify that I have obtained a written permission from the copyright owner(s) to include such material(s) in my thesis and have included copies of such copyright clearances to my appendix.

I declare that this is a true copy of my thesis, and that this thesis has not been submitted for a higher degree to any other University or Institution.

# Abstract

Vehicular Ad hoc Network (VANET) is an evolving subset of MANET. It's deployed on the roads, where vehicles act as mobile nodes. Active security and Intelligent Transportation System (ITS) are integral applications of VANET, which require stable and uninterrupted vehicle-to-vehicle communication technology. VANET, is a type of wireless network, due to which it is quite prone to security attacks. Extremely dynamic connections, sensitive data sharing and time-sensitivity of this network make it a vulnerable to security attacks. The messages shared between the vehicles are the basic safety message (BSM), these messages are broadcasted by each vehicle in the network to report its status to the other vehicles and Road Side Unit (RSU). One common attack is to use position falsification to hamper the roadside safety, leading to road accidents and congestion. Identifying malicious nodes involved in such attacks is crucial to ensure safety in the network. The proposed research presents a neural network based approach for detecting position falsification attacks in VANET.

The proposed Deep Learning-based detection of attackers is done using the dataset called Vehicular Reference Misbehavior Dataset (VeReMi). VeReMi dataset provides five classes of attackers, each broadcasting fabricated coordinates concerning the type. This MLP-based model uses resampled single BSM and two consecutive BSM to detect these attacks.

# Acknowledgements

First and foremost, I would like to thank the Almighty, for His blessings to complete my thesis work successfully. I would like to express my sincere gratitude to my supervisor, Dr.Arunita Jaekel, for her suggestions and assistance throughout my research. Also, I would like to thank my external reader Dr.Animesh Sarker and internal reader Dr.Kalyani Selvarajah for their motivating feedback and recommendations. I would also like to thank my family and friends for their love, care, and sacrifices. I would also like to thank my friends Aditya and Parminder for their continuous support and help throughout my thesis work.

# Table of Contents

# List of Figures

# Chapter 1

# Introduction

## 1.1 Introduction

Over the past couple of decades, communication techniques have transformed the automobile industry by providing instant communication among different devices. This effortless exchange of data on a real-time basis has become a new paradigm of the industry. The advancement in information technology and communication has made the idea of communication between mobile devices possible. Among these advancements, the concept of Ad hoc networks came into the limelight. Ad hoc networks are a combined set of interconnected devices that can communicate with one another. However, the feature that makes an ad hoc network unique is its property of decentralization. Rather than depending on devices such as routers or data points to generate a predefined structure for communication, each host present in the network acts as a router or access point itself and communicates directly with the other hosts. Ad hoc networks are highly advantageous when the network is highly mobile, with hosts coming and frequently going, such as in mobile ad hoc networks (MANET) [34].

A Vehicular Ad Hoc Network (VANET) [11] is a variation of MANET. VANET refers to a type of network created in an ad-hoc manner where each of the moving vehicles and other connecting devices present in the range communicates over a wireless medium and exchange helpful information to one another

VANETs function on the idea of vehicles communicating directly with one another. The type of communication thus being seen is Vehicle-to-Vehicle communication (V2V). However, specific extensions to the basic setup of the V2V structure, which indulge the need for road infrastructure to communicate to vehicles, called Vehicle to Infrastructure (V2I) communication, allow vehicles to communicate with road infrastructures such as overpasses or road-side signs. Dedicated Short-Range Communication is one of the prime technologies used in the VANET, especially in the V2V and V2I types of communications. In the United States of America, Federal Communication Commission has allocated a licensed spectrum of 75MHz in a 5.9GHz frequency bandwidth for *dedicated short-range communication* (DSRC) [33]. DSRC can be understood as "a two-way short- to medium-range wireless communications capability that allows quite high data communication, which is quite critical in communications-based active safety applications." Essentially, DSRC is a fast Wi-Fi with little overhead to allow fast enough communication for VANET usage. 802.11p, a wireless protocol standardized for wireless access in vehicular environments (WAVE), works per DSRC. On a vehicle, we call this an On-Board Unit (OBU), and on infrastructure, we call this a Road-Side Unit (RSU).

## 1.2 Motivation

Why there is a need for vehicles to communicate with one another? The reason can be seen for this, many of which involve safety or accident prevention. According to the "Vehicle Safety Communications (VSC) consortium identified eight severe applications: traffic signal violation warning, curve speed warning, emergency electronic brake light, pre-crash sensing, cooperative forward collision warning, left turn assistant, lane- change warning, and stop sign movement assistant." However, not being able to receive the genuine *Basic Safety Messages* (BSMs) can potentially lead to accidents and loss of life. The Basic Safety message can be understood as its name suggests the information transferred from one vehicle to another vehicle in the VANET network. The BSM consists of the safety messages that include the vehicle position, speed acceleration, heading and other relevant status information. The BSM messages are further explained in the section. Security and privacy are the two vital elements of any network. These are the primary needs of network communication, as

compromising these can lead to severe outcomes. In a wireless network, breach of security and privacy is common, and it gives the attacker an advantage to compromise the system. In the field of VANET especially, there is a dire need for security and privacy as breaching those on the road network can lead to significant consequences.

## 1.3 Problem Statement

In an ideal world, every BSM sent would be from a simple transmitter and received by the genuine receiver. Unfortunately, in VANETs, due to the communication being wireless, many attackers want to compromise the security of the communication. The resultant alteration of the communication can majorly lead to severe congestion on the road and also accidents. In the field of VANET, there are many types of attacks such as Dos, replay, message alteration that can lead to severe accidents. Some of the attacks thus want to compromise Confidentiality, Integrity and Availability. Among these attacks, there is also one more attack that targets the Integrity of the vehicles. This is the position falsification attack. In this particular type of threat, the attacker tries to send the wrong position [14] of the vehicle and thus can mislead the honest vehicle. This is one of the most commonly occurring types of attack seen in the VANET. The main focus of the research is to correctly identify in the VANET network when a BSM is being sent to the receiver, whether that BSM is from an actual vehicle or the attacker vehicle. In this thesis, a "genuine" vehicle [9] is one that sends valid BSMs and does not alter its contents in any way. Conversely, an "attacker" vehicle is engaged in a location spoofing attack by inserting false position coordinates in the BSM [17].

## 1.4 Solution Outline

Location spoofing attacks in BSMs compromise the safety of vehicles and passengers. So, a system needs to determine if a receiving BSM contains accurate information or is sent from an attacker's vehicle with false information. In this thesis, we are proposing a deep learning model-based approach to detect position falsification attacks. The main idea is to automatically recognize which messages are from genuine vehicles and attacker vehicles. The

proposed solution consists of two major stages: the primary stage is the dataset preparation, followed by the secondary stage of classification based on the knowledge gathered by the BSMs. The data extracted from the one ground truth file and multiple vehicle log files are pre-processed, and single and two consecutive BSM datasets are generated. This dataset trains the machine learning and deep learning algorithms to classify the vehicles as genuine or attackers.

## 1.5 Thesis Organization

After this chapter, the remaining portions of this thesis will be organized in the following manner. Chapter 2 will provide a brief sketch of VANET communication and a literature review of previous work done in detecting position falsification attacks. Chapter 3 will discuss the proposed Neural Network based approach in detail and how it differs from the existing approaches. Chapter 4 will discuss the simulation parameters, dataset, and the proposed approach's performance, including a comparison with existing schemes. Finally, chapter 5 will discuss the conclusions of the work completed and directions for future work.

# Chapter 2

# Background Review

## 2.1 Overview of VANET

Vehicular Adhoc Networks (VANET) [1] is a subset of wireless mobile ad hoc networks. It is an ad-hoc network where each node acts as an independent and self-organized entity. In VANET, various communication protocols have been proposed and used. The VANET routing protocol can be identified into two categories: topology-based routing protocols and Position-based routing protocols. The topology-based routing protocols use links knowledge to transmit the packets of data between the nodes using the VANET. The primary two subcategories under this type of mechanism are the proactive approach, which primarily depends on the type of routing techniques related to table-driven methodology and the reactive approach, which majorly depends on the type of routing techniques that are related to on-demand methodology. Position-based routing protocols use algorithms related to the positioning mechanism using location-based applications (For example, GPS) With the increase in the number of vehicles being equipped with smart technologies and wireless interact devices, inter-vehicle communication is becoming a primary field of research, regularity, and advancement. VANETs promises a wide variety of applications, such as restriction of blind crossing collisions, dynamic route scheduling, safety, real-time traffic condition monitoring, etc. The most straightforward route choice and assignment method is All-or-Nothing assignment (AON). This particular method takes that there are no present congestion effects that are being considered by the drivers for the same attributes for route

choice and perceive the same way. AON for VANETs is providing Internet connectivity to vehicular nodes.

## 2.2 VANET Communication

Each vehicle in the network is equipped with an On-board Unit (OBU), with the necessary computation and communication resources to facilitate communication with the nearby vehicles and other nodes. In addition to vehicles, the VANET environment includes Central Authorities (CAs), Roadside units (RSUs), and other devices like smartphones. The roadside units allow vehicles to disseminate messages present in their range and act as a point of access in the road network.

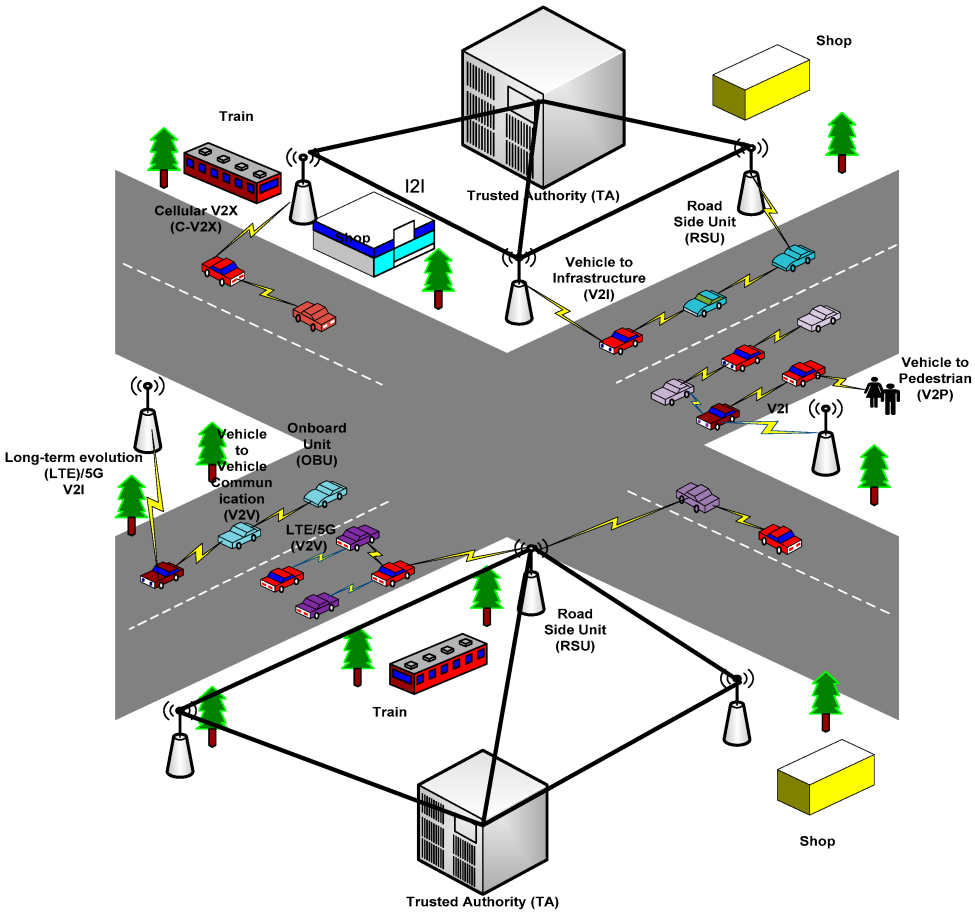


FIGURE 2.1: Types of Communication in VANET [28]

There are four different types of communication in VANET as shown in Fig 2.1

1) **Vehicle-to-Vehicle (V2V)**: In this type of communication, the information is being shared using the OBU's present on the vehicles, which helps the vehicles to communicate with one another. [4]

2) **Vehicle-to-Infrastructure (V2I)**: In this type of communication, the OBUs present on the vehicles interact with the RSUs available as the Road-Side and update the information of the vehicle. [4]

3) **Infrastructure-to-Infrastructure (I2I)**: Infrastructures also communicate with each other to provide backend support to the network

4) **Vehicle-to-Everything (V2X)**: In this, the vehicles communicate with any interconnected device such as mobile phones.

### 2.2.1 Security Requirements

VANET offers facilities and services over the wireless channel and can be quite prone to attacks [18]. To ensure reliable communication, specific security requirements [24] must be met, as discussed below.

1) **Authentication**: Authentication is the process of verifying whether someone (or something) is who (or what) it is declared to be. Authentication gives access control available for systems by ensuring that a user's credentials match the credentials present in a database of authorized users. Authentication is nothing but the process to verify that the information sent in the form of messages is legitimate. To have a safe network, the sender and receiver should be a part of the network. To maintain the legitimacy of the network, information sent and received must be authenticated. Types of authentication attacks are Message Tampering, Replay attack and Sybil attack. A message tampering attack is the type of attack which alters the information present in the message. In replay attack the attacker sends the same message but with different time stamps. In the Sybil attack, the attacker generates many ghost vehicles in the network and mislead the legitimate vehicle.

2) **Integrity**: This is designed to protect data from deletion or modification by any unauthorized party. The information (Knowledge or data) transmitted to the receiver from the transmitter must not be altered or manipulated before reaching the receiver. The message exchange between the sender and the receiver must not be tampered with by the attacker.

The main motive of this type of attack is that the attacker tries to send malicious information to corrupt the network. The types of integrity attacks are Timing attacks, Message alteration, Position Falsification Attacks and Message detection. The timing attack is the type of attack in which the attacker tries to delay the legitimate message. The message alteration and detection type of attack are when the attacker either inserts the false information or erases the simple message. The information altered can be the speed of the vehicle, break status, and where's the vehicle headed.

3)**Availability**: It can be defined as the availability of your data. Authentication helps channels and systems to work correctly for the data they conceal and make sure that it is ready to use when it is required.

High availability (HA) systems are computing resources with pretty large infrastructures designed to enhance availability. The type of HA system is what targets hardware failures or power outages to enhance availability, or it may manage several types of network connections to route around various network outages. A good network can be defined as one which is always available and provides service without any interruption. Any attack on the availability of the network prevents the genuine user from accessing the network. The types of attacks seen in the availability are DOS, DDOS, Malware and spamming. Denial of Service (DOS) is the type of attack in which the attacker tries to make the network unavailable for genuine users. Distributed Denial of Service (DDOS) is the type of attack i which a malicious attempt is made to disrupt the normal traffic of a targeted server, service or network by overwhelming the target with a flood of Internet traffic. In the Malware attack, the attacker tries to insert the malware in the message to compromise the availability of the network. In the spamming attack, the attacker tries to send spam messages in the form of original messages, which make the network unavailable for legitimate users.

4) **Confidentiality**: Confidentiality refers to an organization's efforts to keep its data private or secret. In actual life practice, the foremost thing is about allowing access to data and preventing unauthorized access. This mainly involves making sure that only those authorized by the specific domain have access to desired assets and that those who are not authorized are actively prohibited from gaining access. In this type of attack, the attacker

tries to compromise the confidentiality of the message by listening to the legitimate message. The type of attack seen in the confidentiality is Eavesdropping. In the eavesdropping attack, the attacker tries to listen to the message sent by the genuine user to the receiver user.

Some of the common security attacks in VANET and the security requirements that they violate are shown in Fig. 2.2.

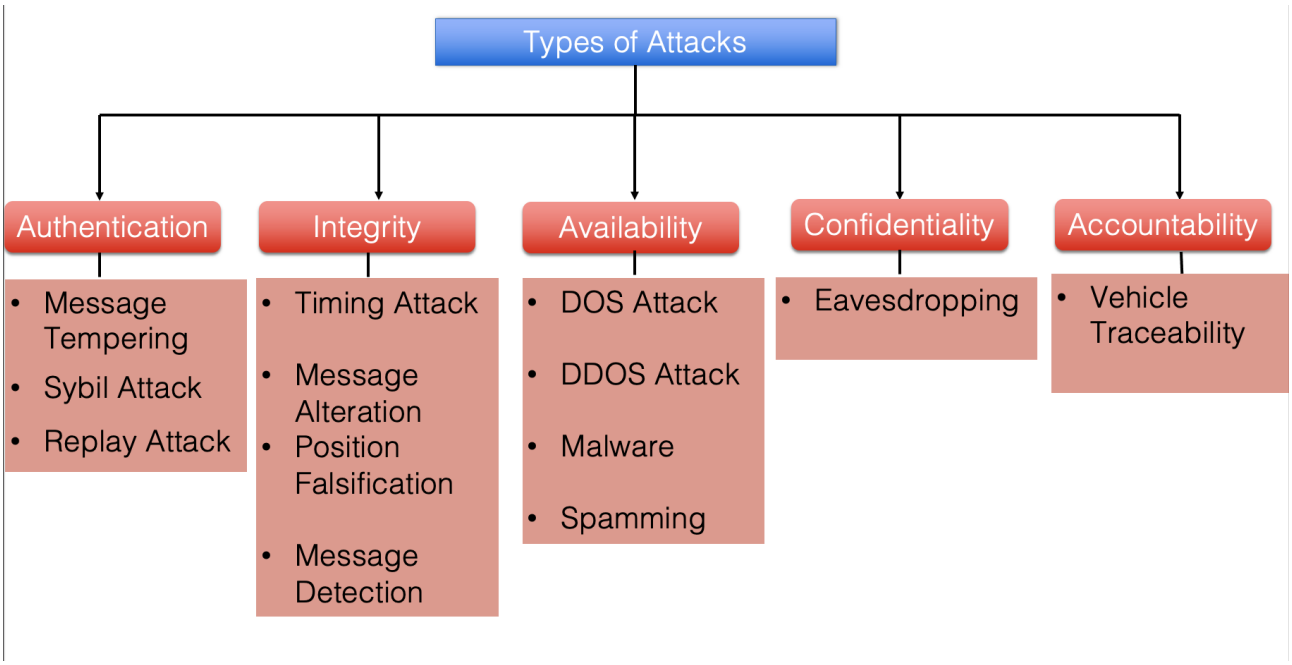


FIGURE 2.2: VANET attacks and threats

### 2.2.2 Types of Attackers

Wireless networks are prone to malicious attacks from attackers having different motives. An attacker can be understood as the entity whose primary aim is to alter the running network by affecting the infrastructure in any way possible. An attacker can create problems in the established network by gaining full access to the communication medium. Here are some types of attackers that are being seen in the VANET, who can potentially alter the network can have a hazardous effect on the road environment [19].

• **Insider**: This type of attacker can be considered as an authentic user of the network and have detail-specific knowledge of the network. Insider attacker has access to insider

knowledge, which is used to understand the design and configuration of the network. When they have all the required information about the system design, then it is easy for them to launch attacks and create a more severe problem as compared to outsider attackers. It can also create problems in the network by changing the encryption keys. In VANET, the insider attackers enter into the network and send some wrong messages. This can result in system jamming and accidents. This type of attack may be detected by using model-based consistency checking.

• **Outsider**: The outsider attacker is not considered an authentic user of the network. It is considered a kind of intruder whose primary aim is to misuse the established network protocols, and the range of such attacks is limited. Outsider attacker also has a limited diversity for launching different kinds of attacks as compared to insider attackers.

• **Vandal**: This type of attacker is part of the network, and the primary aim of this attacker is to showcase their abilities to attack the established network.

• **Malicious Hacker**: The malicious hacker is the type of attacker who is directly involved with the running system, and the primary aim of targeting the system is to have a personal gain or shared profit.

• **Rational Hacker**: The rational hacker is the type of attacker who majorly predicts the personal assistance from the invasion. Due to this, we can say that, unlike the malicious hacker, the rational hacker attacks are more certain and based on a set pattern.

### 2.2.3 Position Falsification Attack

VANET can be considered as one of the significant advancements in the field of autonomous vehicles. VANET supports a lot of types of applications in the field of autonomous vehicles. The two major types of applications supported in VANET are the safety application and the Non-safety applications. The non-safety applications are further divided into two types comfort applications and the traffic information system. Comfort applications are the applications that are related to the convenience of the user present in the network. The services available in the comfort application are mainly weather forecast updates or can be the shortest route to the desired destination. In distinction to the non-safety application,

the safety application mainly deals with the driver's safety present in the network. The safety application is further divided into two types: situation awareness and warning application. The situation awareness application deals with the driver's awareness, as seen in the blind spot warning to the driver or the adaptive cruise control. The warning message is event-driven and generates "alerts" for the driver.

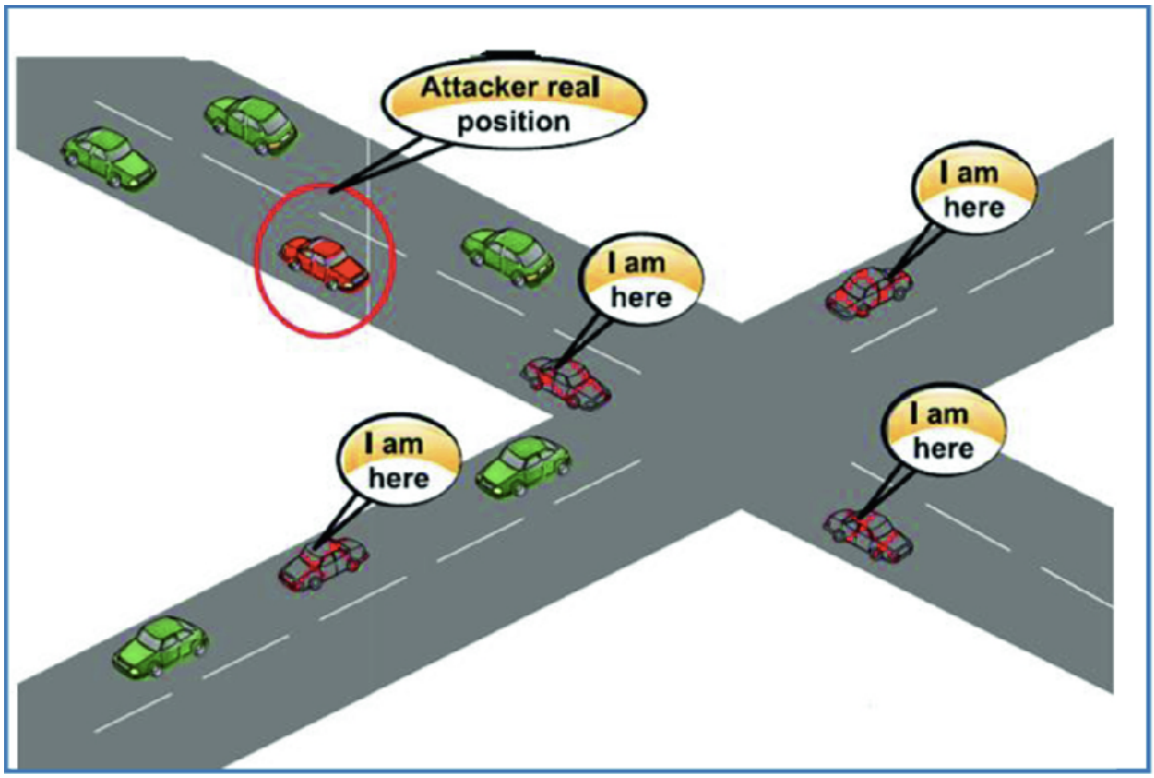


FIGURE 2.3: An example of Position Falsification attack [29]

Vehicles that are part of the network transmit their available status to the nearby present nodes in the road network. All the available nodes in the transmitter vehicle's range will receive a BSM (Basic Safety Message). As part of the thumb rule, these safety messages are being transmitted at a constant period in the network. BSMs, which are being sent to the receiver by the transmitter, are first digitally signed by the transmitter and contain the vehicle's information such as its current position, speed, direction and other information. Cryptographic techniques are used to sign the BSM contents digitally. Malicious vehicles send the false position of the vehicles in the form of BSM that mislead the legitimate vehicles in the network. An incorrect position can also be inserted if the GPS is not working correctly [38]. The attack generated by false position information present within the BSMs is known as the Position falsification attack, as depicted in Figure 2.3. A position falsification attack

violates the data integrity requirement for secure communication.

Position falsification attacks cannot be detected using cryptographic methods only, as they may originate from vehicles with valid credentials. So, additional mechanisms are needed to detect false data and ensure message correctness in the BSM.

## 2.3 Overview of Machine Learning and Deep Learning

### 2.3.1 Machine Learning

Machine learning is one of the prominent fields of Artificial Intelligence that facilitates machines to execute specific jobs faster and skillfully using statistical learning. These days, machine learning and deep learning, which is part of machine learning, is being rigorously used in several fields out of which the major industries that use them are e-commerce and spam detection email system. Machine learning algorithms discover patterns in input data to make predictions, detect or categorize data, and solve real-world problems. In the field of VANET, machine learning algorithms can detect attacks and misbehaviour in the network. There are four main types of machine learning [6]:

• **Supervised Learning**: In supervised machine learning, the algorithm is trained on the labelled data. Supervised learning helps to solve two types of problems: classification and regression.

• **Unsupervised Learning**: In unsupervised learning, the algorithm is trained on the unlabelled data and using that, a pattern is being detected. This type of learning is primarily used for organizing data present in the clusters and association learning.

• **Semi-supervised Learning**: From the start, the data is being mixed with both labelled and unlabelled data; it gives an advantage of both supervised and unsupervised learning.

• **Reinforcement learning**: It features an algorithm that improves upon itself and learns from new situations using a trial-and-error method. Favourable outputs are encouraged or 'reinforced,' and non-favourable outputs are discouraged or 'punished.' Based on the concept of conditioning, reinforcement learning works by using the algorithm in a work environment using an interpreter and a reward-based system.

### 2.3.2 Deep Learning

Deep learning is a subpart of machine learning that deals with algorithms inspired by the brain's structures called artificial neural networks. Deep learning mirrors the functioning of our brains. Deep learning algorithms are pretty similar to how the nervous system is structured, where each neuron is connected and passing information. Deep learning models work in multiple layers, and a general model has at least three layers. Each layer accepts some information from the previous and passes it on to the next one. Types of Deep learning are [5] :

1) **Feedforward neural network** This type of deep learning neural network is a simple neural network where the flow occurs from the base input layer and the secondary output layer. These kinds of networks are only having single layers or only one hidden layer. Since the data moves only in 1 direction, there is no backpropagation technique in this network. In this network, the sum of the weights present in the input is fed into the input layer. These kinds of networks are used in the facial recognition algorithm using computer vision.

2) **Multi-layer perceptron** This type of network has more than three layers used to classify the data, which is not linear form. These types of networks are fully connected with every node. These networks are extensively used for speech recognition and other machine learning technologies.

3) **Convolution neural network (CNN)** CNN is one of the variations of the multi-layer perceptron. CNN can contain more than one convolution layer, and since it contains a convolution layer, the network is very deep with fewer parameters. CNN is very effective for image recognition and identifying different image patterns.

4) **Recurrent neural network (RNN)** is a type of neural network where the output of a particular neuron is fed back as an input to the same node. This method helps the network to predict the output. This kind of network helps maintain a slight state of memory

which is very useful for developing the chatbot. This kind of network is used in chatbot development and text-to-speech technologies.

### 2.3.3 Basic Machine Learning and Deep Learning Terminologies

Basic terminologies and concepts of machine learning and deep learning used in the thesis are discussed below:

**Model**: A model is defined as the machine or deep learning algorithm used to solve the problem statement.

Dataset: Input data that is being used to train the model is known as a dataset.

**Feature Selection**: Features are nothing but the data objects in the dataset with essential characteristics to solve the problem.

**Data Pre-Processing**: Any raw dataset has much redundancy because of duplicates and noise present in the data. Due to this, it is pretty difficult to process the data and train the model leading to lower accuracy of results.

**Cross-Validation**: In this particular type of training and testing approach, the data is randomly split into groups. Of each group, there is train and test data, and the calculated average is the model performance. It is quite an effective technique to avoid overfitting.

**Activation functions**: These are the functions that majorly make the decision, given the inputs into the node because it is the activation function that decides the actual output; we often refer to the outputs of a layer as its "activations."

**Weights**: When an input data is passed to a neuron, it automatically gets multiplied by that value.

**Bias**: This is one of the learnable parameters of the neural network like weights.

### 2.3.4 Classification Algorithm

It is a subclass of supervised machine learning. Where we use labelled datasets as the input data, algorithms used to solve the classification problems are known as classifiers. In the VANET, the machine and deep learning can classify the legitimate and attacker nodes [22].

**Binary Classification**: Binary classification means there are two classes to work with that relate to one another as true and false.

**Multi-Class Classification**: Unlike binary classification, the multi-class classification does not have the notion of normal and abnormal outcomes. Instead, examples are classified as belonging to one among a range of available classes. The number of class labels may be vast on some problems.

**Binary Classification Algorithms**:

**Naive Bayes:** Naive Bayes is a machine learning model used for large volumes of data; if you are working with data with millions of data records, the recommended approach is Naive Bayes. It gives excellent results when it comes to Natural Language Processing tasks such as sentiment analysis. It is quite a fast classification algorithm. It is a type of classifier that works quite well with the Bayes theorem. Membership probabilities are predicted for every class, such as the probability of data points associated with a particular class. The class which has the maximum amount of probability is appreciated as the most suitable scenario. This is also referred to as Maximum A Posteriori (MAP).

**Logistic Regression:** Logistic regression is one of the most common machine learning algorithms used for binary classification. It majorly predicts the probability depending on the occurrence of a binary (0 and 1) outcome using a logit function. The logit function is the natural log of the odds that represents Y equals one of the primary categories. If we assume Y has only two categories and code them as 0 and 1, it is a particular case of linear regression as it predicts the outcome probabilities using log function. The activation function (sigmoid) can be used to convert the outcome into a categorical value.

**Support Vector Machines**: A support vector machine is a machine learning model that can generalize between two different classes if the set of labelled data is provided in the training set to the algorithm. The primary function of the SVM can check for the hyperplane, which then distinguishes between the two classes.

There can be many hyperplanes that can do this task, but the objective is to find that hyperplane that has the highest margin that means maximum distances between the two

classes, so that in future, if a new data point comes that is two be classified, then it can be classified easily.

**K-Nearest Neighbour**: KNN is being used for both classification and regression predictive problems. However, it is mainly used in classification problems in the industry. The K-Nearest Neighbour algorithm assumes that similar items are present within proximity. In simple words, similar things are within proximity to each other. KNN is implemented by finding the distance between all the data points and a query point and selecting k nearest neighbours. Based on the available labels of k nearest neighbours, it then chooses the label emphasized on popularity. This label is assigned to the query point by the maximum vote of the neighbours [32].

**Deep Learning Algorithms:**

**Multi-Layer Perceptron**: Multi layer perceptron (MLP) is a type of feed forward neural network. It consists of three types of layers which are the input layer, output layer and hidden layer, as shown in Fig. 2.4 . The input layer receives the input signal that is needed to be processed. The required task as prediction and classification is done by the output layer. An arbitrary number of hidden layers that are placed between the input and output layer are the true algorithm engine of the MLP. [25] The purpose of the hidden layers is to increase the learning of model in order to increase the accuracy of the results.

Quite similar to a type of feed-forward type of network, in a MLP the flow of data is in the forward direction from the input to the output layer. The neurons present in the Multi layer perceptron are trained and tested with the backpropagation of learning algorithm. MLPs are designed in such a way so that they can approximate any continuous function present and is able to solve problems that are not linearly separable. The primary use cases of MLP are in pattern classification, recognition, prediction and approximation.

## 2.4 VeReMi Dataset

VANET is been deployed on the road side, where vehicles have mobile nodes. Active security and intelligent transportation are important applications of VANET, which need suitable vehicle-to-vehicle communication technology, especially routing technology. Heijden et al.

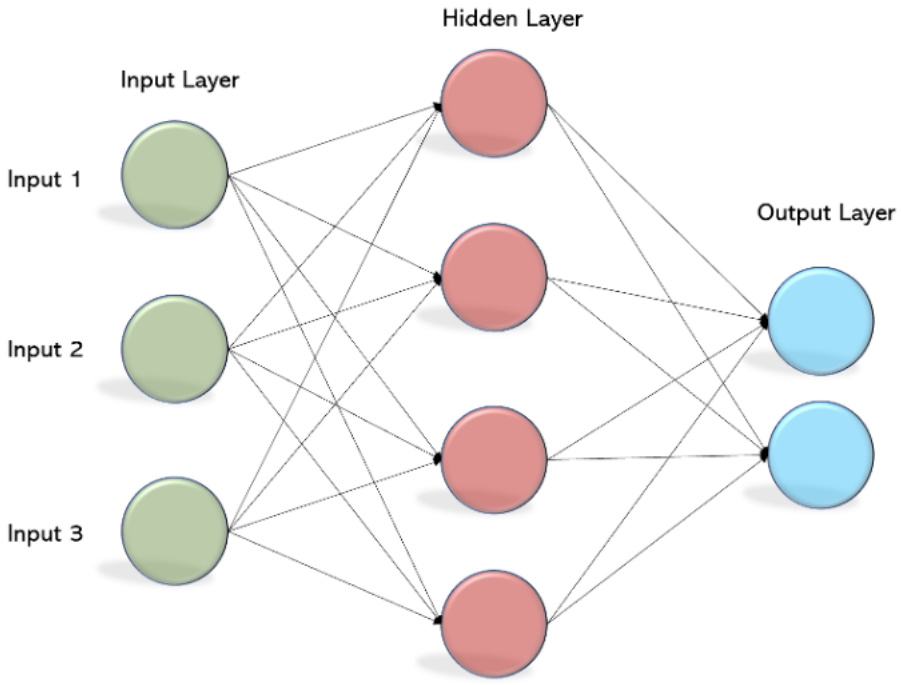


FIGURE 2.4: Multi-Layer Perceptron [25]

[35] introduced the first public extensible dataset, namely Vehicular Reference Misbehavior Dataset (VeReMi). VeReMi is a simulated dataset, generated using LuST (Version 2) and VEINS (with modifications, based on Version 4.6). It consists of message logs per vehicle, containing both GPS data (labelled as type=2) about the local vehicle and BSM messages (labelled as type=3) received from other vehicles through DSRC. It has two primary purposes: it serves as a baseline to assess how misbehaviour detection mechanisms operate on a city scale, and it saves you a lot of computational power typically required to run VEINS sufficiently often. VeReMi consists of three different density levels, five different attacks, and three different attacker densities [35]. The code and configuration files that are the input of VEINS are available in a separate repository on the securecomm2018 branch

| Simulation Parameter | Value | Description |
|---|---|---|
| Duration | 100s | Total Duration of Simulation |
| Vehicle Density | (3,5,7)h | 3:low Density, 5 Medium Density, 7 High Density |
| Attacker Density | 0.1,0.2,0.3 | 10, 20 and 30 percent attacker density |

TABLE 2.1: Simulation Parameter of VeReMi dataset

Each simulation log contains a ground truth file for every message and a set of message

logs for every vehicle that received messages. The filename of a message log identifies the receiver by vehicle number and OMNeT++ module number, e.g., JSONlog-0-7-A0.json refers to the 0th vehicle with OMNeT++ module ID 7. The latter is the number also used to identify the sender as such in any reception log and the ground truth file. A0 refers to the fact that this vehicle is not an attacker.

The VeReMi dataset includes five different types of osition falsification attacks, as discussed below:

1. **Constant attack**: In this type of attack, the attacker vehicle continuously broadcasts fixed position coordinates in the BSM. This attack could misguide genuine vehicles into thinking of it as congestion ahead on the road.
2. **Constant offset Attack**: In this type of attack, the attacker vehicle adds a fixed offset/fixed value to the genuine position and transmits the network's altered position. This attack is quite hard to detect as the attacker seems to be behaving generally by slightly manipulating the genuine position in the BSM.
3. **Random Position Attack**: In the random position attack, the attacker transmits any randomized position using the simulation area of the network. It creates severe chaos in the network as every consecutive BSM will have an entirely new, different and random value from the simulation.
4. **Random Offset Position Attack**: In this particular type of attack, the attackers transmit any random value from a pre-defined area around their vehicle. This attack seems quite similar to a constant offset attack as both attacks slightly manipulate the position information.
5. **Eventual Stop Attack**: In the eventual stop attack, the attacker tries to behave normally for some sample amount of time in the network and once the attacker gains the trust, then suddenly transmits a fixed position repeatedly to act as an eventual stopping of the vehicle. This type of attack can misguide the legitimate vehicles by gaining trust in the network for some time and then deceive them.

The attack types descriptions, along with examples, are summarized the Table 2.2

A part of these vehicles is malicious, which is virtually created using a uniform distri-

| Attack Types in VeReMi Dataset | | | |
|---|---|---|---|
| Attacker Type | Attack Name | Description | Example |
| 1 | Constant | Vehicle transmits a fixed position. | x=5560, y=5820 |
| 2 | Constant Offset | Offset added to vehicle's actual position | x = 250,y = 150 |
| 4 | Random | Transmits random position from simulation area. | Random in Simulation area. |
| 8 | Random Offset | Random position from preconfigured rectangular area around the vehicle. | x,y uniformly random from [300, 300] |
| 16 | Evetual Stop | Attacker behaves normally for some time and then transmits current position repeatedly. | Stop probability + = 0.025 with each position update |

Table 2.2: Summary of Attack Types in VeReMi

bution measure. Then these attacker vehicles send wrong position coordinates in the form of BSMs. Message log files will record the altered position sent by an attacker vehicle, but the ground truth file, which is the primary file for the transmission, maintains the vehicle's original position coordinate. A vehicle can also receive zero BSMs if it does not move closer to another vehicle present in the network.

Attacker type value helps to determine between genuine vehicles and attacker vehicles. By default, the attacker type for genuine vehicles is set to 0, while it is 1,2,4,8,16 are set for different attacks, as shown in Table 2.2. VeReMi dataset is generated on a large traffic scenario, including many highways, prime city and street regions. In this research, the VeReMi dataset provides a standard dataset which is the Single BSM dataset is further extended into two consecutive BSM datasets for misbehaviour classification of five different position falsification attacks.

## 2.5 Literature Review

Machine learning is the desired method these days for misbehaviour detection in VANET. Cryptographic frameworks majorly named the PKI model primarily authenticate the vehicle's identity in the network but do not ensure message authenticity. An add-on model such as machine learning or deep learning can enhance message legitimacy. Machine learning or Deep learning helps to identify the features of a highly volatile vehicular network. It is quite a data-centric approach to maximize network performance by reducing the number of vulnerabilities present in the network. Some of the machine learning or deep learning approaches are explained in this section. Comparative analysis of the literature review is given in Table 2.3

Table 2.3: Position Falsification Detection Schemes

| No. | Paper | Machine and Deep Learning Algorithm Used | VeReMi dataset used | Approach Used |
|---|---|---|---|---|
| 1 | Heijden et al [35] . | No | Yes | Belief theory Approach |
| 2 | Steven So et al [31]. | SVM | No | Plausability Checks |
| 3 | Sohan Gyawali et al [8]. | Random Forest | Yes | Sender Receiver Approach |
| 4 | Mayank Dave et al [15]. | Random Forest | Yes | Ensemble Method Approach is used |
| 5 | J Grover et al [7]. | Naive Bayes | Yes | Ensemble Method Approach is used |
| 6 | P Singh et al [30]. | SVM | Yes | Normalizing the position matrix |
| 7 | J Kamel et al [13]. | SVM | Yes | By predicting New Position |
| 8 | J Montenegro et al [20]. | KNN | Yes | By trust based model |
| 9 | A Sharma et al [27]. | KNN | Yes | Vehicle RSU Approach |
| 10 | Propsed Method | MLP | Yes | Deep Learning Approach in Vehicle RSU |

### 2.5.1 Misbehabiour Detection In VANET

In paper [35], the authors introduced a framework which is called Maat, which mainly ensures the validity of received data transmitted by the sender. Maat is a type of framework based on subjective logic - which can be understood as a mathematical framework that enables unpredictability through objects known as subjective opinions on data. Subjective opinions can be considered as a relationship between actors and objects that can express their confidence with a degree of unpredictability. It is based on belief theory

similar to Dempster-Shafer's theory. Maat uses this information to generate a fusion and data management system to determine the confidence of data. Maat uses a directed graph. The authors used four comparison checks for performance evaluation of the model, namely Acceptance Range Threshold (ART), Sudden Appearance Warning (SAW), Simple Speed Check (SSC), and Distance Moved Verifier (DMV).

In paper [31] the authors proposed integrating plausibility checks and a machine learning framework for misbehaviour detection using the sender-receiver pair approach in the VeReMi dataset. In the paper approach, they added features: numbering from 1-6, using the two checks it become capable of identifying the wrong location and the remaining four checks are being used for quantitative information that is being used to detail vehicle's behaviour present in the network. The two major plausibility checks that authors include are movement plausibility checks and location.

In recent research by Gyawali et al. [8], they introduced a misbehaviour detection model for both false alert verification schemes and position falsification attacks. This available framework is also dependent on the sender-receiver approach. A fake alert is generated when the attacker transmits a wrong alert to the in-range vehicles part of the network. These alerts primarily include hazard condition notification, vehicle stopping warnings. In the proposed paper framework, the authors equipped each vehicle with a misbehaviour detection model..

In paper [15], which assumes a linear speed-density relationship to estimate uninterrupted traffic. The receiver vehicle calculates the change in its speed, position and difference in sender vehicles speed, position, receiving distance and RSSI value

In paper [7], a machine learning approach is being used to classify multiple misbehaviours in VANET using concrete and behavioural features of each node that sends safety packets. A security framework is designed to differentiate a malicious node from a legitimate node. Various types of misbehaviours occur in VANET by tampering with information present in the propagated packet. These misbehaviours are classified based upon multifarious features like speed-deviation of node, received signal strength (RSS), several packets delivered, dropped packets etc. Two types of classification accuracies are measured: Binary and Multi-

Class. In Binary classification, all types of misbehaviours are considered to be in a single "misbehaviour" class whereas, Multi-class classification can categorize misbehaviours into particular misbehaving classes.

In paper [30] the goal is to analyze safety messages and detect false position information transmitted by the misbehaving nodes. In this paper, machine learning (ML) techniques on VeReMi dataset to detect misbehaviour are being used. Demonstrated that the ML-based approach enables high-quality detection of modelled attack patterns.

In paper [13] proposed a machine learning algorithm that tries to predict the next available position of the vehicle present in the network. The authors here use beacon messages as BSM's from neighbouring nodes and create features such as paths between transmitter and receiver. ML algorithms were used for the training and testing of the model. The paper compares the calculated value with the actual value in the BSM and classifies the vehicles based on the comparison done on the actual and predicted values. If the position is not the same as prediction, it is then classified as an attacker vehicle. The authors claimed that Random Forest performs best among other algorithms.

In paper [20], the proposed method aims to evaluate parameters used for the computation of trust metrics applying machine learning techniques. Results show the superior discriminative power of the receiver power coherency metric when detecting misbehaving nodes based on fake position attacks. The approach has a data-oriented model, which defines direct trust and the trust metric that is being calculated depends on the transmitted message's data. The values used for the trust computation are provided by hello messages, also known as beacon messages, which are periodically exchanged by the nodes of the network. The the position and the received power coherency is a useful metric to detect misbehaving nodes in VANETs, i.e., vehicles that announce a fake position in their hello messages.

The paper [27], proposes a novel and efficient data-centric approach to detect location spoofing using machine learning algorithms. Presented a novel ML-based approach for detecting position falsification attacks in VANET. Unlike existing techniques considering individual BSMs, the model uses pairs of consecutive BSMs from vehicles to create an aug-

mented dataset. The augmented dataset combines selected features from the individual BSMs and trains the different ML algorithms. Comparison between approaches with four different algorithms and found KNN and Random Forest classifiers yield the best results. Then compared the results of our approach (using KNN) with other recent techniques available in the literature, using the same VeReMi dataset.

In this research, the proposed methodology uses the vehicle-RSU pair approach for position falsification detection. Deep learning and Machine Learning algorithms are used to classify legitimate vehicles and attacker vehicles.

# Chapter 3

# Single And Consecutive BSM based classification

## 3.1 Introduction

Misbehaviour detection is a class of VANET that deals with the identification of attacks on VANET using various methods. In my research, I'm targeting to detect the position falsification attack in VANET using Deep Learning Techniques. The basic idea is that in an autonomous vehicle environment, the vehicle transmits BSMs into the network, and all the nearby vehicles and infrastructures can receive these BSMs. The BSM's provide us with the position, speed, heading and other relevant parameters of the sending vehicles. BSM's data is used to identify the behaviours of an attacker vehicle. VeReMi dataset is a collection of the BSMs which is being used for the detection of misbehaving vehicles. The proposed methodology aims at:

1) Providing machine learning and deep learning models to correctly identify position falsification attacks.

2)Comparing single and two Consecutive BSM's based approaches using the deep learning

## 3.2 Proposed Architecture

The security of BSMs is ensured by the cryptographic techniques using encryption and decryption. The encryption technique works on the concept of the public and private keys. The user digitally signs the messages before sending them to the network. The generic approach tends to work as all the registered vehicles in the network send BSM to the other vehicles present in the network. The major drawback of this approach is that it cannot protect against insider attacks from vehicles with valid credentials. The reason that it's not able to detect the attacker vehicle is because the vehicle is part of the network and as per the generic technique the vehicle which has the valid credentials is considered to be the legitimate vehicle.

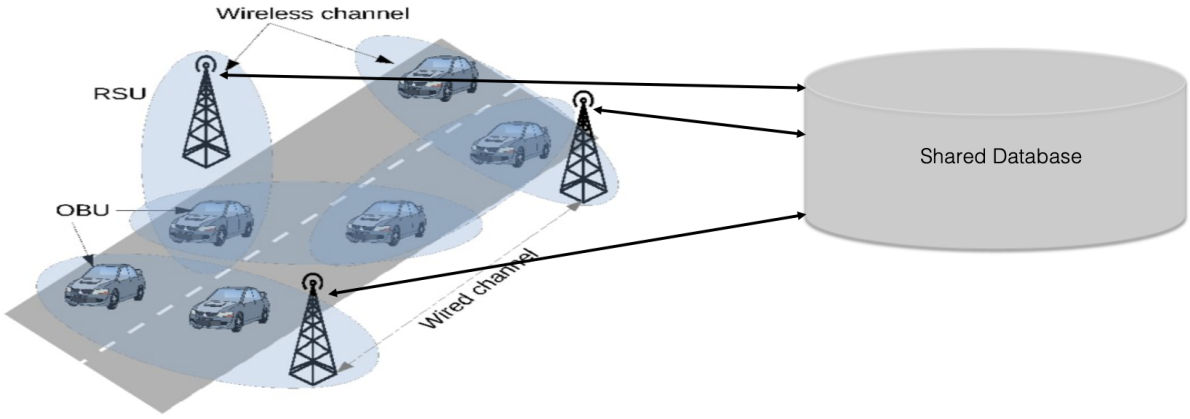


FIGURE 3.1: Proposed Architecture [26]

The proposed approach is an addition of security to the present cryptographic techniques. Vehicles in the network send BSMs, which are received by the RSU and other nodes in the VANET. The received BSMs in the network are then updated in the shared database. The update in the database is done using the transmit time of BSM as shown in Fig 3.1. After receiving the BSM at the RSU, the RSU verifies the correctness of the BSM. The proposed detection framework installed at the RSU retrieves the latest received BSM from the vehicle from the shared database using a unique sender ID assigned to each vehicle during registration. The proposed model creates two types of databases using the single and two consecutive received BSMs. Using the machine and deep learning model, the classification is being done of legitimate and attacker vehicles. Once the vehicle is identified as an attacker or genuine, the RSU shares the information with other RSUs and vehicles

present in the network.

## 3.3 Outline of Proposed Approach

The proposed methodology majorly depends on the three steps of data extraction, data preparation and classification model. The detailed explanation of the three steps are as follows:

### 3.3.1 Data Extraction

VeReMi dataset is a collection of VANET simulation of 225 indexes divided on the basis of the different traffic densities and different attacker densities. Of all those indexes, each index file has two types of files. The first one is the ground truth file which is the main file that has all the required information regarding the genuine and attacker vehicles. The ground truth file can be understood as the actual behaviour file of the network. The second type of file present in the indexes is the log file. There is only one ground truth file in each index, and there are multiple log files in the index as log files are the acknowledgement of the received BSMs in the network.

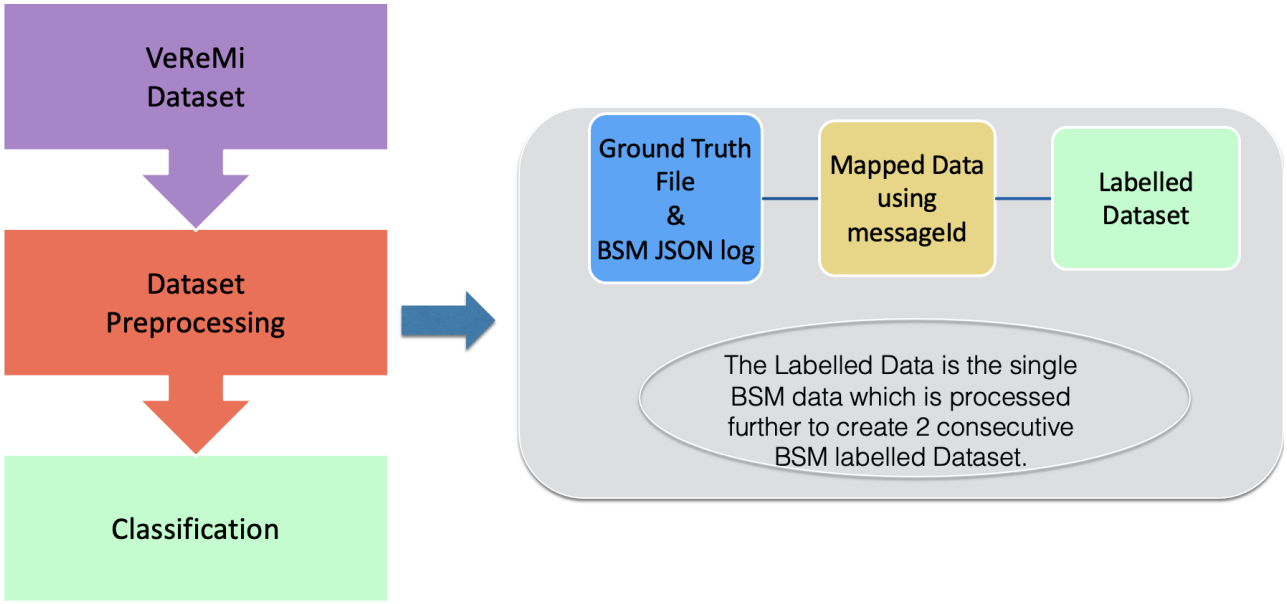


FIGURE 3.2: Data Preparation

Ground truth files and log files must be combined to generate a labelled dataset on which the model can be built as per Fig 3.3. In the data extraction stage, the ground truth file is mapped to log files for each simulation. For a single simulation, the number of log files is equal to the number of receivers; hence the first step is to combine these separate log files into a single file. Ground truth files and log files contain a unique id named messageID. To create a labelled dataset, the ground truth file's attacker type must be mapped to data in the combined log file.

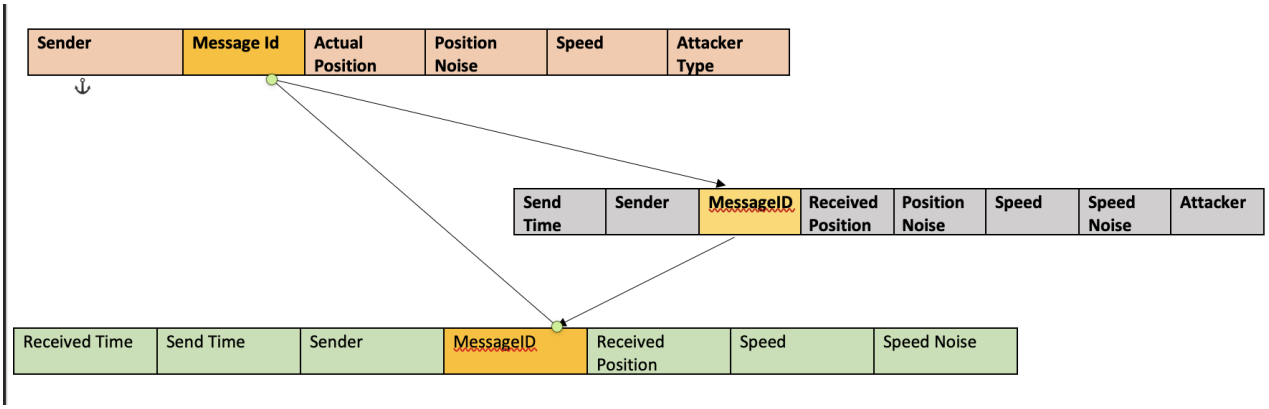


FIGURE 3.3: Generation of the Labelled Dataset

### 3.3.2 Data Preparation

Once the data was successfully extracted, then the merged data is then pre-processed. The central part of pre-processing is combining data and removing the redundancy in the data. As each vehicle has a separate log file, a single BSM was recorded in multiple vehicles, creating duplicates in the dataset. The need to remove redundancy from the dataset is to have a good model and avoid overfitting the model. After the redundancy, the central part was to remove the non-contributing factors. This was the part where the model needs to learn which factors contribute more and which contribute less. The reason to remove the non-contributing element it can affect the accuracy of the model to a great extent.

As per Fig 3.4, the position and speed are the factors playing a significant role in the model compared to the other features. The primary use of the features is to tell which

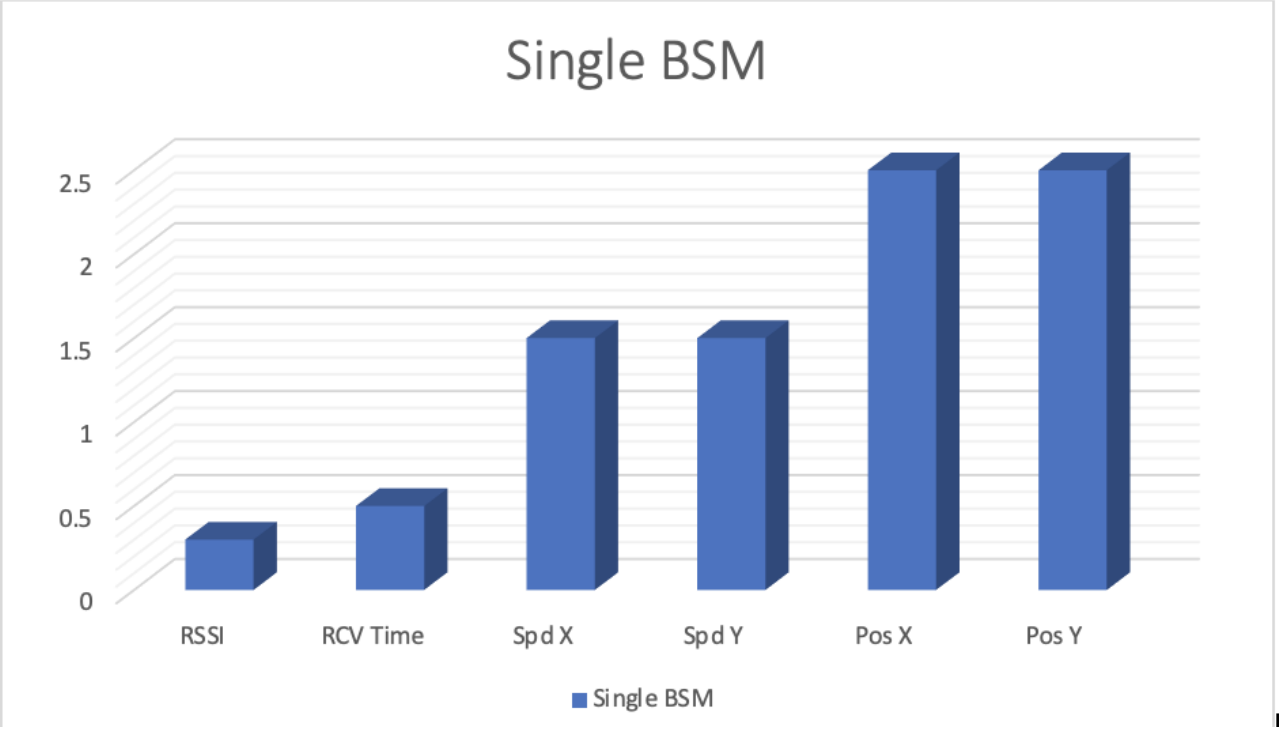


FIGURE 3.4: Contributing Factors

vehicle is not behaving accurately in the network. With the single BSM, the information available to detect misbehaviour is much less compared to the 2 BSM model. The significant factors, as discussed above, are speed and position, which play a vital role in both BSMs models. Many of the other non-contributing features have been removed to increase the learning of the model. Some of the non-contributing features of the dataset are sender id, receiver id and position noise vector.

Table 3.1 shows an example of the items of the dataset, when using features from a single BSM.

| Vehicle No. | pos.0 | pos.1 | pos.2 | spd.0 | spd.1 | message id |
|---|---|---|---|---|---|---|
| 1 | 5409.774101 | 5794.384047 | 1.895 | -11.81679849 | -3.873118663 | 973 |
| 2 | 4432.548015 | 5295.893829 | 1.895 | 3.756319888 | -30.33504372 | 1222 |
| 3 | 6224.82792 | 6021.888413 | 1.895 | -15.28586521 | 5.422040364 | 1857 |

TABLE 3.1: Example of Single BSM Dataset

When using two consecutive BSMs, the features include x and y coordinates of position and speed for both BSM1 and BSM2. An example of a two-consecutive BSM dataset is shown in Table 4.2:

| pos.X1 | pos.Y1 | spd.X1 | spd.Y1 | Pos X2 | Pos Y2 | Spd X2 | Spd Y2 |
|---|---|---|---|---|---|---|---|
| 3588.8 | 5912.4 | 1.277 | 30.02 | 3590.48 | 5942.29 | 2.22 | 29.93 |
| 3590.48 | 5942.29 | 2.228 | 29.93 | 3592.70 | 5972.12 | 2.22 | 29.87 |
| 3592.70 | 5972.12 | 2.224 | 29.87 | 3594.92 | 6001.93 | 2.22 | 29.90 |

TABLE 3.2: Example of Consecutive BSM Dataset

### 3.3.3 Classification

The third and final stage of the proposed model is the classification of the dataset. In this thesis, we will implement both types of classification using machine learning and deep learning models. The machine learning binary classification is performed on separate attacks, and all five position falsification attacks are combined in a single BSM dataset and two consecutive BSM datasets. The machine learning algorithm used for classification is K-Nearest Neighbours, and the Deep Learning algorithm used for classification is Multi-Layer Perceptron. Classifiers give a better correct classification rate out of the machine, and deep algorithms will be used in a detection framework. The correct classification rate depends on the performance metrics, which are explained in detail in section 4.1.4. These algorithms train the model using a training set and classify the future data as legitimate or attacker.

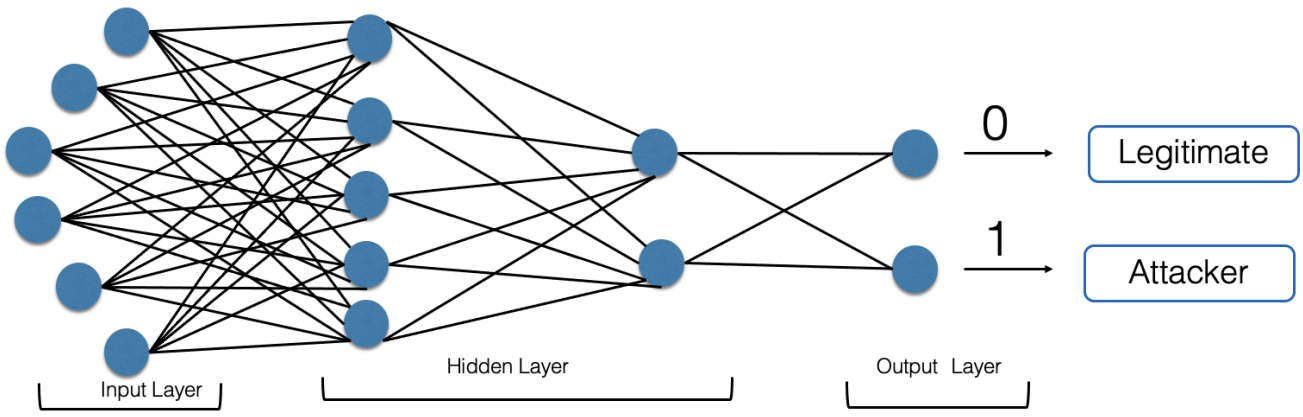


FIGURE 3.5: Multi-Layer Perceptron

The major part of the classification is using deep learning, for that, Multi-Layer Perceptron is being used. The MLP is a type of feedforward artificial neural network (ANN). For our classification model, we used two hidden layers for the neural network; in the first hidden layer, we used five neurons. The reason to use five neurons for the first hidden layer is that it is recommended that the number of neurons should be less than the number of features [10]. As the features used for the model were six, so the first layer had five neurons. The second hidden layer has two neurons. The second layer plays a crucial role in the output generated. As the output of the model is binary, classifying a vehicle legitimate or attacker, thus the second hidden layer absorbs the extra features and presents the binary result of the model. For the activation function, Rectified Linear Unit (ReLU) [23] is used as the linear function so that input yields a positive output, otherwise, it will output zero. The reason to use it with MLP in a model is that it is easier to train and gives better results.

The MLP classifier model implemented to detect the position falsification is Limited memory-Broyden Fletcher Goldfarb Shanno [21], which is an optimization algorithm in the family of quasi-Newton methods and uses cross-entropy loss function [36]. This algorithm measures the performance of a classification model whose output is a probability value between 0 and 1. The primary purpose of the optimizer is when the model is learning by the dataset. Once the learning is done, the model tries to predict the result of the test data. The cross-entropy loss function decides whether the algorithm is performing correctly by using the loss function. If the value of the loss function increases, then the optimizer adjusts the parameters to improve accuracy.

### 3.3.4 Sender-RSU Based approach

Many researchers have introduced a misbehaviour detection model to detect position falsification attacks using VeReMi dataset. Some of the occurring approaches use features of change seen in speed and position of the vehicle to train the model, some other uses the trust-based models to detect an attack. The majority of them have worked on sender-receiver pairs to identify misbehaviour in the network.

In the sender-receiver pair approach, as many researchers do, a detection framework is configured on the OBU in vehicles to detect misbehaviour. In this proposed methodology, a

| No. | Sender-Receiver Approach | Sender-RSU Approach |
|---|---|---|
| 1 | Detection Performed at OBU | Detection performed at RSU |
| 2 | Computation overhead on OBU | No computational overhead at OBU |
| 3 | Sender Receiver Approach | Sender RSU Approach |
| 4 | Machine Learning Approach is used | Deep Learning is Used height |

TABLE 3.3: Comparison of sender-receiver and sender-RSU approaches

single BSM calculation and two consecutive BSMs are considered features in a dataset. For the proposed method, the detection framework is installed on the RSU rather than OBU, thus reducing computational overhead on the vehicles. The proposed method alleviates the V2V dependency in the network as RSU gives a clearer view of any misbehaviour in the network. For sender-receiver-based approaches, the attacker's vehicle needs to be in the range of other legitimate vehicles to get detected. In the vehicle-RSU pair approach, the detection of the attack can potentially be done even before it comes within the range of the legitimate vehicles.

# Chapter 4

# Results

In order to avoid high infrastructure costs, facilities, and resource requirements, the experiments were performed in a virtual environment using simulation tools . This is a relatively economical and safe way of determining the performance of algorithms. Chapter 4 has setup discussion regarding simulation tools and parameters used in the VeReMi dataset, experimental setup toolkits, classification parameters, and evaluation metrics for measuring the proposed classification model's performance. Also, apart from the experimental parameters this chapter presents the simulation results for the model and comparisons with the existing approaches

## 4.1 Setup

### 4.1.1 VeReMi Dataset

The VeReMi dataset is being used in the research, which is a publicly available dataset for VANET. The dataset is prepared by using three tools which are SUMO, VEINS and OMNET++. SUMO (Simulation Of Urban Mobility) is being used to produce traffic density present in the network. The Vehicular Reference Misbehavior (VeReMi) dataset evaluates misbehaviour detection mechanisms for VANETs (vehicular networks). This dataset consists of message logs of onboard units, including a labelled ground truth file, generated from a simulation environment, the simulation parameters used to generate the dataset are

shown in Table 4.1. The dataset includes malicious messages intended to trigger incorrect application behaviour, which misbehaviour detection mechanisms aim to detect. The initial dataset contains several simple attacks: the idea of this dataset release is to provide a baseline for comparing detection mechanisms and serve as a starting point for more complex attacks. [12] :

| **Parameter** | **Values** |
|---|---|
| Mobility | SUMO Lust |
| Simulation Start | (3,5,7) Vehicle Density |
| Simulation Duration | 100s |
| Attacker Probability | (0.1,0.2,0.3) Attacker Density |
| Simulation Area | 2300,5400,6300 Types of road |
| Signal Interfernce Model | Two Ray Interference |
| Obstacle Shadowing | Simple |
| Fading | Jakes |
| Shadowing | Log-Normal |
| MAC implementation | 802.11p |
| Thermal Noise | -110dbm |
| Transmit Power | 20mV |
| Bit Rate | 6Mbps |
| Sensitivity | -89dbm |
| Antenna Model | Monopole on Roof |
| Beaconing Rate | 1Hz |

Table 4.1: Simulation Parameters [12]

### 4.1.2 Attacks Implementation

The attackers we consider are of 5 types: the constant attacker, the random attacker, the constant offset attacker, the random offset attacker, and the eventual stop attacker. The regular attacker transmits a fixed, preconfigured position; the constant offset attacker transmits a fixed, preconfigured offset added to their actual position; the random attacker sends a random position from the simulation area; the random offset attacker sends an arbitrary position in a preconfigured rectangle around the vehicle; the eventual stop attacker generally behaves for some time, and then attacks by transmitting the current position repeatedly that is if it has stopped. The random attacks (4 and 8) take a new random sample for every message. The 5 attackers with their respective attack paramters are shown in Fig. 4.1

| ID | Attack | Parameters |
|---|---|---|
| 1 | Constant | $x = 5560, y = 5820$ |
| 2 | Constant offset | $\Delta x = 250, \Delta y = -150$ |
| 4 | Random | uniformly random in playground |
| 8 | Random offset | $\Delta x, \Delta y$ uniformly random from $[-300, 300]$ |
| 16 | Eventual stop | stop probability $+= 0.025$ each position update $(10Hz)$ |

FIGURE 4.1: Attack Parameters [12]

The available VeReMi dataset contains 225 indexes simulation, split into three types of density categories that can be further classified. The low-density Vehicles have 35 to 39 vehicles. In comparison, the medium density contains between 97 and 108 vehicles, and the high-density vehicles include between 491 and 519 vehicles. Out of these available vehicles, a part of the dataset includes malicious vehicles. The required decision is made by sampling a uniform distribution ([0, 1]) and comparing it to the attacker fraction parameter, essentially assigning each vehicle to be an attacker with that probability.

### 4.1.3 Dataset Analysis and Classification Parameters

In this research, four different traffic scenarios are combined to create four datasets, as shown in Table. 4.2. A combination of low, medium, high and merged attacker and vehicle densities are combined to evaluate the proposed model in all four cases. In the current research, we will refer to the above-mentioned dataset combinations as low, medium, high and merged-density datasets. All the attacks are evaluated in low, medium, high, and merged density to measure the impact of vehicle and attacker density on the proposed model's performance. An individual simulation in the VeReMi dataset contains multiple JSON log files, which are combined into one single log file, and the "Attacker type" label present in the Ground truth file is combined with the log file to generate a labelled dataset. This process is repeated for all five repetitions. Extraction of data is done by downloading the simulation scenarios and generating mapped data from these files.

Pre-processing the data is done by filtering out non-contributing features and removing duplicate data is implemented using a Python script. After generating a clean, pre-processed

| S.No. | Attack Types | Vehicle Density | Attacker Density | Repetition |
|---|---|---|---|---|
| 1 | 1,2,4,8,16 | LOW(3) | LOW(0.1) | 0 to 4 |
| 2 | 1,2,4,8,16 | MEDIUM(5) | MEDIUM(0.2) | 0 to 4 |
| 3 | 1,2,4,8,16 | HIGH(7) | HIGH(0.3) | 0 to 4 |
| 4 | 1,2,4,8,16 | MERGED(3,5,7) | MERGED(0.1,0.2,0.3) | 0 to 4 |

TABLE 4.2: Example of Consecutive BSM Dataset

two-consecutive and Single BSM datasets are generated on which , we perform classification. The classification includes the following:

**Model Used** A model is being used to perform classification. There are different algorithms for classification, as discussed in section 2.3.4. In this research, two types of learning are being used i) a standard classification algorithm, viz., K Nearest Neighbour, and ii) a Deep Learning algorithm, viz., Multi-Layer Perceptron.

**Cross-validation** Cross-validation is a type of re-sampling procedure that is being used to evaluate machine or deep learning models on a limited data set. The procedure contains a single parameter called k that refers to the number of groups that a given data sample is split into. As such, the procedure is often defined k-fold cross-validation [2], when a specific value for k is chosen. In the proposed model, k=100 is used, which creates 100-fold cross-validation. Cross-validation is used in applied machine or deep learning to estimate the skill of a machine or deep learning model on unseen data.

### 4.1.4 Evaluation Metrics

The VeReMi dataset contains both the types of vehicles, the attacker and the legitimate vehicles. The primary thing is that though the dataset has two kinds of distribution, the dataset is imbalanced. Due to imbalanced data, the accuracy can't be considered the optimal way to determine the performance of the model. To assess the correctness of the imbalanced data apart from the accuracy, the precision, recall and F1 score are determined using the confusion matrix. The confusion matrix can be understood as the table showing the correct and incorrect prediction in tabular form as shown in Fig 4.2.

**Precision** The model precision score represents the model's ability to correctly predict

| | | Predicted Label | |
|---|---|---|---|
| | | Predicted Negative | Predicted Positive |
| Actual Label | Actual Negative | True Negative | False Positive |
| | Actual Positive | False Negative | True Positive |

FIGURE 4.2: Confusion Matrix

the positives out of all the positive predictions it made. The precision score is a valuable measure of success of prediction when the classes are very imbalanced [37]. Mathematically, it presents the ratio of true positive to the sum of true positive and false positive.

$$Precision = \frac{TruePositive}{TruePositive + FalsePositive} = \frac{TruePositive}{TotalPredictedPositive}$$

**Recall** Model recall score represents the model's ability to accurately predict the positives out of actual positives. This is unlike precision which measures how many predictions made by models are positive out of all positive predictions made [3].

$$Recall = \frac{TruePositive}{TruePositive + FalseNegative} = \frac{TruePositive}{TotalActualPositive}$$

**F1-Score** Model F1 score presents the model score as a function of precision and recalls score. F1 - score is a machine learning model performance metric that gives equal weight to both the Precision and Recalls for measuring the performance in terms of accuracy, making

it a better alternative to Accuracy metrics (it doesn't require us to know the total number of observations) [16].

$$F1 = 2X\frac{Precision * Recall}{Precision + Recall}$$

### 4.1.5 Environment and Tools used

All the simulations conducted for this research were conducted in the following environment and configuration:

• Operating system: MacBook Air - macOS Big Sur • Processor: 2.3 GHz Quad-Core Intel Core i5

• Memory: 8 GB

Tools and libraries used for the implementation of this research are: • Programming language: Python 3.7

• Integrated Development Environment: Visual Studio Code

• Libraries: Scikit-learn, matplotlib, NumPy, pandas

## 4.2 Classification Results for Two Consecutive BSM

In this section, we implemented the algorithms on the two consecutive BSM dataset created using the VeReMi dataset. The machine learning algorithm used is the K Nearest Neighbour, and the deep learning algorithm used is the Multi-Layer Perceptron on each of the attack types, and the tables below represent the precision, recall and F1 score of the algorithms on two BSM' data.

**Attack - 1** : The reason that both the algorithms show such promising results for attack type 1 is because a vehicle constantly transmits a fixed location but not a fixed velocity, making it easily observable which makes the learning easy for the model which can be seen in all the four tables.

| Attack -1 | | | |
|---|---|---|---|
| Algorithm | Precision | Recall | F1-Score |
| K-Nearest Neighbour | 100 | 100 | 100 |
| Multi-Layer Perceptron | 100 | 100 | 100 |
| Attack -2 | | | |
| Algorithm | Precision | Recall | F1-Score |
| K-Nearest Neighbour | 100 | 100 | 100 |
| Multi-Layer Perceptron | 95 | 93 | 94.6 |
| Attack -4 | | | |
| Algorithm | Precision | Recall | F1-Score |
| K-Nearest Neighbour | 98.9 | 99.2 | 98.9 |
| Multi-Layer Perceptron | 99 | 99.2 | 99.1 |
| Attack -8 | | | |
| Algorithm | Precision | Recall | F1-Score |
| K-Nearest Neighbour | 97.2 | 98.1 | 98.3 |
| Multi-Layer Perceptron | 98.7 | 98.5 | 98.6 |
| Attack -16 | | | |
| Algorithm | Precision | Recall | F1-Score |
| K-Nearest Neighbour | 96.7 | 94.2 | 94.7 |
| Multi-Layer Perceptron | 97.3 | 98.4 | 97.5 |

TABLE 4.3: Classification results of Consecutive BSM model-LOW

**Attack 2** : In this, two consecutive BSM are created as features in the model, making machine and deep learning algorithms able to detect patterns and recognize this attack type as this deals with the constant offset which is relatively easy for the model to learn. KNN algorithms performed exceptionally well as compared to MLP as there attack 2 is a constant offset attack, so the model learns the offset and is able to perform exceptionally well in all the tables except for the Table 4.3 as the low dataset doesn't have sufficient data for the model to learn properly .

**Attack 4** : In the attack type 4 the attack is detected with high precision and recall by both the two algorithms in all four densities. In this attack, the vehicle sends the random position from the simulation playground. With a two-consecutive BSM approach, ML and DL models could detect the attack as there was a range gap between the two position coordinates from a vehicle as per the Table 4.5.

**Attack 8** : Similar to attack type 4, this attack transmits random positions from a fixed area near the vehicle. Since the range distance between two positions is small, detecting this attack is difficult. However, our proposed model performed well with Multi Layer

| Attack -1 | | | |
|---|---|---|---|
| Algorithm | Precision | Recall | F1-Score |
| K-Nearest Neighbour | 100 | 100 | 100 |
| Multi-Layer Perceptron | 100 | 100 | 100 |
| Attack -2 | | | |
| Algorithm | Precision | Recall | F1-Score |
| K-Nearest Neighbour | 99.8 | 99.5 | 97.2 |
| Multi-Layer Perceptron | 100 | 100 | 100 |
| Attack -4 | | | |
| Algorithm | Precision | Recall | F1-Score |
| K-Nearest Neighbour | 98.9 | 99.2 | 98.9 |
| Multi-Layer Perceptron | 99 | 99.2 | 98.1 |
| Attack -8 | | | |
| Algorithm | Precision | Recall | F1-Score |
| K-Nearest Neighbour | 92.5 | 96.5 | 98.3 |
| Multi-Layer Perceptron | 98.7 | 97.5 | 98.6 |
| Attack -16 | | | |
| Algorithm | Precision | Recall | F1-Score |
| K-Nearest Neighbour | 96.7 | 94.7 | 93.5 |
| Multi-Layer Perceptron | 98.3 | 98.4 | 97.5 |

Table 4.4: Classification results of Consecutive model-MEDIUM

| Attack -1 | | | |
|---|---|---|---|
| Algorithm | Precision | Recall | F1-Score |
| K-Nearest Neighbour | 100 | 100 | 100 |
| Multi-Layer Perceptron | 100 | 100 | 100 |
| Attack -2 | | | |
| Algorithm | Precision | Recall | F1-Score |
| K-Nearest Neighbour | 98.9 | 98.6 | 99.4 |
| Multi-Layer Perceptron | 100 | 100 | 100 |
| Attack -4 | | | |
| Algorithm | Precision | Recall | F1-Score |
| K-Nearest Neighbour | 100 | 98.4 | 97.9 |
| Multi-Layer Perceptron | 99 | 99.2 | 99.1 |
| Attack -8 | | | |
| Algorithm | Precision | Recall | F1-Score |
| K-Nearest Neighbour | 96.2 | 98.1 | 97.3 |
| Multi-Layer Perceptron | 98.7 | 97.5 | 98.6 |
| Attack -16 | | | |
| Algorithm | Precision | Recall | F1-Score |
| K-Nearest Neighbour | 96.7 | 93.4 | 94.7 |
| Multi-Layer Perceptron | 98.3 | 98.4 | 97.5 |

Table 4.5: Classification results of Consecutive model-HIGH

| Attack -1 | | | |
|---|---|---|---|
| Algorithm | Precision | Recall | F1-Score |
| K-Nearest Neighbour | 100 | 100 | 100 |
| Multi-Layer Perceptron | 100 | 100 | 100 |
| Attack -2 | | | |
| Algorithm | Precision | Recall | F1-Score |
| K-Nearest Neighbour | 98.9 | 98.6 | 99.4 |
| Multi-Layer Perceptron | 100 | 100 | 100 |
| Attack -4 | | | |
| Algorithm | Precision | Recall | F1-Score |
| K-Nearest Neighbour | 98.9 | 99.2 | 94.7 |
| Multi-Layer Perceptron | 99 | 99.2 | 99.1 |
| Attack -8 | | | |
| Algorithm | Precision | Recall | F1-Score |
| K-Nearest Neighbour | 97.2 | 98.1 | 98.3 |
| Multi-Layer Perceptron | 98.7 | 98.5 | 98.6 |
| Attack -16 | | | |
| Algorithm | Precision | Recall | F1-Score |
| K-Nearest Neighbour | 96.7 | 94.2 | 94.7 |
| Multi-Layer Perceptron | 98.7 | 99.4 | 98.5 |

TABLE 4.6: Classification results of Consecutive model-MERGED

Perceptron classifiers in low and medium density as per Tables 4.4 , 4.3.

**Attack 16** : In this attack, the attacker acts typically for a brief period of time before repeatedly transmitting the exact location in the BSMs. As per the Table 4.6 The model showed no improvement in performance with an increase in the data density. One reason may be that the vehicle is labelled as an attacker even though it is acting normally, confusing the machine learning model.

## 4.3 Comparison of results for Single BSM with Existing Approaches

In addition to two consecutive BSMs, both the Multi-Layer Perceptron and K-Nearest Neighbour algorithms were tested with Single BSM approach. In this case the deep learning model (MLP) clearly outperformed the ML algorithm (KNN).

Tables 4.7, 4.8 and 4.10 show the performance of the two algorithms using single BSM approach for Low, Medium and High vehicle densities respectively; while Table 4.9 shows

| Attack -1 | | | |
|---|---|---|---|
| Algorithm | Precision | Recall | F1-Score |
| K-Nearest Neighbour | 100 | 100 | 100 |
| Multi-Layer Perceptron | 100 | 100 | 100 |
| Attack -2 | | | |
| Algorithm | Precision | Recall | F1-Score |
| K-Nearest Neighbour | 32.6 | 40.2 | 44.3 |
| Multi-Layer Perceptron | 100 | 100 | 100 |
| Attack -4 | | | |
| Algorithm | Precision | Recall | F1-Score |
| K-Nearest Neighbour | 54.8 | 69.2 | 53.9 |
| Multi-Layer Perceptron | 94.6 | 93.2 | 94.7 |
| Attack -8 | | | |
| Algorithm | Precision | Recall | F1-Score |
| K-Nearest Neighbour | 86.7 | 88.4 | 88.9 |
| Multi-Layer Perceptron | 95.7 | 94.8 | 92.9 |
| Attack -16 | | | |
| Algorithm | Precision | Recall | F1-Score |
| K-Nearest Neighbour | 34.6 | 54.8 | 34.7 |
| Multi-Layer Perceptron | 91.3 | 90.4 | 92.6 |

TABLE 4.7: Classification results of Single BSM Proposed model-LOW

the results for the merged dataset with different vehicle densities. For all cases the DL model achieves consistently higher values of percision, recall and F1-score. The performance of the DL model is somewhat lower for low vehicle densities, as it is a data-hungry model (Table 4.7), where due to the lower number of vehicles and attacks there is less opportunity for the model to learn effectively. Still, compared to the KNN, the performance is significantly higher.

Based on these observations, we conclude the the DL model can achieve high quality results, even with the simpler single BSM approach; whereas, the performance of the ML model is quite poor in this case. To achieve acceptable results using ML model, it is necessary to use the consecutive BSM approach, which increases the complexity of the algorithm needed to create the items in the dataset. This is because in consecutive BSM approach, the previous BSMs must be stored and retrieved in real time, which is not required for the single BSM approach.

| Attack -1 | | | |
|---|---|---|---|
| Algorithm | Precision | Recall | F1-Score |
| K-Nearest Neighbour | 100 | 100 | 100 |
| Multi-Layer Perceptron | 100 | 100 | 100 |
| Attack -2 | | | |
| Algorithm | Precision | Recall | F1-Score |
| K-Nearest Neighbour | 46.7 | 40.2 | 44.7 |
| Multi-Layer Perceptron | 100 | 100 | 100 |
| Attack -4 | | | |
| Algorithm | Precision | Recall | F1-Score |
| K-Nearest Neighbour | 95.8 | 79.2 | 52.50 |
| Multi-Layer Perceptron | 99.8 | 99.5 | 99.3 |
| Attack -8 | | | |
| Algorithm | Precision | Recall | F1-Score |
| K-Nearest Neighbour | 87.4 | 88.6 | 89.8 |
| Multi-Layer Perceptron | 97.7 | 98.6 | 98.6 |
| Attack -16 | | | |
| Algorithm | Precision | Recall | F1-Score |
| K-Nearest Neighbour | 38.8 | 43.6 | 45.4 |
| Multi-Layer Perceptron | 97.3 | 98.4 | 97.5 |

Table 4.8: Classification results of Single BSM Proposed model-MEDIUM

| Attack -1 | | | |
|---|---|---|---|
| Algorithm | Precision | Recall | F1-Score |
| K-Nearest Neighbour | 100 | 100 | 100 |
| Multi-Layer Perceptron | 100 | 100 | 100 |
| Attack -2 | | | |
| Algorithm | Precision | Recall | F1-Score |
| K-Nearest Neighbour | 35.7 | 40.2 | 45.6 |
| Multi-Layer Perceptron | 100 | 100 | 100 |
| Attack -4 | | | |
| Algorithm | Precision | Recall | F1-Score |
| K-Nearest Neighbour | 88.9 | 69.2 | 62.5 |
| Multi-Layer Perceptron | 99.3 | 98.2 | 97.3 |
| Attack -8 | | | |
| Algorithm | Precision | Recall | F1-Score |
| K-Nearest Neighbour | 87.4 | 88.6 | 89.8 |
| Multi-Layer Perceptron | 97.7 | 98.9 | 98.1 |
| Attack -16 | | | |
| Algorithm | Precision | Recall | F1-Score |
| K-Nearest Neighbour | 27.8 | 56.9 | 45.7 |
| Multi-Layer Perceptron | 98.3 | 99.4 | 96.5 |

Table 4.9: Classification results of Single BSM Proposed model-HIGH

| Attack -1 | | | |
|---|---|---|---|
| Algorithm | Precision | Recall | F1-Score |
| K-Nearest Neighbour | 100 | 100 | 100 |
| Multi-Layer Perceptron | 100 | 100 | 100 |
| Attack -2 | | | |
| Algorithm | Precision | Recall | F1-Score |
| K-Nearest Neighbour | 35.7 | 40.2 | 45.6 |
| Multi-Layer Perceptron | 100 | 100 | 100 |
| Attack -4 | | | |
| Algorithm | Precision | Recall | F1-Score |
| K-Nearest Neighbour | 95.9 | 49.2 | 62.5 |
| Multi-Layer Perceptron | 99.6 | 99.2 | 99.3 |
| Attack -8 | | | |
| Algorithm | Precision | Recall | F1-Score |
| K-Nearest Neighbour | 87.4 | 88.6 | 84.8 |
| Multi-Layer Perceptron | 98.7 | 98.5 | 98.6 |
| Attack -16 | | | |
| Algorithm | Precision | Recall | F1-Score |
| K-Nearest Neighbour | 27.8 | 54.5 | 40.7 |
| Multi-Layer Perceptron | 97.3 | 98.4 | 97.5 |

Table 4.10: Classification results of Single BSM Proposed model-MERGED

# Chapter 5

# Conclusion and Future Work

## 5.1 Conclusion

This thesis proposes a novel Deep Learning-based approach for automatically detecting position falsification attacks in VANET. We have created modified datasets that consist of selected features from the individual BSMs based on feature importance and are used to train the proposed model using machine and deep learning algorithms. The performance of the two classification algorithms was compared. It was found that both K-Nearest Neighbour and Multi-Layer Perceptron give excellent results when the Consecutive BSM approach is used. But with the simpler Single BSM dataset, the MLP outperforms the KNN classifiers in yielding the best results. The proposed model is based on the notion of sender and RSU pair approach. This approach aims to reduce the computational overhead from vehicles (OBUs) by designing a detection model built on RSU to detect the attack and provide a broader view for detecting the position falsification attack. It also aims to remove the vehicle-to-vehicle dependency in the network for detecting misbehaviour.

## 5.2 Future Work

The VeReMi dataset, which has been used for this thesis, is limited to five specific types of position falsification attacks and do not fully represent all the possible attacks in VANETs.

The proposed model in this thesis is bound to only the data present in the VeReMi dataset. For the future work of the thesis, the models can be trained to detect other types of position falsification attacks, as well as different attacks such as DoS or replay attacks. In addition to MLP, other Deep Learning algorithms such as Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN) can be investigated, which may yield better results for specific types of attacks.

# Bibliography


[1] Hakim Badis and Abderrezak Rachedi. Modeling tools to evaluate the performance of wireless multi-hop networks. In *Modeling and Simulation of Computer Networks and Systems*, pages 653–682. Elsevier, 2015.

[2] Michael W Browne. Cross-validation methods. *Journal of mathematical psychology*, 44(1):108–132, 2000.

[3] Michael Buckland and Fredric Gey. The relationship between recall and precision. *Journal of the American society for information science*, 45(1):12–19, 1994.

[4] Ahmed El-Mowafy, Nobuaki Kubo, and Allison Kealy. Reliable positioning and journey planning for intelligent transport systems. *Intelligent and Efficient Transport Systems*, page 41, 2020.

[5] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[6] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. Machine learning basics. *Deep learning*, 1(7):98–164, 2016.

[7] Jyoti Grover, Nitesh Kumar Prajapati, Vijay Laxmi, and Manoj Singh Gaur. Machine learning approach for multiple misbehavior detection in vanet. In *International conference on advances in computing and communications*, pages 644–653. Springer, 2011.

[8] Sohan Gyawali and Yi Qian. Misbehavior detection using machine learning in vehicular communication networks. In *ICC 2019-2019 IEEE International Conference on Communications (ICC)*, pages 1–6. IEEE, 2019.

[9] Hamssa Hasrouny, Abed Ellatif Samhat, Carole Bassil, and Anis Laouiti. Vanet security challenges and solutions: A survey. *Vehicular Communications*, 7:7–20, 2017.

[10] Jeff Heaton. *Introduction to neural networks with Java*. Heaton Research, Inc., 2008.

[11] Monika Jain and Rahul Saxena. Overview of vanet: Requirements and its routing protocols. In *2017 International Conference on Communication and Signal Processing (ICCSP)*, pages 1957–1961, 2017.

[12] Joseph Kamel, Michael Wolf, Rens W. van der Hei, Arnaud Kaiser, Pascal Urien, and Frank Kargl. Veremi extension: A dataset for comparable evaluation of misbehavior detection in vanets. In *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, pages 1–6, 2020.

[13] Joseph Kamel, Michael Wolf, Rens W van der Hei, Arnaud Kaiser, Pascal Urien, and Frank Kargl. Veremi extension: A dataset for comparable evaluation of misbehavior detection in vanets. In *ICC 2020-2020 IEEE International Conference on Communications (ICC)*, pages 1–6. IEEE, 2020.

[14] Ankita Khot and Mayank Dave. Position falsification misbehavior detection in vanets. In Nikhil Marriwala, C. C. Tripathi, Dinesh Kumar, and Shruti Jain, editors, *Mobile Radio Communications and 5G Networks*, pages 487–499, Singapore, 2021. Springer Singapore.

[15] Ankita Khot and Mayank Dave. Position falsification misbehavior detection in vanets. In *Mobile Radio Communications and 5G Networks*, pages 487–499. Springer, 2021.

[16] Zachary Chase Lipton, Charles Elkan, and Balakrishnan Narayanaswamy. Thresholding classifiers to maximize f1 score. *ArXiv*, pages 1402–1892, 2014.

[17] Mohamed Nidhal Mejri, Jalel Ben-Othman, and Mohamed Hamdi. Survey on vanet security challenges and possible cryptographic solutions. *Vehicular Communications*, 1(2):53–66, 2014.

[18] Rashmi Mishra, Akhilesh Singh, and Rakesh Kumar. Vanet security: Issues, challenges and solutions. In *2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT)*, pages 1050–1055. IEEE, 2016.

[19] Rashmi Mishra, Akhilesh Singh, and Rakesh Kumar. Vanet security: Issues, challenges and solutions. In *2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT)*, pages 1050–1055, 2016.

[20] Jordan Montenegro, Cristhian Iza, and Mónica Aguilar Igartua. Detection of position falsification attacks in vanets applying trust model and machine learning. In *Proceedings of the 17th ACM Symposium on Performance Evaluation of Wireless Ad Hoc, Sensor, & Ubiquitous Networks*, pages 9–16, 2020.

[21] Philipp Moritz, Robert Nishihara, and Michael Jordan. A linearly-convergent stochastic l-bfgs algorithm. In *Artificial Intelligence and Statistics*, pages 249–258. PMLR, 2016.

[22] FY Osisanwo, JET Akinsola, O Awodele, JO Hinmikaiye, O Olakanmi, and J Akinjobi. Supervised machine learning algorithms: classification and comparison. *International Journal of Computer Trends and Technology (IJCTT)*, 48(3):128–138, 2017.

[23] Prajit Ramachandran, Barret Zoph, and Quoc V Le. Searching for activation functions. *arXiv preprint arXiv:1710.05941*, 2017.

[24] Ram Shringar Raw, Manish Kumar, and Nanhay Singh. Security challenges, issues and their solutions for vanet. *International journal of network security & its applications*, 5(5):95, 2013.

[25] Dennis W Ruck, Steven K Rogers, and Matthew Kabrisky. Feature selection using a multilayer perceptron. *Journal of Neural Network Computing*, 2(2):40–48, 1990.

[26] Mukesh Saini, Abdulhameed Alelaiwi, and Abdulmotaleb El Saddik. How close are we to realizing a pragmatic vanet solution? a meta-survey. *ACM Computing Surveys (CSUR)*, 48(2):1–40, 2015.

[27] Aekta Sharma and Arunita Jaekel. Machine learning approach for detecting location spoofing in vanet. In *2021 International Conference on Computer Communications and Networks (ICCCN)*, pages 1–6. IEEE, 2021.

[28] Muhammad Sameer Sheikh, Jun Liang, and Wensong Wang. A survey of security services, attacks, and applications for vehicular ad hoc networks (vanets). *Sensors*, 19(16), 2019.

[29] Pranav Kumar Singh, Shivam Gupta, Ritveeka Vashistha, Sunit Kumar Nandi, and Sukumar Nandi. Machine learning based approach to detect position falsification attack in vanets. In Sukumar Nandi, Devesh Jinwala, Virendra Singh, Vijay Laxmi, Manoj Singh Gaur, and Parvez Faruki, editors, *Security and Privacy*, pages 166–178, Singapore, 2019. Springer Singapore.

[30] Pranav Kumar Singh, Shivam Gupta, Ritveeka Vashistha, Sunit Kumar Nandi, and Sukumar Nandi. Machine learning based approach to detect position falsification attack in vanets. In *International Conference on Security & Privacy*, pages 166–178. Springer, 2019.

[31] Steven So, Prinkle Sharma, and Jonathan Petit. Integrating plausibility checks and machine learning for misbehavior detection in vanet. In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 564–571, 2018.

[32] Pascal Soucy and Guy W Mineau. A simple knn algorithm for text categorization. In *Proceedings 2001 IEEE International Conference on Data Mining*, pages 647–648. IEEE, 2001.

[33] Mostafa M. I. Taha and Yassin M. Y. Hasan. Vanet-dsrc protocol for reliable broadcasting of life safety messages. In *2007 IEEE International Symposium on Signal Processing and Information Technology*, pages 104–109, 2007.

[34] Ravi Tomar, Manish Prateek, and GH Sastry. Vehicular adhoc network (vanet)-an introduction. *International Journal of Control Theory and Applications*, 9(18):8883–8888, 2016.

[35] Rens W van der Heijden, Thomas Lukaseder, and Frank Kargl. Veremi: A dataset for comparable evaluation of misbehavior detection in vanets. *arXiv preprint arXiv:1804.06701*, 2018.

[36] Vladimir Vovk. The fundamental nature of the log loss function. In *Fields of Logic and Computation II*, pages 307–318. Springer, 2015.

[37] M Norton Wise. *The values of precision*. Princeton University Press, 1997.

[38] Gongjun Yan, Stephan Olariu, and Michele C Weigle. Providing vanet security through active position detection. *Computer communications*, 31(12):2883–2897, 2008.

# Vita Auctoris

| | |
|---|---|
| NAME: | Smarth Kukreja |
| PLACE OF BIRTH: | New Delhi, India |
| EDUCATION: | B.Tech in Computer Science and Engineering, Guru Gobind Singh Indraprastha University, New Delhi, India, (2017) |
| | M.Sc. Computer Science, University of Windsor, Windsor, Ontario, Canada, 2021 |