AN ABSTRACT OF THE THESIS OF

Shahad Almasari for the degree of Master of Science in Computer Science presented on December 10, 2021.

Title: Delay-Aware Resource Management Frameworks for Intelligent Vehicular Networks

Abstract approved: ______________________________
Bechir Hamdaoui

Autonomous vehicles bring great societal benefits but also potential impact and disruption to road safety, traffic congestion, and driving behaviors. One important technology that is indispensable to the success of such systems is vehicular networks. Vehicular networks provide the backbone for ensuring communication and connectivity among vehicles, all crucial to ensuring road safety and traffic control efficiency. One key requirement of such networks is low latency and high reliability, both of which needed for supporting vehicular applications that are often real-time and QoS-sensitive in nature. In this study, we designed a framework that uses integrated emerging network technologies, namely Software-Defined Networking (SDN), Deep Neural Networks (DNN), and Disruption Tolerant Network (DTN), to improve vehicular network latency and reliability. To evaluate the performance of our framework, we used two simulations. In the first simulation, we relied on different scenarios to assess the performance of continuous transmission with and without DNN. We found that while using DNN, the continuous transmissions performed more efficiently. In the second simulation, we show that DTN improves data transmissions in networks with disconnections occurring due to physical barriers. We found that DTN preserves and improves the accuracy of the DNN results. Our approach has made the infrastructure intelligent enough to handle congestion in vehicular networks, thereby improving the latency and reliability of such networks.

Delay-Aware Resource Management Frameworks for Intelligent Vehicular Networks

by

Shahad Almasari

A THESIS

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Master of Science

Presented December 10, 2021
Commencement June 2022

Master of Science thesis of Shahad Almasari presented on December 10, 2021.

APPROVED:

Major Professor, representing Computer Science

Head of the School of Electrical Engineering and Computer Science

Dean of the Graduate School

Shahad Almasari, Author

ACKNOWLEDGEMENTS

First of all, I would like to thank Allah for all his graces, and without his mercy, this accomplishment would not have been possible. I am also thankful to my advisor Professor Bechir Hamdaoui for his support, patience, and guidance. Finally, I am also grateful to my committee members: Professor Lizhong Chen, Professor Bella Bose, and Professor William Warnes, for accepting to serve as the committee members of my defense.

I want to thank my dear husband for his love, support, and continuous encouragement during this journey. Finally, I will be ever grateful for my parents' assistance, and I am sorry that they have not lived to see me graduate.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF FIGURES (Continued)

LIST OF TABLES

LIST OF ALGORITHMS

Chapter 1: Introduction

Electric vehicles are ubiquitous. Many of these vehicles come with self-driving abilities, and as more appear on roadways, the need for the vehicles to communicate with each other to navigate the roads safely increases. This communication happens with the help of vehicular networks. Vehicular networks require low-latency and high reliability to communicate successfully. To meet these requirements, researchers organized vehicular ad-hoc networks (VANET) to provide services related to road safety over the past several decades. Nowadays, the automotive industry has improved by integrating software and hardware components to provide a new level of connectivity in vehicular networks. Intelligent vehicular networks combine the early advancements made to create a new generation that supports the quality of services and the reliability of communication that both today's consumers and urban settings require. As a result, intelligent vehicles successfully operate on the world's roadways as part of smart city infrastructure [40, 41, 9]. Furthermore, researchers have focused on improving vehicular networks with advanced wireless networks and Artificial Intelligence (AI) technology [54].

Well established networks are getting much research attention because of their ability to provide services efficiently. SDN infrastructure is a network infrastructure management method that enables dynamic and efficient network configuration to improve network performance by distributing control and data planes. Thus, SDN and advanced wireless networks in vehicular networks are solutions to manage and improve the quality of service while satisfying performance requirements such as ultra-low latency and high reliability. Moreover, this technology is applied to overcome these requirements by having a programmable and flexible network infrastructure that includes differently integrated technologies, such as network functions virtualization (NFV) and network slicing [21, 5].

There are many challenging issues in vehicular networks, one of which is extensive traffic on the roadways which requires management to protect human life. Considerable progress in the research is needed to make autonomous vehicles work smoothly in vehicular systems with heavy traffic [48]. This study proposes a method for managing traffic prediction and for optimizing the traffic avoidance system at the top of a well-established network infrastructure. As a result, vehicles send their next lane and route data to one or more central controllers in SDN infrastructure to collect, analyze, and direct vehicles to alternative paths to avoid traffic.

Communication is another challenge and a very critical factor affecting network performance. The amount of delivered data to the servers and controllers, for example, dramatically affects the response time and reliability of vehicular networks. The Internet of Vehicles (IoV) will benefit from significantly minimizing latency and energy usage. Thus, with recent advances in artificial intelligence, vehicles can get many services faster than usual by predicting the decisions to manage the network communication and accomplish services successfully. This latency is minimized by offloading the computationally intensive tasks from mobile devices to a nearby Mobile Edge Cloud (MEC) [48, 6]. As a result, we have applied the DNN technique in high-density vehicular environments called Deep Reinforcement Learning (DRL) to distribute the processing and prediction process at the top of the traffic avoidance algorithm in the control plane. DRL is used to study the environmental inputs and produce associated outputs, which are rerouting by changing directions or increased speed to reach the destination faster.

In real-life scenarios, physical barriers and limited wireless ranges are other challenges that could prevent all nodes from steadily communicating with others in real time. Examples include rapidly changing architecture, accidents, disconnects, delays, and harsh communication environments like natural disasters and inclement weather incidents [80, 41]. Hence, the system needs a protocol that could manage the buffers and data storage criteria to keep critical data instead of losing it. DTN protocol is used to preserve the data and manage the buffer.

The number of self-driving vehicles that need to be connected to wireless access networks will increase as a result of the growth of the Autonomous Vehicles (AVs) industry. Therefore, our goal in this thesis is to consider the learned policy in vehicular environment properties to preserve stability and security before autonomous vehicles travel on our highways and city roads. Moreover, we need to study the real-life vehicular environment inputs, such as the number of vehicles and vehicle occupancy limits to ensure smooth movement throughout streets to reach the destination faster. Nevertheless, the biggest obstacles and challenges are connectivity and network latency [70]. Some studies have developed algorithms by using the queuing system to organize response priorities to overcome latency challenges. Many studies have used network slicing and implemented algorithms to speed up processing based on latency constraints, but they have used high numbers of controllers to manage the vehicular infrastructure. Most of the previous work relies heavily on simplifications, such as assuming unrealistic properties of real-world networks such as modeling roads using the Manhattan Poisson Line Process (MPLP) distribution or probabilistic routing. Therefore, there were inaccuracies when applied to large networks which did not predict complex situations with realistic routing configurations. These inaccuracies resulted in increased latency and reduced response time that could lower performance metrics [21]. Our work has enhanced this previous work by improving data processing and offloading the analysis between BSs and Fog controllers, using DNN instead of simply increasing the Fog controllers' numbers and consuming them. Moreover, our work has studied emergent scenarios such as disconnections that could happen in real life. Therefore, we addressed the low latency and reliability requirements in an intelligent way without consuming more controllers.

In this thesis, we provide a system architecture based on SDN technology by using the latest mobile networks based on 4G capabilities. This system allows us to improve transmission efficiency and meet low latency constraints. Moreover, we developed prediction models using Deep Reinforcement Learning (DRL) to distribute the processing and prediction between data plane and controller plane infrastructure. Advantages to our system include the fact that we can reduce centralization and have the ability to deal with big data analytics by offloading processing for a large number of vehicles. Also, our system can also address the problems of harsh communication environments and it has incorporated the ability to keep data without losing it. Thus, our approach uses DTN protocol to improve the quality of autonomous driving and ensure its continuity.

The rest of this thesis is as follows. In section two, we present relevant works. The third section explains the SDN design and how mobile networks were used. The fourth section presents the DNN and its use in the network layers. In the fifth section, we describe DTN and its application on the network. We present the simulation setup in the sixth section. The seventh section explains the results and performance analysis. Finally, the eighth section presents the future work and the conclusion.

The following table describes the significance of various abbreviations and acronyms used throughout the thesis:

Acronym	Meaning
VANET	Vehicular ad-hoc networks
AI	Artificial Intelligence
SDN	Software Defined Networking
NFV	Network functions virtualization
IoV	Internet of Vehicles
MEC	Mobile Edge Cloud
DNN	Deep Neural Network
DRL	Deep Reinforcement Learning
DTN	Delay-Tolerant Networking protocol
AVs	Autonomous Vehicles
MPLP	Manhattan Poisson Line Process
BS	Base Station
RSU	Road Side Unit
V2V	Vehicle to Vehicle
V2I	Vehicle to Infrastructure
IoT	Internet of Things
V2X	Vehicle to Everything
ITS	Intelligent Transport System
OMNET++	Network Simulator
Venis	Vehicles in Network Simulator
SUMO	Road Traffic Simulator
TraCI	Trac Control Interface

Chapter 2: Literature Review

This section examines previous studies and discusses vehicular network challenges, and how technologies are used to overcome those challenges. It also shows the related work of the technologies, which are SDN, DNN, and DTN protocol.

In SDN, Chekired et al.[21] studied the performance of vehicular networks with ultra-low latency and high reliability. They deployed new emerging technologies such as SDN, NFV, and network slicing using architecture called slicescale. In addition, they used a distributed SDN core network architecture with Fog, edge, and cloud computing nodes in a 5G wireless network. Also, they split the physical network infrastructure into multiple logical networks functions which are controlled and managed separately by the slice's owners to satisfy the low latency challenge. Finally, they used an algorithm to manipulate the resources and to provide the vehicle missions depending on time preferences. Based on their proposed algorithm, the resource blocks of each Fog cell were scheduled when it arrived at the associated Base Station (BS) and checked whether the number of local resource blocks satisfied the latency requirement. The simulation results show that they met the low-latency requirement of the autonomous system; however, they did not study the content of the delivered data in real situations, and the amount of delivered data in their system was enormous, particularly in the upper layer.

In DNN, [46] considered a wireless powered MEC network that utilizes an offloading policy so that each processing task of wireless devices can be executed locally or offloaded to a MEC server. They used an algorithm to offload the decisions and resource allocations between servers based on wireless channel conditions. Because of channel coherence time and its requirements, which are a quick processing and response that are hard to achieve with traditional methods, they used DRL based on the offloading framework to learn the offloading decisions. They successfully decreased the computation time. However, they only considered channel information as DNN input. Vita et al. [28, 29] studied the MEC to bring data and computational resources close to mobile users that reduce latency and improve resource utilization. They used DRL to relocate applications in

MEC scenarios and to manage data migration by learning during system development. Furthermore, they used 5G technology to improve the network performance and to allow mobile users to access distributed services efficiently. They showed the feasibility of this approach for the latency challenge via simulation. Nevertheless, they presented the continuity of the services throughout the 5G wireless network without considering urgent scenarios. Also, they did not study different scenarios of stochastic environments that include a large number of users and lower levels of wireless technology such as WIFI or 4G. Moreover, vehicular environments could have urgent situations such as limitations in wireless communication due to physical barriers, so it needs alternative solutions on top of DRL to provide the services in lower latency. [57, 58] proposed an approach based on swarm intelligence to evaluate the traffic flow during congestion to avoid collision using Vehicle to Vehicle (V2V) and Vehicle to Infrastructure (V2I) communications. They considered the details of vehicular environment inputs to exchange and collect information about vehicles and roads. They studied the road experience to reduce road congestion and improve traffic management to ensure road safety. Their system requirements achieved fast and efficient communication between smart devices. The results showed improvement in the performance to maintain the traffic flow. Also, they improved the work by using DRL on smart rerouting, but they did not use robust networks such as SDN infrastructure. Thus, V2I in SDN managed traffic flow even with a large number of mobile nodes as they get the services faster from the servers to cover the information from the whole coverage area.

In DTN, [45, 56] studied the unreliability issue in mobile networks to improve the delivery rate in a stochastic environment. They found that Internet of Things (IoT) applications need to achieve efficient data transmission using the Delay-Tolerant Network (DTN) to enhance delivery that ensures high throughput and low latency requirements. Furthermore, comparing it with other routing protocols demonstrated that the DTN can perform better than the existing protocols by managing the buffer between nodes in urgent scenarios such as losing the communications.

In this work, we used the network structure SDN in [21]. We improved their architecture by making it more intelligent by using DNN instead of a specific algorithm that depends on the queuing system. Thus, we were able to achieve ultra-low latency results even with WIFI and 4G wireless networks issues. Also, we considered the disconnection scenarios, and we used the DTN protocol to make the system more reliable overall. This work is more specific because it depended on environmental and vehicle information, thus implementing complex policies to avoid vehicular traffic. The idea of setting up the simulation environment came from [29] that used the OMNeT++/SimuLTE simulator integrated with the Keras framework. It helped us to study the RL technique and observe the dynamics of an environment, thus understanding an optimal policy for one or more environmental situations. Moreover, we studied the traffic at each point of the vehicle's coverage area to avoid extensive traffic, so we controlled the traffic in a high mobility environment. Thus, based on this literature review, we used these integrated technologies on top of the proposed traffic avoidance algorithm.

Chapter 3: System Technologies

Recent advances in wireless communication and networking technologies and techniques emerge as key enablers to vehicular communication and autonomous vehicular systems. These include advancements in wireless bandwidth management [37, 13, 39, 69, 38, 33], spectrum resource sensing and discovery [35, 36, 34, 79, 81, 51, 50], in-network content caching and placement [68, 78, 72, 53, 67], edge cloud computing and offloading [2, 71, 3, 24, 76, 1, 7, 4], software defined networking (SDN) and network function virtualization [25, 17, 5, 47], cloud datacenter networking and resource management [11, 60, 62, 26, 10, 64, 22, 23, 27], AI-enabled networking and communication [75, 44, 31, 32, 15, 43, 59, 74], among many others. Nowadays, vehicles can communicate with one another (aka V2V) and with everything (aka V2X) including infrastructure. They can communicate without the need for complex protocols and algorithms that enhance the ability to determine each vehicle's emergency signals based on their positions [66].

In the new generations of cellular networks such as 4G and 5G [12, 20, 42], the download and upload rates are increasing rapidly. As a consequence, the amount of data communicated is enormous. This data needs robust infrastructure for vehicular networks with rapid speeds [21]. Thus, the robust infrastructure distributes the analysis of the collected data in a way such that the data has low latency in processing and controlling.

VANETs, a type of wireless system, are used to share traffic information between vehicles to avoid accidents. However, the industry faces many challenges because of a dynamic topology and high mobility environments, such as sharing the information among vehicles and infrastructure nodes. Moreover, the processing capacity of this network will be much higher because it communicates with services in the cloud. To address these problems, IoV is used between wireless systems, which are VANETs, Wi-Fi, 4G, and other advanced wireless networks. IoV is a scalable structure that splits and enlarges services across diverse networks. Therefore, it brings intelligence to the network because the information is shared and distributed between layers [66, 21].

This work presents the used layers in IoV which share the information among the vehicles and achieve the mission in real-time using the following integrated technologies:

1. SDN manages and coordinates the infrastructure of the environment. In SDN, we study how this data is collected, exchanged, and offloaded between data and control layers.
2. DNN focuses on realizing and analyzing the delivered information to and from BSs to offload the processing with fog controllers on vehicles' intelligence routing procedures.
3. DTN protocol manages the buffer to improve the intelligence network feeding in disconnection scenarios.

3.1 Software Defined Networking (SDN)

Vehicular networks need to be intelligent, centrally controlled, or programmed using software applications. Software-Defined Networking (SDN) is a network architecture strategy that allows us to meet these needs. SDN helps operators to manage the entire infrastructure consistently and holistically, regardless of the underlying network technology. In addition, SDN operators can manage the infrastructure with several competing forces surrounding enterprises, carriers, and service providers. Moreover, the extensive growth in multimedia content, the influence of increasing mobile usage, and the explosion of cloud computing tends to wreak havoc on traditional models. Therefore, many systems are turning to SDN technology to reform network design and operations regardless of the complexity of the underlying network technology.

SDN was created by separating the control and data levels used in the public telephone network to simplify management in data networks. SDN allows us to control the programming of network functions using open APIs. The API for OpenFlow was first created in 2008 to build the network operating system. OpenFlow allows communication between the control plane and the network infrastructure [21].

My work is based on the SDN infrastructure, and one of its benefits is its ability to be developed in consecutive steps. This network is the base point in controlling the vehicular environment, re-routing, and extracting the base infrastructure from applications and network services.

There are many advantages to SDN architectures. Several of these are listed as follows:

- Their architectures are directly programmable.
- They reduce delay in computing services by having data and control layers.
- They can dynamically adjust traffic flow at the network level to meet changing needs.
- They have a network that is managed centrally through specific controllers.
- They allow network administrators to rapidly identify, manage, secure, and optimize network resources.

We have designed the nodes and communication protocol in different layers, which can determine the path of the network packets and manage them in the most efficient way. We did not rely on OpenFlow because we do not need most of its services. In addition, the simulation system version is a newer version compared to OpenFlow available versions. Therefore, it was hard to merge OpenFlow with the system, so we have designed a communication protocol between the control level and the network infrastructure as clarified in the following network design, network layers, and messaging sections. As a result, a large part of the searching and linking strategies were shortened.

3.1.1 Network Design

Vehicles have more than one way to connect wirelessly, so our network design uses two different wireless networks, which are WIFI and 4G. Also, the system identifies the communication between layers of SDN in these different types of wireless communication. Figure 3.1 clarifies the design of the network infrastructure in all scenarios. It includes number of nodes, which are vehicles, RSUs, BSs, and Fog.
Figure 3.1 shows the following types of communications between nodes:

- V2V communication, such as the communication between vehicles to share buffer storage information.

- V2I communication, such as sending the inputs data from vehicles to the infrastructure.
- Infrastructure to Infrastructure (I2I) communication, such as the communication between the BSs to share the resources in the shared coverage area or the communication between the BS and the Fog to update the topology information.

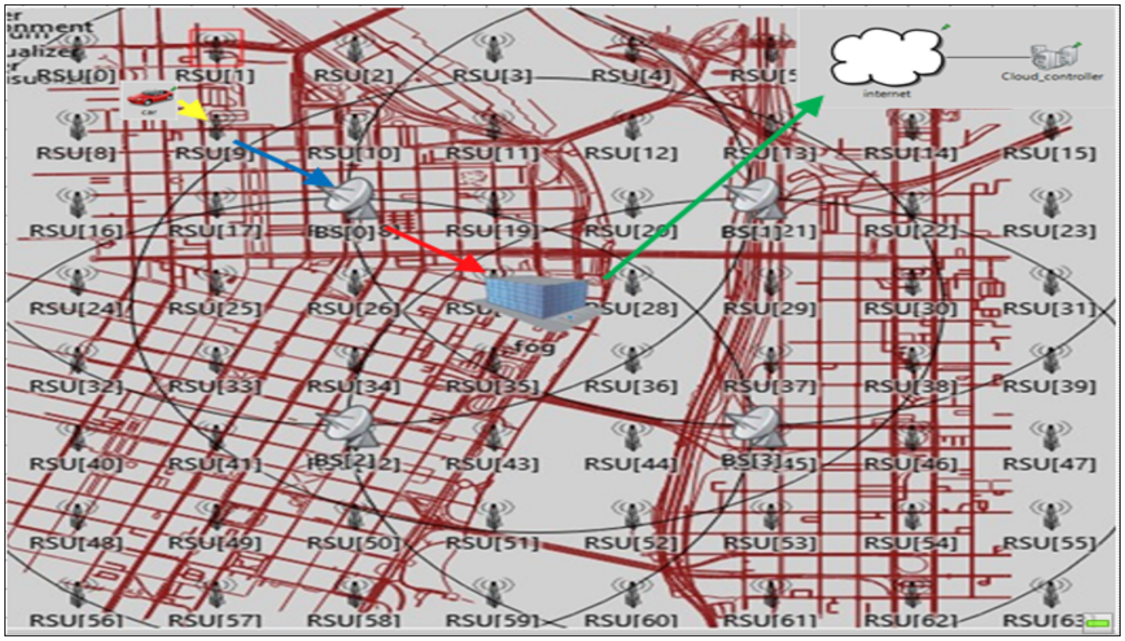


Figure 3.1: The Structure in Different Scenarios

3.1.1.1 WiFi Scenario

In this scenario, RSUs are distributed over the entire area to cover the map equally to communicate with the vehicles, whether receiving data or sending routing commands.

Communication with BS units uses the WIFI method by communicating with the RSUs reaching the BSs. Using a WIFI wireless network is good if the system can not deal with the expensive or unprepared type of advanced wireless network. This scenario has several characteristics, such as simplicity, and it distributes offloading mechanisms that are supportive because of SDN architecture. The WIFI system works as an alternative to the 4G system. Thus, it maintains service levels when the advanced network service is interrupted.

3.1.1.2 4G Scenario

The significant feature of 4G is short-range wireless communications that support the novelty of V2I and I2I communications. Considering the IoV requirement of using intelligence, we used 4G to establish communication directly between the vehicle and BSs. Also, we used 4G to communicate between BSs and Fog.

4G is suitable and reliable to be used type for IoV communications through a distributed node. In addition, its solution enhances vehicular network communication delays and overhead by reducing the need for multi-hop routing. Preliminary simulation results show that it provides short setup times and improves vehicular communication by reducing delays.

3.1.2 Network Layers

The system layers are shown in Figure 3.2 which includes data and control planes.

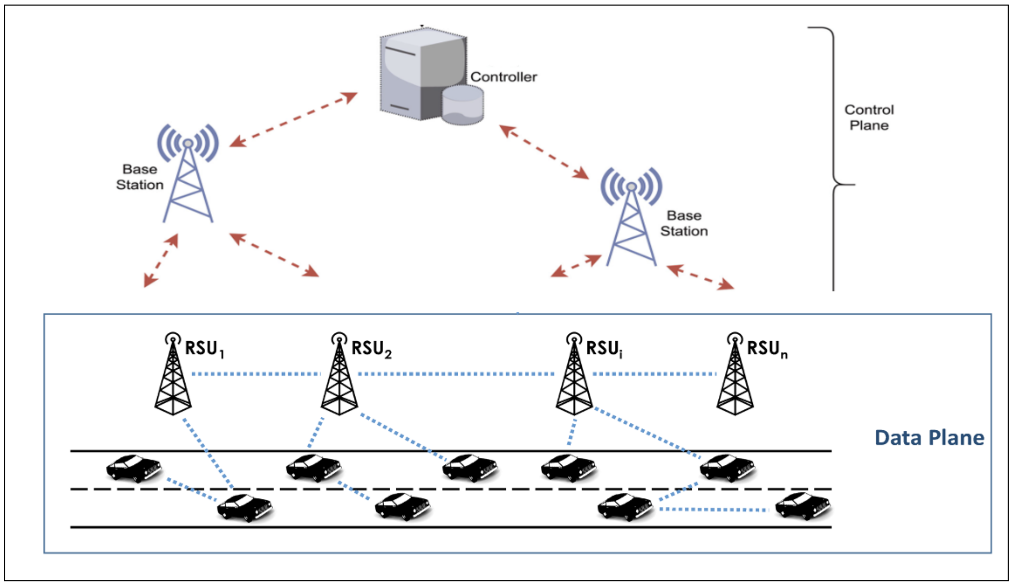


Figure 3.2: Network Structure Layers.

3.1.2.1 Data Layer

This layer involves the vehicles and the Road Side Units (RSUs) in which the data is being transmitted throughout the logical area. The primary use of this layer is collecting the data and re-sending it to the control layer. First, each vehicle will have a trip that includes initial and final positions. It is responsible for localization, planning the route based on the instructions from the upper layer, and managing the driving mode to accelerate, decelerate, or park. Second, RSU will maintain network coverage, so vehicles can send their data to the closest RSU. Also, RSUs combine GPS and sensors to determine their locations recorded as data. Thus, depending on traffic rules and given map data, vehicles determine the allowed speed in the current location. Moreover, based on the collected data, vehicles choose the driving mode and the functions that are used for driving. The vehicle and the RSU communication is called Vehicle to RSU (V2R) and is done using WIFI wireless networks.

3.1.2.2 Control Layer

Control Layer is divided into two different levels:

- Aggregation:
  This level involves the duty of the BS. It collects data from the RSUs in the WIFI scenario or collects data from vehicles in the 4G scenario. After collection, primary processing is done on the collected data at this level. It aggregates the data based on geographical area positions to avoid traffic. Therefore, it helps to reduce the Fog processing load, as the work is divided between Fog and the BS units [21]. Moreover, BS is responsible for monitoring the number of occupied resource blocks at the current coverage area to virtualize the unoccupied resources to the adjacent BS.

- Forecasting and Control:
  The Fog's main duty is completing the work done by the BS in the previous level to collect the data of the intersecting areas between them. Then, it analyzes the data and extracts the places of expected congestion. After that, it makes automated decisions about the traffic congestion [21]. Next, it determines a set of commands to the units at specific area, which are the decision for cars to redirect them and changing their associate speeds. Then, Fog sends these commands to the BS units. Finally, BSs recognize vehicle location and send instructions to the RSUs responsible for the commands in a WIFI scenario or they command vehicles directly in the 4G scenario. The Fog contributes by distributing the load and decreasing the overhead of large services. Thus, the work is done through regular direct guidance for all vehicles to alternative or leading routes.

3.1.2.3 Cloud Layer

This layer first performs the process of saving data in public databases used in the final analysis of the system. It is responsible for building global network connectivity by aggregating information received from the lower layers. Hence, it tracks work and analyzes its performance. This layer also works on available application services that provide many services to the Fog through API applications that help in many tasks.

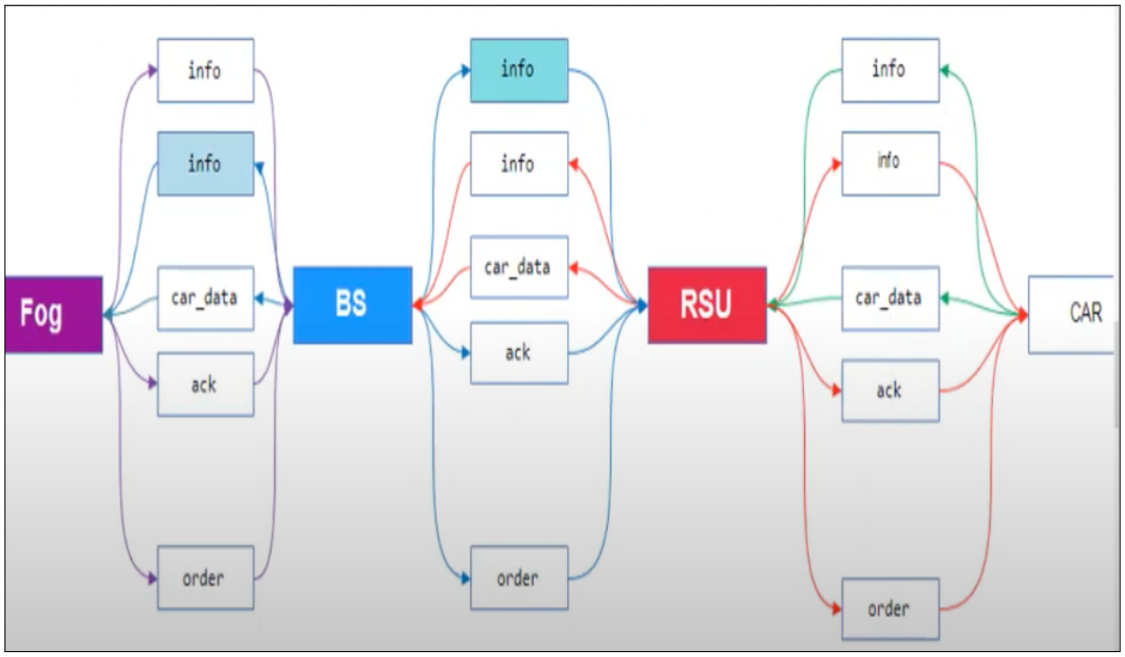


Figure 3.3: Messaging Stages

3.1.3 Messaging

Messaging identifies the nodes to each other and designs the communication between data and control levels. It also adds or replaces any unit easily to avoid any error as each layer is working separately. Also, it provides information about traffic jams based on vehicle location to avoid accidents and long trip time. Thus, vehicles can change routes and speeds when they are provided almost real-time information [21, 66]. The work between units goes in the following messaging stages as shown in Figure 3.3:

1. Definition Stage:
   It is the first gate when anyone enters the system. At first, it starts with the highest unit, which is the Fog. It sends an introduction message that includes its data, which is Information (Info). Upon reception, all BS units respond to it with Info messages that have their data, such as ID and position after registering the data of the Fog unit. When the Fog receives these messages, it records the BSs data. At the same time, the RSUs get an Info message, so they select and register

with the nearest BS unit and send an Info message to it. The BS receives this message and records the data of its RSUs and in this stage, the system nodes have known each other, communicated, and registered. When registering a new unit of any kind, it sends an Info message, then the infrastructure units record it, and this process occurs when the system starts, stops, or restarts. Thus, this stage is repeated periodically when there is a new added node.

The introduction between vehicles and RSUs is a systematic process where the vehicle sends message Info at a fixed time. When received for the first time by the RSU, it replies to the message, clarifies its data, and records it. In addition, when the vehicle gets the message that bears the same Info message number, it will be with the same RSU; otherwise, it will find the closest RSU.

2. Data Stage:
   After the vehicle receives the message from the RSU, it records it. Then, after the specified period for sending, the vehicle sends data (Car-data) to the last associate RSU, which includes its data such as its ID, speed, location, destination, a group of upcoming route points. Next, the RSU resends this data to the BS unit without any delay or modification in the message. After that, when the BS receives the message, it adds the data within a real-timing table which includes all of its vehicles' data. Then after a specified period, it analyzes the data and collects it in packets that include the specific area of this BS. In other words, it calculates the vehicle speed that depends on permissible street speed, and it collects the vehicles and their directions in the local area numbers. Finally, it resends it to the Fog unit to extract the congestion data in the future following routes in chronological order.

3. Command Stage:
   The Fog receives vehicle messages from BSs after each BS collects data from all units in one table, processes, and assembles in the form of packets. The Fog records them in another table. Then, it analyzes and makes decisions as a message with the area number. The decision message (order) is to request the rerouting or change the speeds. After that, the outcome is either to direct the vehicles to alternative routes or to to direct them to change their speeds based on the specified percentage of vehicles of the maximum occupancy vehicle number within the range. Finally,

with the RSU or the BS from these areas, they direct orders to vehicles.

4. Preservation and Evaluation Stage:
   After the Fog sends operating commands to the units, it sends all the packets with the decisions specified to the cloud, where it saves them. In addition to the performance analysis data that the Fog creates to the cloud on a regular and separate basis, the Fog's decisions can be evaluated and modified for future forecasting.

3.1.4 Used Algorithm for Vehicles Directions Management

In cities, traffic jams are widespread. In this thesis, we have used a routing technique to avoid extensive traffic. This technique is used to reroute the vehicles that are not yet reached the crowded points of the route but coming toward the points. This algorithm addresses this issue to enhance the delay of vehicles' trips.

When a new vehicle joins the current road based on its route, and the following points are crowded, Simulation of Urban Mobility (SUMO) gives shortest alternative routes from the map that reach the same destination point. It has characteristics of the environment and its topology. Additionally, it can provide different routes for a specific destination and help Fog with a table of routes based on positions and destinations. On the other hand, Vehicles in Network Simulation (Veins) measures the costs of the roads based on the shortest path to the following route points that have lower traveling times. Then, the upcoming steps are rerouted to the alternative routes that have lower costs [57, 58].

Algorithm 1 will be done periodically, and it shows the procedure as steps to avoid extensive traffic in a vehicular environment. Thus, this algorithm observes the most basic information about traffic on any road in order to calculate different parameters such as threshold (the distance of the coverage area after calculating the spacing between lanes) and vehicles' occupancy (vehicles' number limit that can locate in the current area) as shown in equation 3.1. The processes in lines 1-10 are to set the traffic parameters. In 11-21 lines, we measure the density and rerouting technique to avoid extensive traffic. Finally, in the last line, the system refreshes the number of vehicles and the threshold periodically to monitor the density of vehicles in real-time.

Algorithm 1 Vehicles Directions Management

Input : Position, Speed, Destination, Next route points

Output: Changing speeds, Changing next route points

```
1  Fog stores routes file from SUMO that includes routes and alternative routes of
   each position based on source and destination
2  Vehicles send the inputs, and their identification to count them
3  RSUs and BSs resend the data to the Fog
4  Fog receives the inputs of each vehicle that are associated with different RSUs
5  Group vehicles that have identical following points of routes
6  Obtain the dimension of each vehicle and calculate the average size of vehicles
7  Compare the vehicles' average size with coverage area distance based on the
   position to set the threshold
8  Set the vehicles' occupancy limit of the coverage area
       $vehicles'occupancy = \frac{Threshold}{vehiclesAverageSize}$     (3.1)
9  Get vehicles' occupancy and divide it by 3 to get the maximum number of
   vehicles allowed in the current distance
10 Locate the points of traffic that could have a larger number of vehicles than the
   occupancy limit of the area
```

```
    while The following route points are in the area that has higher number of vehicles than vehicles' occupancy limit do
11      Calculate the difference between the vehicle's number and the vehicles' occupancy limit.
12      if difference <30% then
            Change the fourth of vehicles' speed
        if difference is between 30% and 60% then
13          Veins will calculate the eligibility to access the alternative route by setting the cost based on travel time of each route and the shortest path
14          Eliminate the alternative routes by choosing the lowest route cost
15          The infrastructure chooses the next route
16          Direct the half of the vehicles to the alternative route
        if difference >60% then
17          Do from 13-15
18          Direct the two thirds of the vehicles to the alternative route
19      Updated vehicles' routes table
20      Add the vehicles to the alternative position
21  Update the threshold and the occupancy
```

Figure 3.4 shows the the flowchart of Algorithm 1.

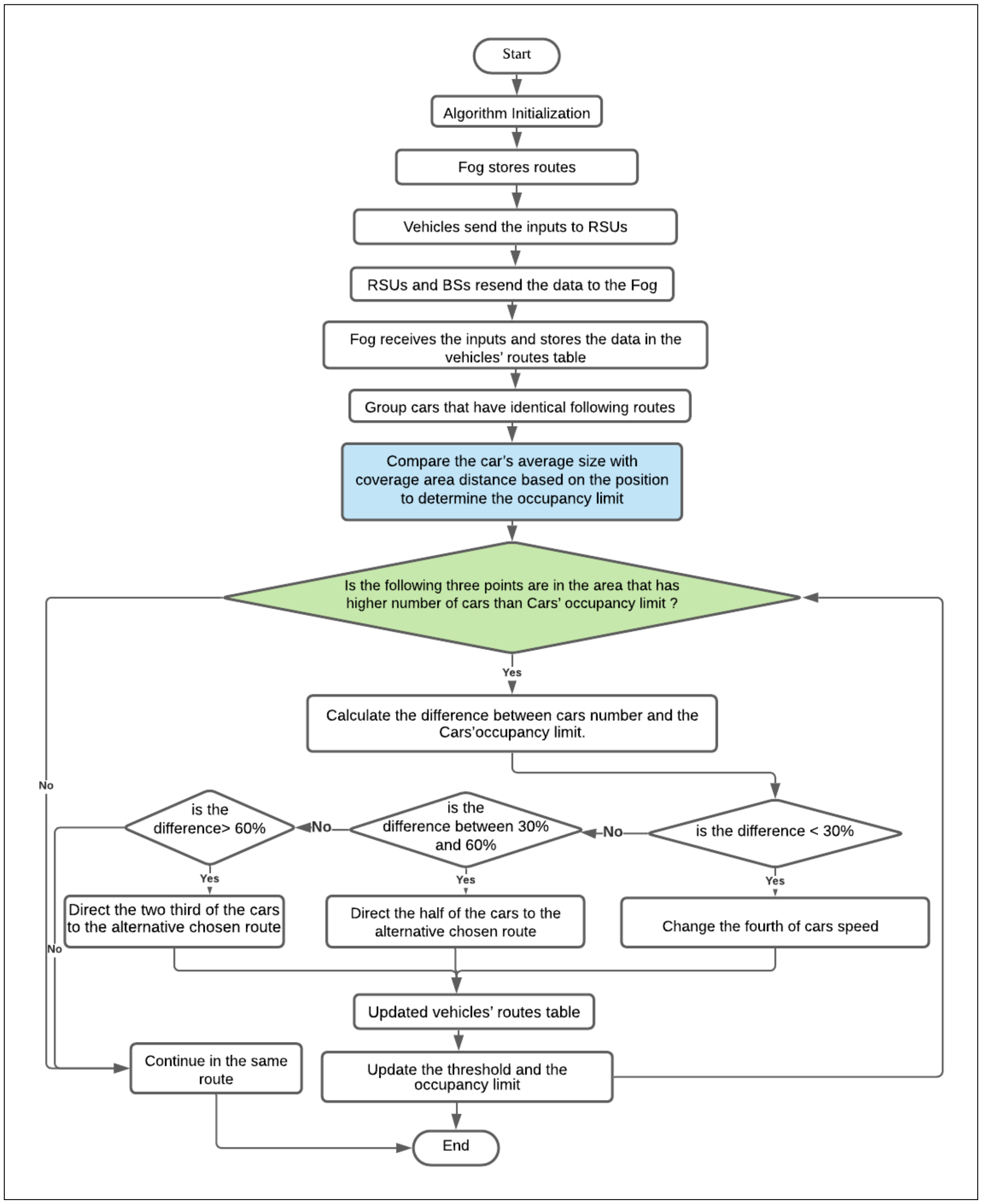

Start
Algorithm Initialization
Fog stores routes
Vehicles send the inputs to RSUs
RSUs and BSs resend the data to the Fog
Fog receives the inputs and stores the data in the vehicles' routes table
Group cars that have identical following routes
Compare the car's average size with coverage area distance based on the position to determine the occupancy limit
Is the following three points are in the area that has higher number of cars than Cars' occupancy limit ?
Yes
No
Calculate the difference between cars number and the Cars'occupancy limit.
is the difference < 30%
No
is the difference between 30% and 60%
No
is the difference> 60%
Yes
Yes
Yes
No
Direct the two third of the cars to the alternative chosen route
Direct the half of the cars to the alternative chosen route
Change the fourth of cars speed
Updated vehicles' routes table
Continue in the same route
Update the threshold and the occupancy limit
End

Figure 3.4: Vehicles Directions Algorithm

3.2 Deep Neural Network (DNN)

The best methods for the development of electric vehicles services and applications have not existed yet. Nowadays, 5G and WAVE only have solved part of the whole set of requirements for vehicular environments. The motivation to solve them will have come when these networks start to be extensively used in everyday life, particularly IoV. Vehicular networks have reached a point where the cloud paradigm can no longer ensure low latency, especially when the distance between nodes is too high. In that context, where network infrastructure has become more complicated, machine learning and DNN can be helpful as a process to deal with environmental forces by finding the right strategy to improve the whole system performance, such as traffic delivery in real-time awareness. Recent works have intended to use SDN to address some challenges of vehicular networks. For example, environment data and actions can be stored to save more lives by predicting traffic congestion to avoid accidents. AI can be used for many reasons in vehicular environments ranging from network structure to driver benefits. For example, some systems have used large data services, so it is necessary to distribute the computing throughout the network nodes. Thus, DNN is used as the best method today to offload the computational services to protect the lower network layer from slow responses and poor QoS [66, 28].

Deep neural networks are ML algorithms developed to make decisions according to learned actions. DNN-based on the Vehicle-to-Everything (V2X) system gathered the input information from many sources, such as vehicles. DNN system increased the realization and forecasting deeply to avoid traffic. This improvement has the opportunity to understand smart driving from a large number of states and avoid human mistakes to bring safety to drivers [8].

In this work, we have focused on building an autonomous AI-enabled algorithm that can understand the system's state and appropriately transfer vehicles requests data with the ultimate goal of improving the Quality of Service (QOS).

In particular, we are interested in developing DNN based on enhanced Reinforcement Learning (RL), which is DRL.

3.2.1 Deep Reinforcement Learning (DRL)

RL is a machine learning technique used to recognize the dynamics of an environment. Thus, learning an optimal policy concerning one or more performance indexes. RL is the only feasible solution to correctly train a system in many contexts, especially if it is impossible to work with labeled data in a stochastic environment. The learning is through a trial and error process, so it is almost like the human learning one that makes the best choice to solve decision-making problems. Therefore, RL uses a basic model, which is the MDP formalism, to model the decision-making. It includes [55]:

- State: s is the state that defines the current situation (e.g., the point of the traffic in the street).
- Action: a is the choice that the car makes at the current time step (e.g., it can change the speed or find an alternative route throughout the street, etc.).
- Probability transition matrix
- Reward function: R(s) $s{:}\rightarrow R$

  (e.g., It is the outcome of these actions. If the BS can direct the car to an alternative route that reaches the destination faster, we can call it a positive reward. While getting stuck in the chosen route will be called a negative reward)
- Discount factor: $\gamma \in$ [0,1], which defines the importance of future rewards.

The objective of RL is to find an optimal policy that maximizes the reward while performing actions in the environment. The policy will be the thought process behind picking actions. In other words, it is a probability distribution assigned to the set of actions. For example, highly rewarding actions will have a high probability and vice versa. Thus, the agent will repeatedly traverse the environment states and change the system policy to maximize the reward. When the probability transition matrix is unknown, The RL approach is Q-learning. It is a model-free technique that learns the relationship between actions in a given state and their associated rewards by updating a Q-value to have the optimal reward in the state.

$$\underbrace{\text{New } Q(s,a)}_{\substack{\text{New}\\\text{Q-Value}}} = \underbrace{Q(s,a)}_{\substack{\text{Current}\\\text{Q-Value}}} + \underset{\substack{|\\\text{Learning}\\\text{rate}}}{\alpha} \Big[\underbrace{R(s,a)}_{\text{Reward}} + \underset{\substack{|\\\text{Discount}\\\text{rate}}}{\gamma} \overbrace{\max Q'(s',a')}^{\substack{\text{Maximum predicted reward, given}\\\text{new state and all possible actions}}} - Q(s,a) \Big]$$

Where α is the learning rate, and γ is the discount factor. In our situation, the number of states describing an environment is too large, and it is not likely to use traditional methods like RL alone and its corresponding MDP. Therefore, DRL is the best solution to this problem instead of using a traditional method, especially in a stochastic vehicular environment. It is used as a useful tool to improve the learning rate for RL algorithms. Furthermore, the obtained experiences will be saved and used to train the neural network throughout the real-time learning process. The latter will support the agent in making optimal decisions in real-time. Thus, the neural network in the DRL will be trained, and we will have new experiences obtained through real-time interactions with surrounding environments [29, 77].

3.2.1.1 DRL and Problem Formulation

DRL is an advanced model of the RL technique introduced by DeepMind [49]. DL over RL is used as a powerful tool to increase the learning rate for RL algorithms. This work tends to use the DRL approach to form an optimal migration policy to improve user QoS.

DRL will be used to predict the Q-values of a state. The idea behind this is to provide the main and target DNN, so the main will predict Q-values associated with states and the target will be used to produce the target Q-values needed to train the Main one. Hence, the obtained result will be more stable, which is not affected by learning loops because of the self updated network targets that can lead to oscillations [28].
The gathered data inputs starting from vehicles; then, RSUs and BSs, are clarified below.

1. Position.

2. Speed.
3. Destination.
4. The following route points.

Then, the data will be forwarded to RSUs and BSs that can be used as grouped information based on associate coverage areas such as the number of cars.

The system can understand the environment from the knowledge of system parameters. Then, it can learn a complex policy that is very difficult to deploy by using heuristic methods.
The output is the direct vehicle commands to avoid extensive traffic, e.g., change directions and speeds, including brake and park. However, the system behaves like a black box because realizing the motion is complicated, it needs to follow all the driving duties [65].
In parallel, BS will learn to predict the following points of the path based on the streets states of the associated areas.

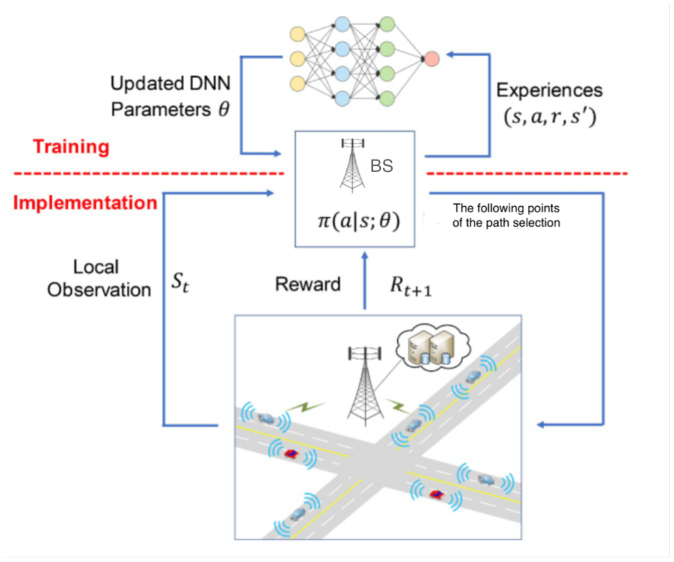


Figure 3.5: Infrastructure of Learning

Figure 3.5 shows the interaction with the environment in multiple looped steps.

1. The system checks the initial observation s.
2. The infrastructure will take decisions based on this observation for action a for all vehicles of the associate geographical area.
3. The data plane performs the data transmission.
4. The information is exchanged between edge nodes to cover the topology and predict the following vehicle's movement.
5. Rewards evaluate the actions.
6. Having new observations from new environmental situations is the last part of each step, and it is the basis for the next step.

The learning includes two phases. The first phase is the training phase for a particular area that includes the number of states. The second phase is the operational phase when the training is completed but still has the ability for minor optimization [52, 29].

The path selection is constructed based on the Djikstra algorithm. It is computed by specifying the shortest path between the position of the vehicle and its destination. Moreover, the routing path is not stable, so the system has measured the costs of the roads based on the threshold and the maximum vehicles number limit to guarantee a smooth trip that reaches the destination at a lower trip time. Then, the upcoming steps are rerouted to the alternative routes that have lower costs. As a result, the direction, current area, and velocity are affected to have lower traveling times. In DRL, the path selection as actions will be determined and offloaded between BSs and the Fog.

Table 3.1 and Table 3.2 show how we offloaded the processing and decision-making to avoid waiting for the response for the decision from the Fog controller. The Fog will be responsible for decisions of a more significant area level controlled by the number of BSs. Hence, applying the DNN on the BSs can reduce the effort to process collected data and assist in decision-making to accomplish lower latency.

	Data collections	Processing	Decisions making
Vehicles	✓		
RSUs	✓		
BSs	✓	✓	
Fog	✓	✓	✓

Table 3.1: The Functions before Applying DNN

	Data collections	Processing	Decisions making
Vehicles	✓		
RSUs	✓		
BSs	✓	✓	✓
Fog	✓	✓	✓

Table 3.2: The Functions after Applying DNN

The outcome of our system is a reduced amount of delivered data to the controller layer of the infrastructure because it will offload the type of data that needs to be delivered. For example, the BS will take all the gathered data from the data layer as the initial perspective in the first stage. Then, after learning, it will have efficient features instead of just having initial perspective information. In other words, the system will consume the channel efficiently, and it will process and decide the actions in lower latency by making the decision by the BS instead of waiting for the Fog. As a result, we will have data offloading of decision-making between Fog and BSs by using DRL. Moreover, reducing computing service delay and service failure penalty.

3.2.1.2 The used Algorithms in DNN

Algorithm 2 DRL

```
1  initialize the environment's experience
2  initialize the main DNN network in random weights
3  set the target DNN network equal to the main one
4  set the discount factor
5  set the batch size
6  set update step K
7  set the exploration rate $\epsilon$
8  for episode = 1 to end do
9      observe the current state $s_i$
10     $p = random([0,1])$
11     if $\epsilon > p$ : then
           $action = random([a_1, a_Z])$
       else
           $action = argmax(Q(si, \theta))$
       end
12     execute the action
13     observe the new state $s_{i+1}$
14     observe the reward r
15     store the t-uple $(s_i, action, s_{i+1}, r)$ in experience
16     sample a batch from the experience
17     $y = Q(s_i, \theta)$
18     $ytarget = Q(s_i + 1, \theta)$
19     $yaction = r + \gamma.max(ytarget)$
20     execute one training step on the main DNN network
21     every K steps set $\theta = \theta$
   end
```

Algorithm 2 shows deep reinforcement learning procedures. Line 1 is for setting up the initial experience. It plays a vital role as it stores all the necessary information for the DNN training. First, the neural network weights are set to the same values in lines 2-3.

then other DNN parameters are set in lines 4-7. At each for-loop iteration, the node observes the current state. Then, it selects the action depending on the exploration rate that establishes if it has to be chosen randomly from the action set to direct vehicles. On the other hand, it can be returned by the main DNN. Then, it observes the new state of the environment and the correspondent reward, as shown in lines 13,14. After that, the algorithm stores the new experience. In 17-20 lines, the main Q network will predict the Q-values for the given state s_i. Then, prediction and Q-value evaluation stages set the target DNN network using the Bellman update formula. Still, only the Q-value related to the action sampled from the batch will be updated. Finally, in line 20, train the main DNN network by executing one training step on the cost function, and if K steps are completed, the target DNN network weights will be set equal to the main DNN network ones [28, 29]. Figure 3.6 shows the the flowchart of Algorithm 2.

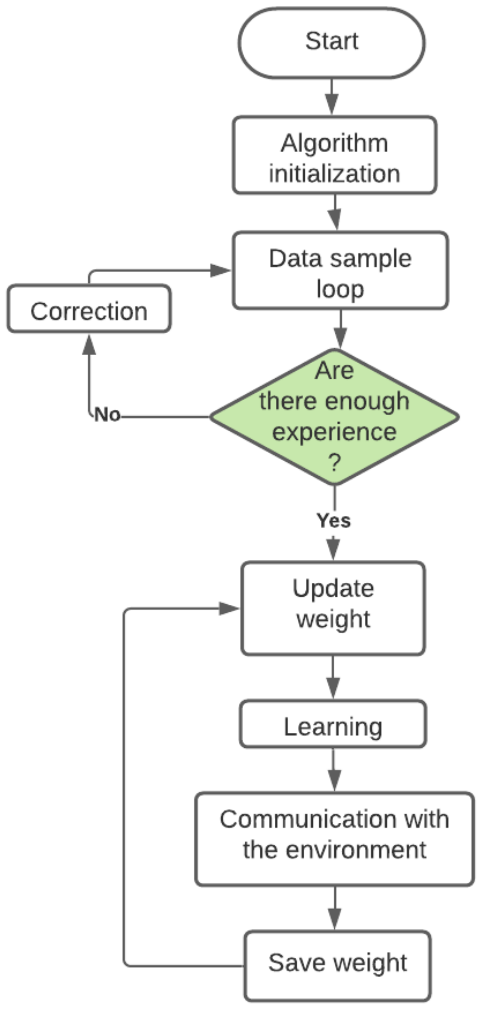


Figure 3.6: DRL Algorithm

Algorithm 3 Vehicles Directions Managements by using NN

```
Input  : Position, Speed, Destination, Next route points
Output: Changing speeds, Changing next route points
1  The vehicles send the inputs with their identifications.
2  RSUs resend the data to the BSs.
3  BSs save a copy before sending the data to the Fog.
4  BSs insert the data in the vehicles' routes table.
5  BSs put the next move as the target field based on the vehicles' information inputs.
6  Add the inputs in the process folder as a text file for processing.
7  Python code will work when it receives the updates as a text file and uses them as DRL input.
8  Get what has been done in Algorithm 1 from 10 to 21 to direct vehicles as actions
9  Do DRL Algorithm
10 if the experience is enough then
       The BSs update the data
   else
       The target has not been determined and get the instructions from the Fog
   end
```

Algorithm 3 is a looped algorithm that shows traffic management using DRL in clear stated steps. When the target is enough to update and execute the action by BSs, It can do the process without waiting for Fog for the routes instructions. The Python code will work in parallel with C++, and the updates between them are periodically updated using the text file.

The following flowchart shows the steps of Algorithm 3.

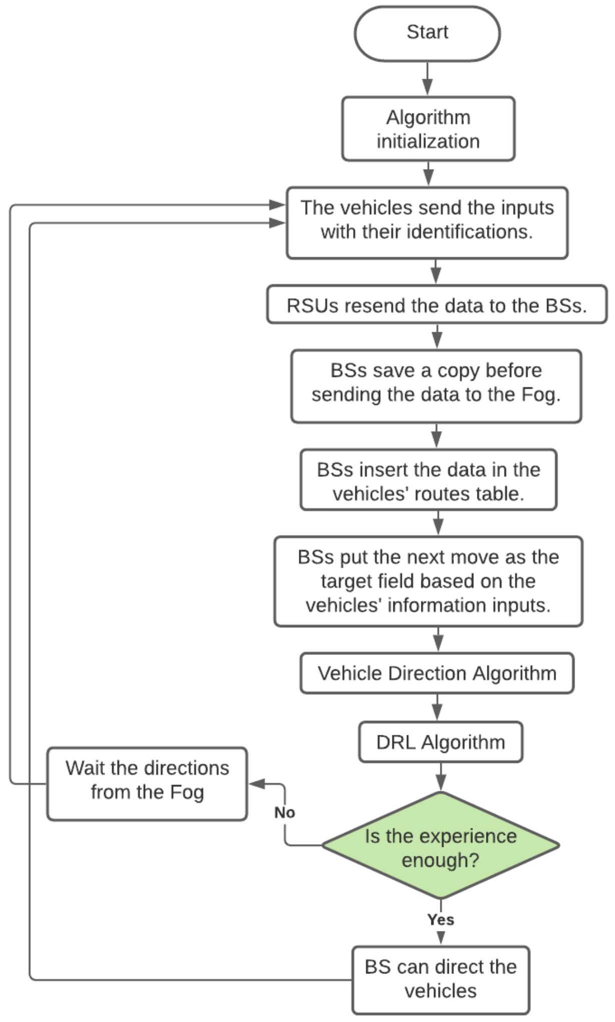

Start
Algorithm initialization
The vehicles send the inputs with their identifications.
RSUs resend the data to the BSs.
BSs save a copy before sending the data to the Fog.
BSs insert the data in the vehicles' routes table.
BSs put the next move as the target field based on the vehicles' information inputs.
Vehicle Direction Algorithm
DRL Algorithm
Is the experience enough?
No
Wait the directions from the Fog
Yes
BS can direct the vehicles

Figure 3.7: Vehicles Directions Managements Algorithm by using NN

3.3 Delay-Tolerant Networking (DTN)

Data transmission failures in vehicular networks could occur due to low signal quality. DTN protocol improves the QoS during the continuous disconnection of a network (e.g., [30, 16, 18, 61, 19]). It gives an opportunity of saving the environmental information while minimizing the end-to-end delay.

On the other hand, an urban environment usually has a lack of connectivity because of dynamic mobility and interruptions. The stochastic environment can be considered a delay/disruption tolerant network type. Moreover, the DTN protocol will show high reliability because of its data storage and transmission ability throughout the nodes. Because of communication disconnection cases, data should be forwarded to another data holder using the "store-carry-forward" mechanism. Thus, instead of dropping the data, the data can be held and transferred in a limited time [45, 56].

In DTN, we need to choose the data holder based on some parameters. First of all, the node will determine the data holder in its range that has most same following routes point and speed. Moreover, it will prefer the closest vehicle that has available buffer space. Thus, the main aim of this protocol is to improve transmission efficiency and help preserve the most data of the environment to benefit the applied DNN in our system.

3.3.1 Network Structure

- Vehicles: Choosing a new data holder will be done as long as there is a disconnection with RSU. All the vehicles' movements are random, so when nodes encounter each other in the communication range, they can get an opportunity for data transfer. However, we will have preferences of choosing the data holder. Hence, each node keeps a table locally, containing its identification, received messages, following routes, available buffer space, etc.
  If node A faces other nodes, they will exchange their tables to decide whether to transfer messages or not.

- Transferring and preferences of choosing the data holder: The selection depends on the available closest vehicle with enough buffer space.

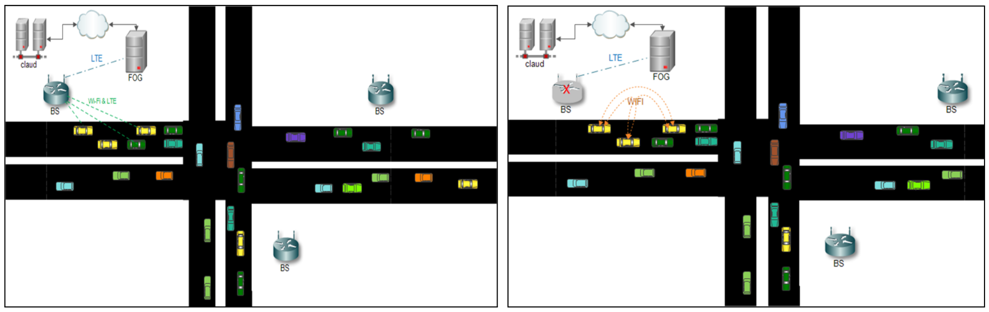
- Storage: This protocol manages the buffer space by prioritizing the currently available messages to be preserved or deleted according to the protocol priorities. When the chosen data holder receives the message, the current vehicle will remove it from the buffer. In this case, there is a time restriction to keep this data in the data holder [56].

The messages will be deleted only in the following cases:

1. When the messages are transmitted to the chosen data holder, the vehicle, after receiving acknowledgment, will remove the message from the buffer.
2. If time is up for messages to stay in the buffer, they will be forwarded and deleted.
3. When the buffer is full, it will delete the lowest priority messages to improve the utilization of network resources.

3.3.2 DTN Scheme

In this section, we present the scheme for the DTN protocol procedure on the vehicular network. The road environment that has a disconnection case will have messages that are coming from vehicles. In this case, the vehicle has a buffer and table with local information. First, each vehicle sends its information periodically to the closest RSU. Then, the vehicle will need to send the data for directions instructions; however, if there is no acknowledgment from RSUs, the vehicle will follow the DTN protocol procedures as shown in Figure 3.6 and the following steps.



(a) Before disconnection scenario (b) After disconnection scenario

Figure 3.8: DTN Scenarios

In the disconnection scenario first, the vehicle will recognize the disconnection if does not receive a response from RSUs. Then, the vehicle will calculate the buffer storage ability to keep the information in the current buffer. Next, if the buffer is not enough, it will forward the existing data to the closest preferred vehicles because of some parameters

such as closest node, same following route points, and available buffer storage for its current data.
Also, DTN protocol is a store-carry-forward, so once the bundles are forwarded and acknowledged by vehicle X, the sender will remove the data and change it to be a mobile node that could provide and receive data. Finally, the protocol will compare the free buffer space with the new message size, and it will keep it as long as it is fitting; otherwise, it will forward it to the closest node.

The following flowchart shows the steps of the DTN protocol in our system.

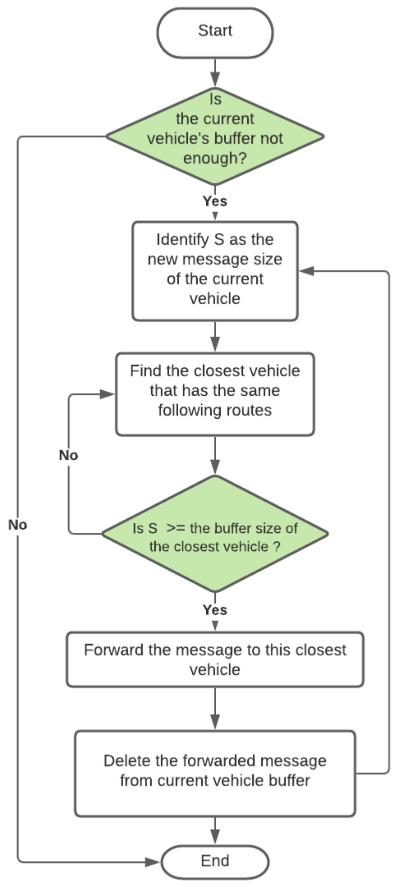

Start
Is the current vehicle's buffer not enough?
Yes
No
Identify S as the new message size of the current vehicle
Find the closest vehicle that has the same following routes
No
Is S >= the buffer size of the closest vehicle ?
Yes
Forward the message to this closest vehicle
Delete the forwarded message from current vehicle buffer
End

Figure 3.9: DTN Protocol

Chapter 4: Simulation Framework

In recent years with the advancement of technology, the Intelligent Transport System (ITS) enhances VANET to study the implementation of safety on roads. However, the high mobility and often high speed travel of large-scale scenarios will lead to several main disadvantages: deploying, performing, and testing VANET projects in a real-world environment because it can be risky, limiting, and time-consuming, particularly during tests. Therefore, simulation is the only possible solution to test our improvement idea. Simulation of VANETs networks, in particular, needs two different components: a network simulator and a trac (or mobility) simulator [63, 14].

• Mobility Simulator: to analyze performances and vehicular ad hoc network characteristics, trac simulators are needed to generate movement information and position of every single vehicle in VANETs environment.

• Network Simulator: to form and analyze the functionality of VANETs, a network simulator is needed to exchange messages between vehicles and support protocols. A proper network simulator has to hold some features of routing protocols and communication standards, such as IEEE 802.11p.

Figure 4.1 shows different available simulator tools.

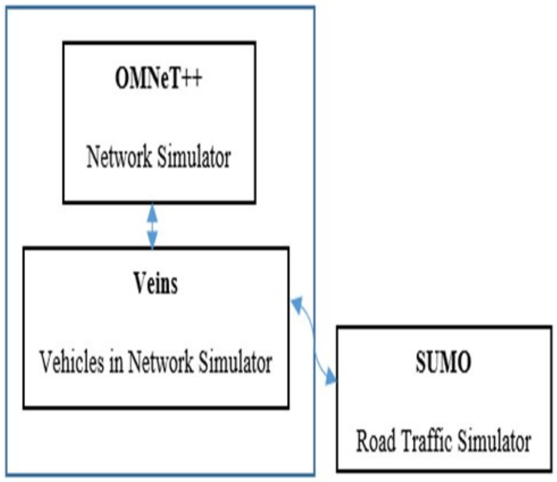


Figure 4.1: Simulation Tools.

In this work, we've chosen two open-source tools to recreate the scenarios: SUMO, a visualizing the road by trac simulator, and OMNeT++, a free vehicular network simulator.

4.1 Simulation Tools

4.1.1 OMNeT++

OMNeT++ is a simulator with a component-based C++ simulation library and framework. It has generic architecture, and it can model many diverse problems:
• Wired and wireless communication networks.
• Protocol.
• Queuing networks.
• Multiprocessors and other distributed hardware systems.
• Hardware architectures.
• Assessing the performance of software systems.
• Event approach, and mapped into entities communicating by exchanging messages.
It is an excellent and powerful tool in the simulation and research field because of its flexibility. Another key strength is providing feasible architecture by combining several of them to implement different algorithms to create complex models.

OMNET++ is a Discrete Event Simulation (DES) framework. Therefore, the situation change events occur only at a discrete time, taking zero time to execute entirely. All the structures introduced so far are clarified in NEtwork Description (NED) language. NED allows declaring the network topology and its connection. Another important file for the configuration is omnetpp.ini, which has the parameters needed to start the simulation [65]. I have used OMNET++ as the simulation software "Framework" to simulate all the scenarios that will be talking about in the following section, inside OMNET++ [73], I have used the following libraries to achieve my results in the desired way as listed as follows:

4.1.1.1 SimuLTE

SimuLTE for OMNeT++ can be applied to interpret and evaluate the performance of LTE and LTE Advanced networks. It is an open-source project developed by researchers to assess the network environments and allows extension with new algorithms and protocols. SimuLTE should be installed on top of OMNET++ and INET Framework. It is a specific framework made to simulate 4G in some of the scenarios to check the vehicular behavior [65].

4.1.1.2 INET Framework

OMNET++ has external extensions that we can use to design and simulate the wireless network, such as INET Framework. The INET Framework is an open-source model that can be installed on top of OMNET++. In addition, this framework is used to simulate wired and mobile networks. It contains IPv4, IPv6, TCP, SCTP, UDP protocols implementations. Furthermore, the INET framework also uses for the communication between the network layer and application layer [14].

4.1.2 SUMO

Simulation of Urban MObility (SUMO) is an open-source, trac simulator mainly formed in 2001 by the Institute of Transportation Systems (ITS) employees at the German Aerospace Center. A trac is a particular type of mobility modeling approach for which several parameters determine each single, such as (positioning, velocity, and direction).

It is a time-discrete simulator, and it has an expression about the number of variables associated with each vehicle and the opportunity to set/retrieve their value at each instant of the simulation.

Simulations are deterministic, but we can introduce randomness. In addition, the tool NETEDIT easily allows you to create your map. On the other hand, SUMO can import an existing urban configuration and edit it if necessary. SUMO is written in C++, and everything is easily defined by xml, such as net.xml and rou.xml inside. This simulation environment is provided with a user-friendly GUI interface.

Its primary duty is to be connected with OMNET++ to download the maps needed, such as "Portland map." So, the vehicles will be tracked for the simulation results [65].

4.1.3 Veins and TraCI

Vehicles in Network Simulation (Veins) is an open-source framework for running vehicular simulations. It is based on two well-established simulators: OMNeT++ and SUMO. This library contains a specific extension responsible for communicating the two simulators, where SUMO acts as a server, and the OMNET++ is the client. Thus, we will have queries about mobility to update its network topology accordingly. The Trac Control Interface (TraCI) allows the developers to recover and edit vehicle parameters, such as changing a particular route because of an event. It's made for vehicular networks parts and its connection with infrastructure and mobility, and it's helpful to simulate the scenarios in actual behavior [65].
Our system frameworks hierarchy is shown in Figure 4.2.

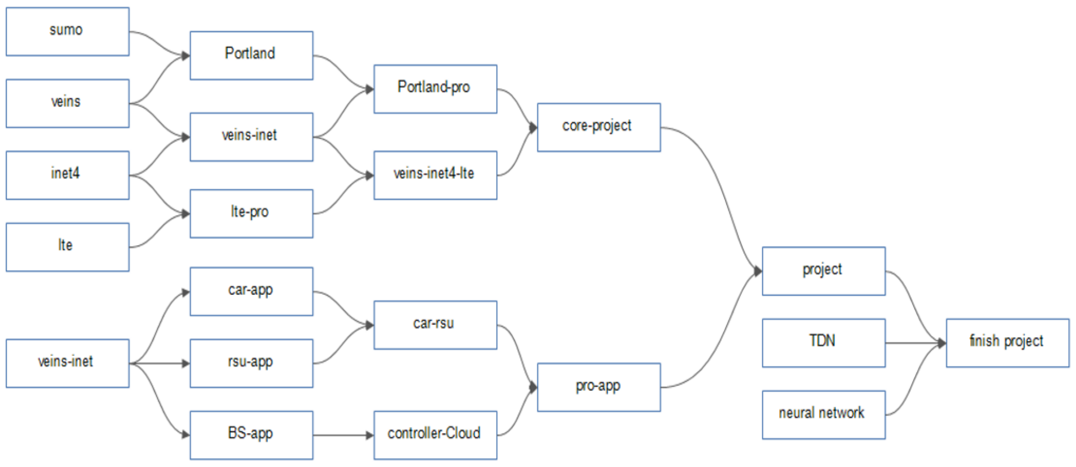


Figure 4.2: System Phases

4.1.4 Deep RL Engine

We are using Keras on top of TensorFlow to build complex neural network topologies. OMNeT++ (C++) and Keras (Python) are integrated by using text files to perform a mechanism to communicate between them.

4.1.4.1 Keras

It is an open-source library written in Python that runs on top of machine learning frameworks like TensorFlow. Using Keras lets us build complex neural network topologies with simply a few lines of code while keeping the power of the neural network. We created a feedforward fully connected deep neural network composed of n hidden layers between the input and output layers. The integration between the Python and C++ environments lets them communicate using text files so that the deep RL engine will wait for the generation of the files from OMNET++. When it receives the data, it will generate the output as a text file containing the action to execute on the simulator. On the OMNeT++ side, there is an asynchronous timer that checks periodically for action availability. When the file is available, the simulator will be able to read the action and change the speed or the direction instructions based on the street situation. Also, it will control the data migration between BSs and Fog controllers from one to another. After the action execution, the RL agent (BS or Fog) observes the reward obtained. Then, the performance indexes provided by the OMNeT++ will check if the action performed has increased it. Finally, the agent will use the reward in this thought as outcomes that help it to know if the executed action is an appropriate choice in that specific state [28, 29].

4.2 Simulation Setup

This section shows the used environment for simulation and all conditions of the setups as clarified below.

- 70% of the vehicles in the simulation are fully autonomous "self-driving cars".
- 30% of the simulations' vehicles are driven by humans "simulated by SUMO".
- There are two primary communication types used in this simulation, 4G and WIFI.
- There are two primary techniques used in this simulation, which are NN, and DTN protocol for disconnections scenario.
- Simulation is done in a part of Portland map. Figure 4.3 shows the used map from the OpenStreetMap.
- The system processes will migrate throughout the data and control plane (Vehicles, RSU, BS and Fog).
- We distributed the RSUs and BSs equally throughout the map, and by defining their location and ranges, the nodes can recognize each other.
- We used "randomTrips.py". It is an open-source code used to generate a set of random trips for a given network structure in a given map in SUMO. It is usually done by choosing source and destination at random or with an altered distribution. Thus, each vehicle would have a route with known start and destination points. The resulting trips will be stored in an XML file.

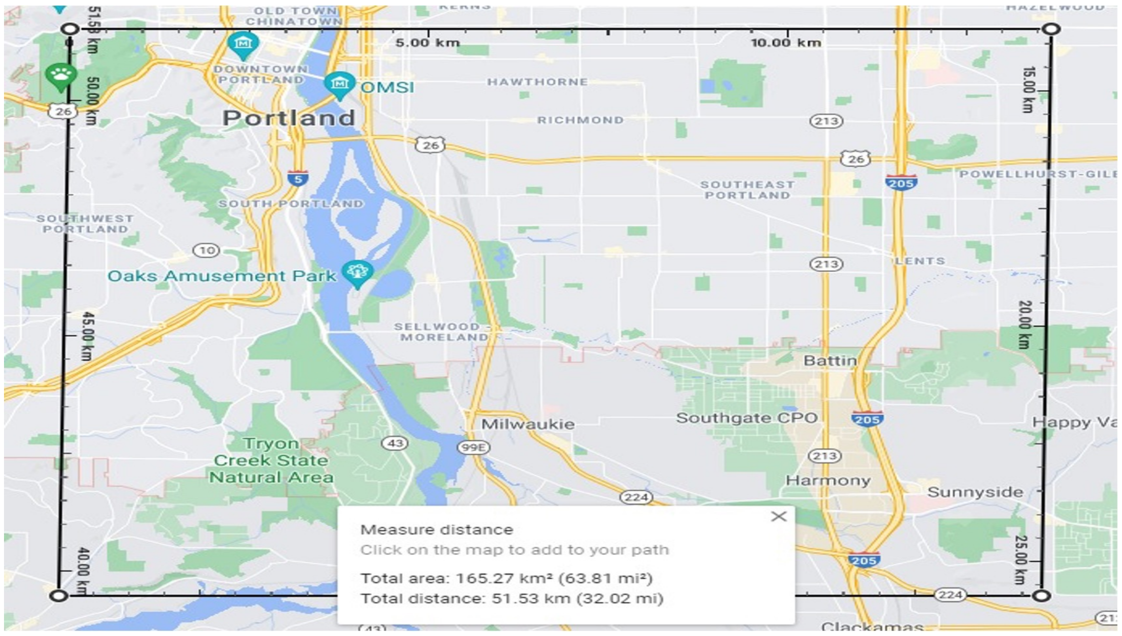
- We evaluate the path by analyzing at least the following three-points of the car's path instead of the whole path as there will be many changes throughout the street points. Figure 4.4 is shows the used system process as an example.
- The infrastructure aim is to learn how to predict the following points of the path based on the state of the streets to avoid heavy traffic at a specific time step.
- In 4G, we are using OFDM that transmit the data over many narrow-band carrier of 180 KHZ, and the details of communication technologies is shown in Table 4.1.
- Table 4.2 describes the infrastructure settings used in this simulation in detail.
- Table 4.3 clarifies the disconnection's scenario settings in our system as an experiment to see the effect of DTN protocol.
- Table 4.4 describes the settings of DRL.

Communications Technologies Comparison		
Feature	WIFI	4G
Channel width	10 MHz	Up to 100 MHz
Bit rate	3-54 Mbps	Up to 1 Gbps
Range	Up to 1 km	Up to 30 km
Coverage	Intermittent	Ubiquitous
V2I support	Yes	Yes
V2V support	Yes	No

Table 4.1: The Used Wireless Networks in the System

Numbers of cars in all scenarios	5000 cars
Numbers of RSUs in all scenarios	64 RSUs
Numbers of BSs in all scenarios	4 BSs
Numbers of Fog controller in all scenarios	One Fog
Simulation time	400 S
Interval time update	0.01 second
The total area of the Portland city map	165.27 Km^2
The trip	Random, and each vehicle has a destination.
The path controlling	Changing the route points or update the speed.

Table 4.2: Simulation setup



Measure distance
Click on the map to add to your path
Total area: 165.27 km² (63.81 mi²)
Total distance: 51.53 km (32.02 mi)

Figure 4.3: Part of Portland City Map.

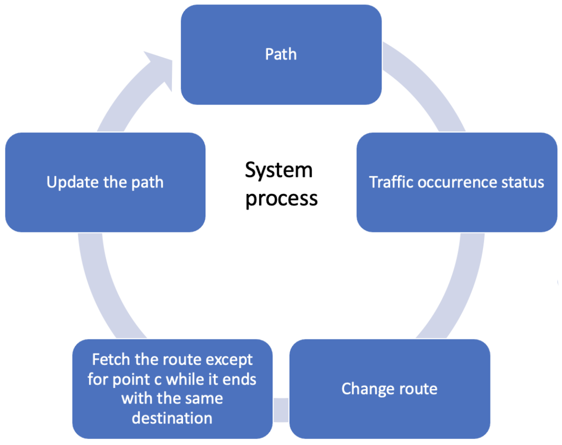

Path
Traffic occurrence status
Change route
Fetch the route except for point c while it ends with the same destination
Update the path
System process

Figure 4.4: System Process Example.

RSU-number	Shutdown Time	Operation Time
RSU[1]	100	110
RSU[16]	170	180
RSU[19]	350	360
RSU[16]	500	510
RSU[31]	700	710
RSU[42]	810	820

Table 4.3: Disconnection's Scenarios Settings in WIFI

Numbers of hidden layers	3
Numbers of neurons	16
Input	5
Output	2
Learning rate	0.001
Activation function	ReLU
Update step	30
Batch size	32

Table 4.4: DRL setup

Chapter 5: Performance Evaluation

The main challenges of our system are low latency and high reliability. The system is built in a highly structured network that includes SDN. Thus, we will have low latency, high speed, and throughput to complete the mission even in the least effective scenario, which is WIFI. Moreover, using DRL will help to reduce the latency compared to a plan without DRL. Furthermore, we will have high reliability as the throughput increases using the DTN protocol in disconnection scenarios.

5.1 Performance Results

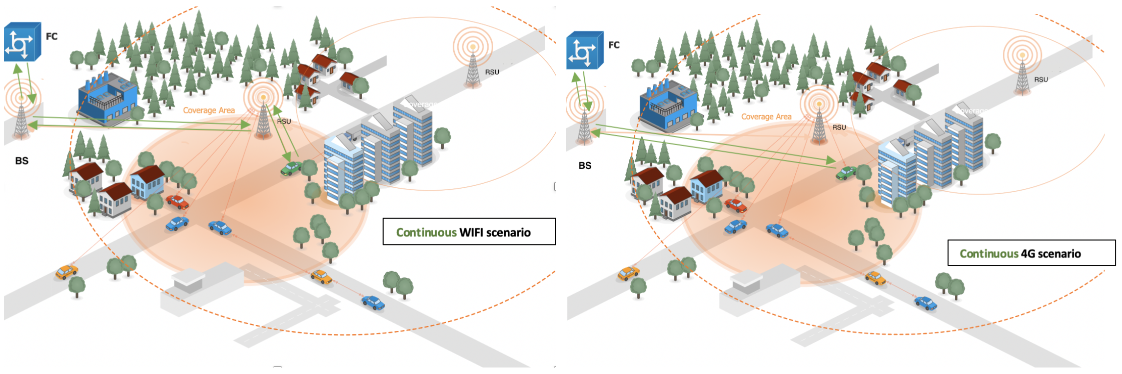
In this section, we perform a simulation-based evaluation of the proposed system. We measure the performance effectiveness of the system in different scenarios for autonomous environment. We present the 4G-NN, which means 4G scenario that uses DNN based on RL, as the best scenario. The reason is that 4G reduces the need for multi-hop routing. Preliminary simulation results show that it provides short setup times and improves vehicular communication by reducing delays. Also, DRL offloads the processing between BSs and Fog by interacting with the environment, so the system can process and respond faster. The WIFI scenario remains the least effective one compared with others. However, after applying SDN infrastructure, the WIFI scenario in all performance metrics will have good performance as this network infrastructure has different planes, which are data and control planes.

5.1.1 Results of Continuous Scenarios

In this section, we performed a simulation without any disconnection throughout trip time.

5.1.1.1 Latency

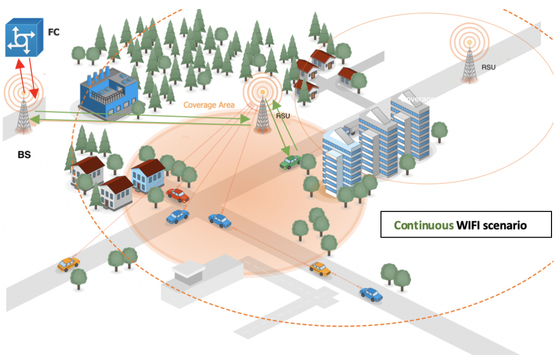
By total Latency, we mean the duration time starting from sending car information until response. In other words, it is the round trip time "RTT," which is the time needed to let the message travel to the destination and get back again. This time also includes propagation time, which is reaching the infrastructure, waiting, and processing to get the instructions for the vehicle.
The following formula calculates the average: $\frac{LatencyTotal}{NumberofCarsAtEachSingleTimeStep}$
The following figures show the continuous communications scenario that includes the car's round time to be responded to get the instructions. The arrows' green color represents that there must be communication as long as the nodes want to initiate messages.



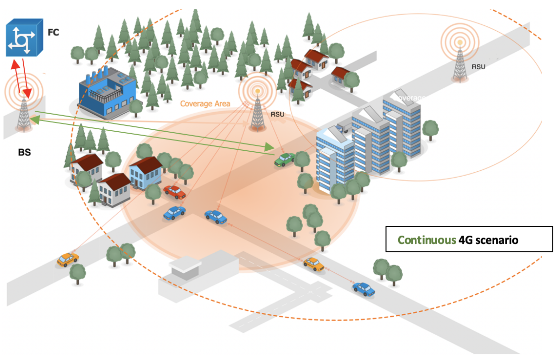
(a) Continuous WIFI Communication (b) Continuous 4G communication

Figure 5.1: Continuous Scenarios for Latency Results

The following figures show the system after applying NN. The arrows' green color represents that there must be communication as long as the nodes want to initiate messages. Also, the red color clarifies that it might not migrate throughout all layers to reach the Fog for the instructions.



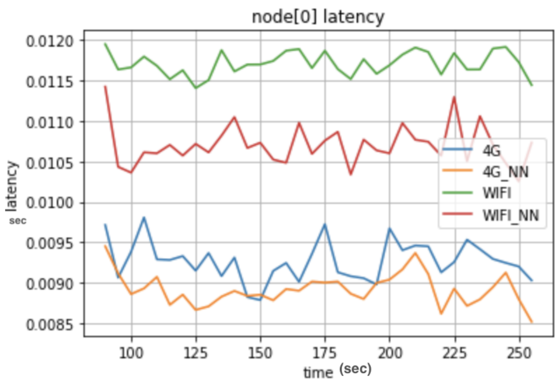
(a) Continuous WIFI Communication



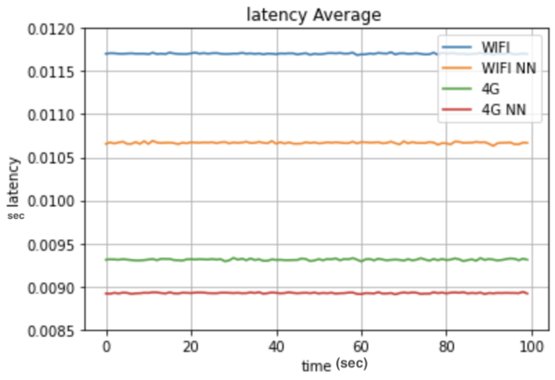
(b) Continuous 4G Communication

Figure 5.2: Continuous Scenarios for Latency Results after Applying NN

After simulating to check the differences between different scenarios and the latency response, I have plotted the following graph to clarify the result:



(a) For car node zero



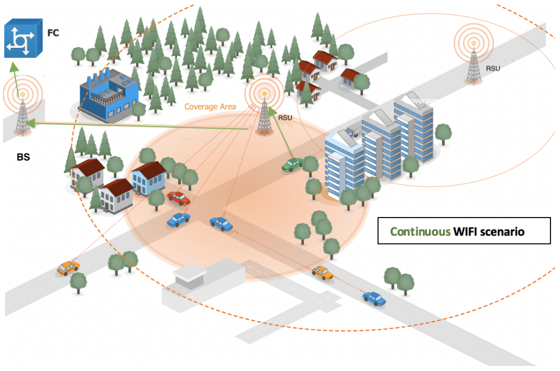
(b) For all cars

Figure 5.3: Latency Results in One and All Cars Scenarios

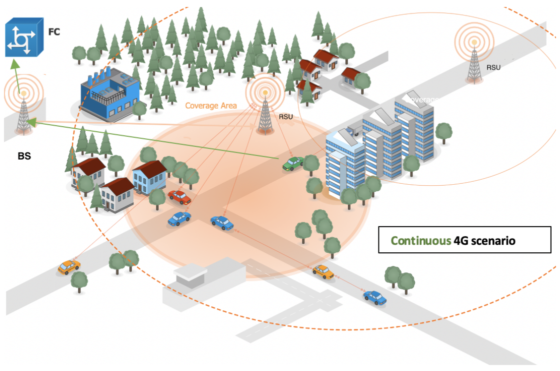
Figure 5.3 (a) shows the latency of one vehicle in four different scenarios: WIFI, WIFI-NN, 4G, and 4G-NN. It shows that the latency increases in WIFI scenarios compared with 4G because there is an extra layer of communication in WIFI, which is the communication between the car and the RSU. However, there is a significant improvement between WIFI and WIFI-NN because after applying DRL, it will offload the processing between Fog and BS. Figure 5.3 (b) shows the average latency throughout the first 100 seconds of the simulation time.

5.1.1.2 Delay

In communication between vehicles, the delay is considered the difference between the time that data has arrived at its destination and when the data has been created. The following figures show the continuous communications scenario of delay, including the message traveling time from the car to the infrastructure. The arrows' green color represents that there must be communication as long as the nodes want to initiate messages.



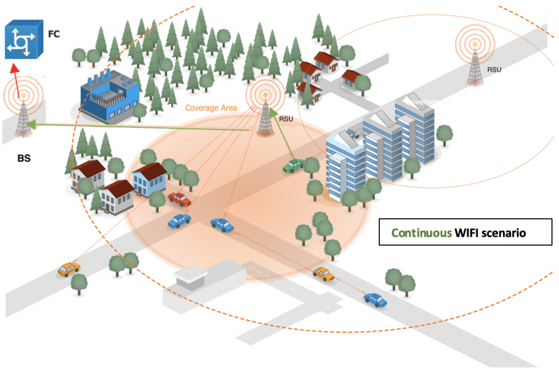
(a) Continuous WIFI Communication



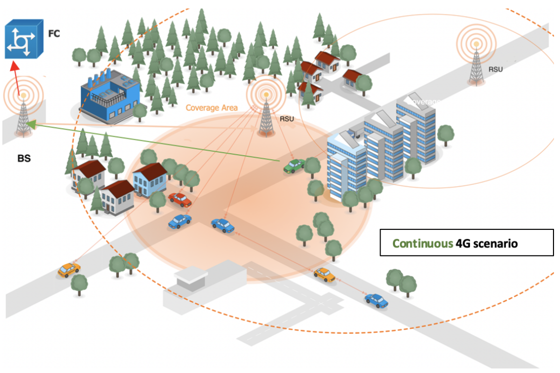
(b) Continuous 4G Communication

Figure 5.4: Continuous Scenarios for Delay Results

The following figures show the system after applying NN. The arrows' green color demonstrates that there must be communication as long as the nodes want to initiate messages. Also, the red color clarifies that it might not migrate throughout all layers to reach the Fog.



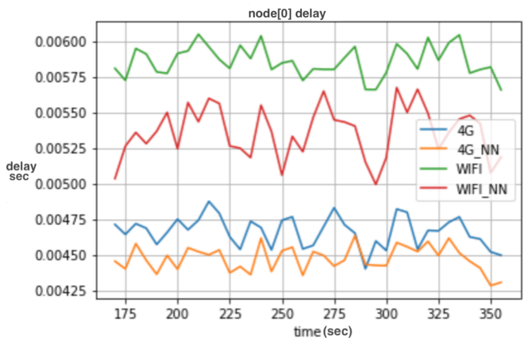
(a) Continuous WIFI communication



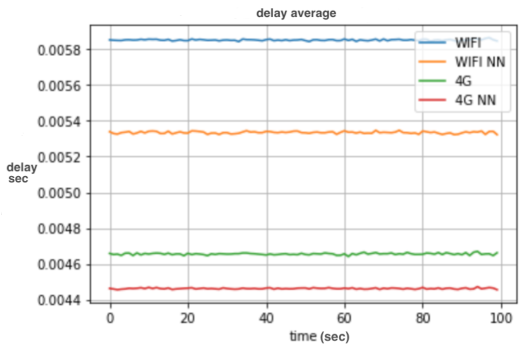
(b) Continuous 4G communication

Figure 5.5: Continuous Scenarios for Delay Results after Applying NN

The following graphs clarify the delay response results of different scenarios:



(a) For car node one



(b) For all cars

Figure 5.6: Delay Results in One and All Cars Scenarios

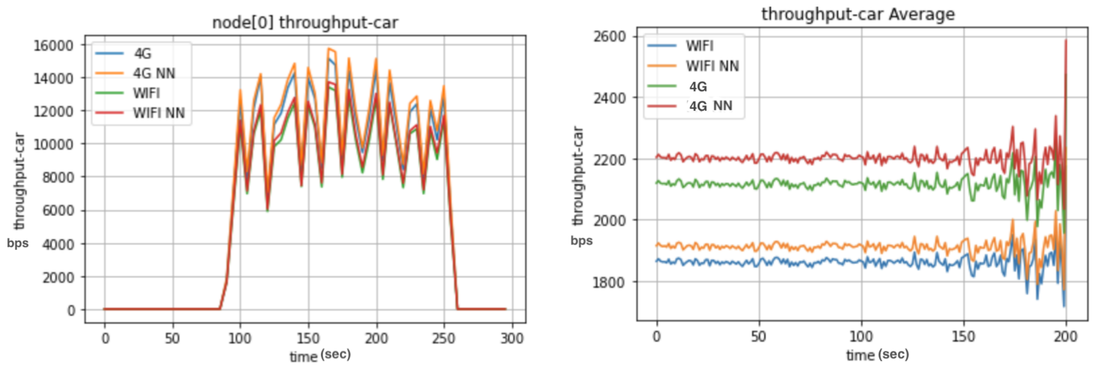
In Figure 5.6 (a), the time taken to reach the Fog controller is low. Therefore, we can consider ultra-low communication time with the upper layer, especially in the NN scenario compared with scenarios without NN. Moreover, 4G has better improvement as the vehicle reaches the BS faster. In other words, the traveling time will end with BS in the NN scenario. Figure 5.6 (b) shows the average delay throughout the first 100 seconds of the simulation time.

5.1.1.3 Throughput

• Vehicles Throughput

The throughput is the total size of received messages in bit per time in second. In other words, It is a successful amount of data that has been transmitted.

The throughput of the system is shown in the below figures.



(a) For car node zero (b) For all cars

Figure 5.7: Throughput Results in One and All Cars Scenarios

In Figure 5.7 (a), when the car[0] started to send the data successfully, the throughput increased gradually, and it reached almost 16000 bits per second as the highest throughput in the car[0]. The throughput is high in NN scenarios because applying DRL on the infrastructure such as BS will make the processing and delivering faster, so the car will have the ability to send more successful bits. Therefore, we can consider high reliability, especially in the NN scenario compared with scenarios without NN. However, the system works with the network efficiently because there will be offloading in processing and responding. Figure 5.7 (b) shows the average throughput throughout the first 200 seconds of the simulation time.

• Fog Throughput

The Fog's throughput is the total size of successful transmission of messages in bit per time in second. The Figure 5.8 is clarifying the throughput of the Fog in different scenarios.

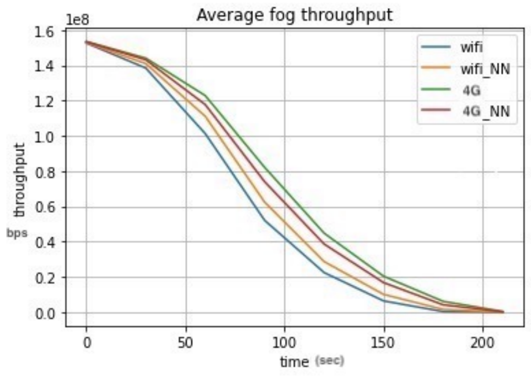


Figure 5.8: Fog Throughput.

In Figure 5.8, the Fog sends messages to identify itself as a node in this architecture at the beginning of the simulation to have a highly successful transmission. Then, vehicles start to pass the streets, and the throughput of the Fog is changing. After that, the vehicles are gradually decreased throughout time, and the fog throughput is reduced depending on the number of cars.

The throughput of the Fog in the WIFI-NN scenario will be higher than the WIFI scenario as the bandwidth of WIFI is lower than 4G. On the other hand, The throughput of Fog in 4G-NN sometimes will be lower than the 4G scenario because the data transfer of 4G, in general, is high, and it could succeed to have higher throughput than NN one. Also, in some cases, Fog transmission will be low in the NN scenario as the lower layer nodes will not deliver massive amounts of data to the upper layers.

5.1.1.4 Duration Trip

It is defined as the difference between the arrival time and start time in different scenarios of our system.
The following figures show the difference between a scenario without using NN and with using it.

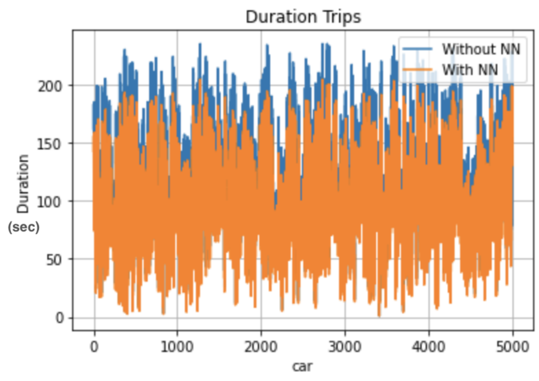


Figure 5.9: Duration trip.

Figure 5.9 shows a significant difference between vehicles' trip times in the scenario of NN and without it. The reason is that using NN will reduce the trip time as the vehicles will get responded faster. Thus, BSs can make decisions more quickly than waiting to have all the data to reach the Fog layer.

5.1.2 Results of Disconnections Scenarios

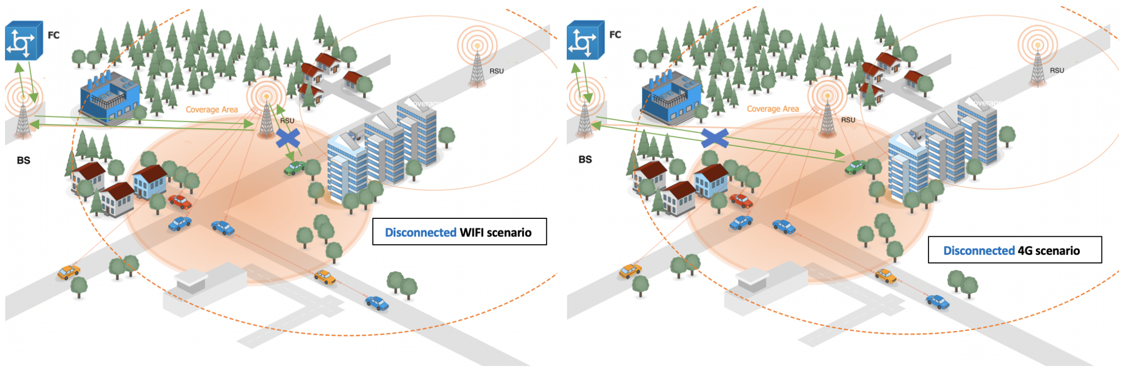
In this section, we will show the disconnection scenarios without DTN protocol when the vehicles drop the data immediately. On the other hand, we will clarify the disconnection scenario by using the DTN protocol that keeps and sends the preserved data when the communication is retrieved.

In the WIFI Scenario, the disconnection will be between the vehicles and the RSU. In the 4G scenario, the disconnection will be between the vehicles and the eNB attached to the BS.

The aim of this protocol is preserving the data as much as possible by managing the buffers between vehicles.

5.1.2.1 Latency

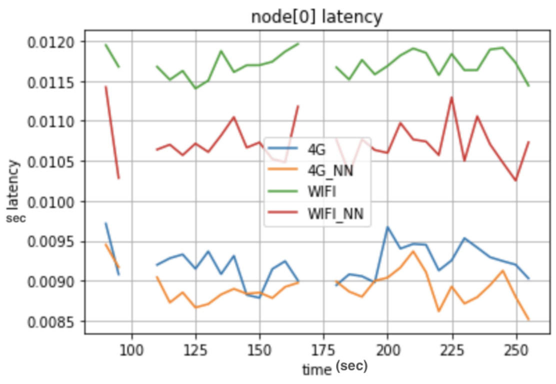
The latency or RTT in the disconnection scenario will lose the latency data in the time steps of disconnection.
The following figures shows the disconnection scenarios in WIFI and 4G.



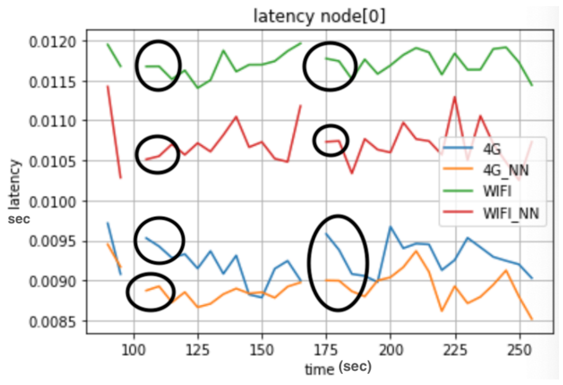
(a) WIFI (b) 4G

Figure 5.10: Disconnection Scenarios for Latency Results

For example, the below figures show the system of communication disconnection that happened in vehicles number zero passed under RSU (1) from 100 to 110-time steps and RSU (16) from 170 to 180-time steps. The simulation also contained the difference between before applying the DTN protocol and after using it, as shown in Figure 5.11.



(a) Before Applying DTN protocol



(b) After Applying DTN protocol

Figure 5.11: Disconnections Latency Results in Car (0)

We can see the difference in disconnections between the two above figures, "Before applying DTN protocol and After," which is clarified in black circles, confirming that DTN protocol enhanced the communication and made it safer to keep the communication alive compared with the latency data before applying the DTN. Thus, we can say:

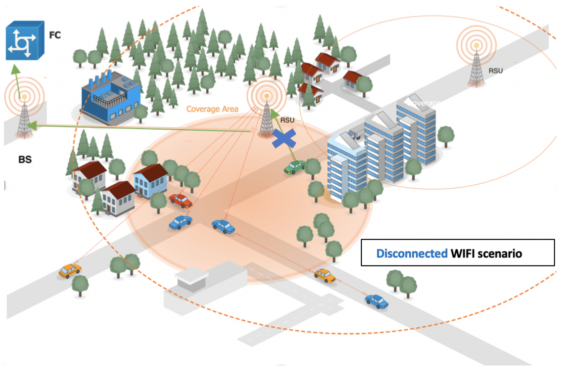
• Disconnection scenario without DTN:
As shown in Figure 5.11 (a), the car drops the data immediately.

• Disconnection scenario with DTN:
As shown in Figure 5.11 (b), it keeps the data and sends the old and new data when the communication is retrieved.

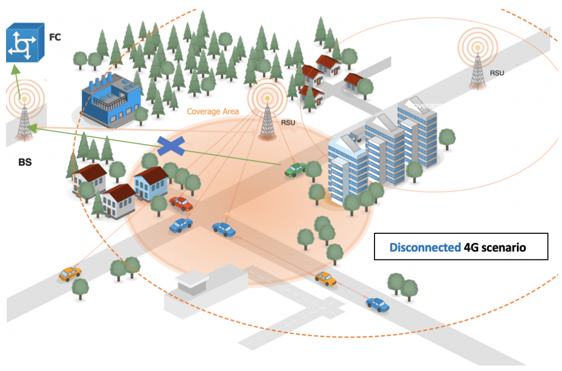
The benefit of this procedure is that we will preserve much of the environmental data, which will help the continuity of DNN feeding.

5.1.2.2 Delay

The delay data will be lost in the time steps of disconnection. The following figures show the disconnection scenario in delay.



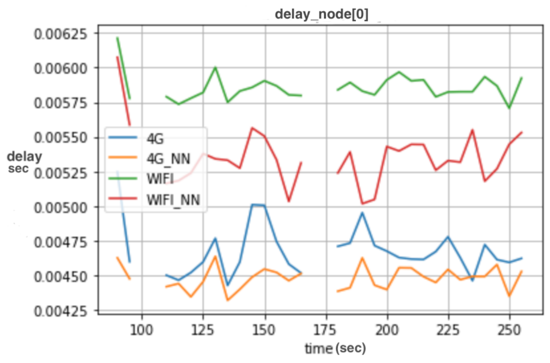
(a) WIFI



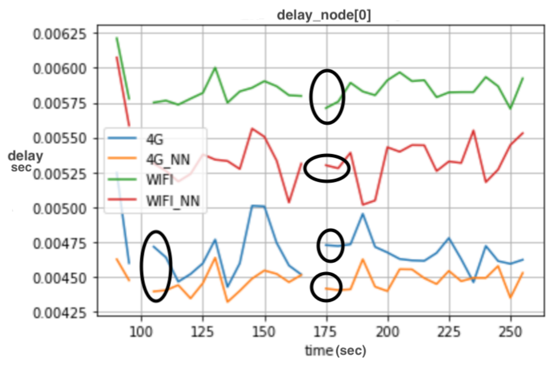
(b) 4G

Figure 5.12: Disconnection Scenarios for Delay Results

Figure 5.13 shows the system of communication disconnection that happened in the first vehicle that passed under RSU (1) from 100 to 110-time steps and RSU (16) from 170 to 180-time steps. The simulation clarifies the difference between before applying the DTN protocol and after using it, as shown in below figures.



(a) Before Applying DTN protocol



(b) After Applying DTN protocol

Figure 5.13: Disconnections Delay Results in Car (0)

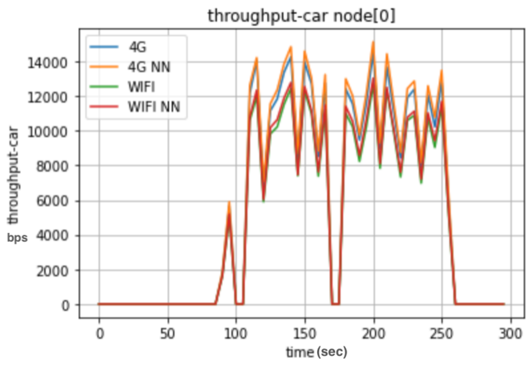
The same result can be obtained when applying DTN protocol, which again ensures the enhancement in keep the data in a place away from connections, and the graphs above clarify this point. We can see the difference in disconnections between the two figures, "Before applying DTN protocol and After," clarified in black circles.

5.1.2.3 Throughput

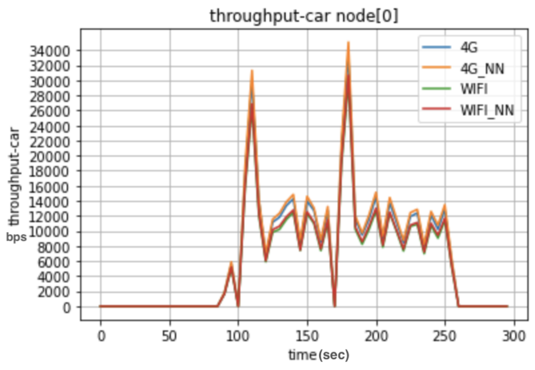
The vehicles' throughput will be high after applying the DTN protocol. However, the Fog throughput will be almost same because the Fog can communicate with lower nodes layer such as BS.

• Vehicles' Throughput

The throughput will be decreased in the disconnection scenarios. For example, Figure 5.14 (a) shows the system of communication disconnection that happened in vehicle number zero that passed under RSU (1) from 100 to 110-time steps and RSU (16) from 170 to 180-time steps. In addition, Figure 5.14 demonstrates the difference between before applying the DTN protocol and after using it. Also, in Figure 5.14 (b), We can see a significant increase in the throughput from 16000 to 35000 bits per second compared with Figure 5.14 (a).



(a) Before Applying DTN protocol



(b) After Applying DTN protocol

Figure 5.14: Disconnections Throughput Results in Car (0)

5.1.2.4 Duration Trip

It is defined as the difference between the arrival time and start time in different scenarios of our system. In addition, to clarify it, we are showing the difference between a scenario without using DTN and using it in a disconnection situation.

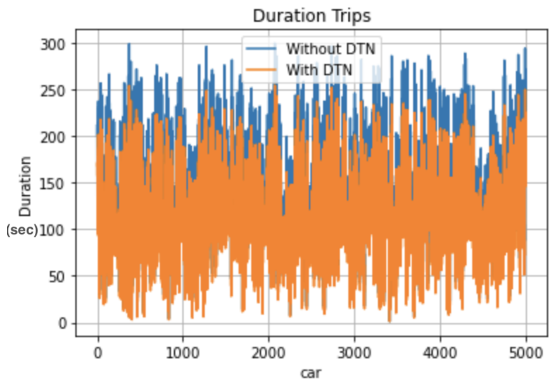


Figure 5.15: Duration trip.

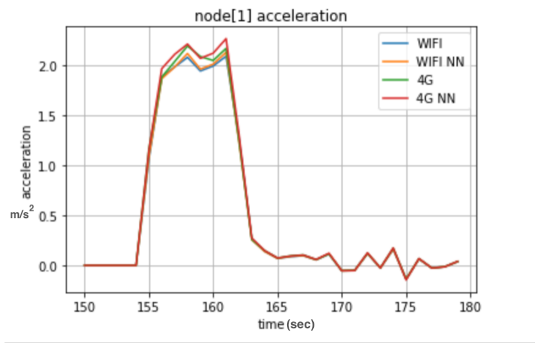
Figure 5.15 shows a significant difference between vehicles' trip times in the scenario of DTN and scenarios without it. The reason is that using DTN will reduce the trip time in disconnection situations because instead of losing the data that make the vehicle slower, we can preserve the data that help to direct the vehicle to its destination faster. In other words, keeping the vehicles' data will improve the NN accuracy as we are feeding the network continuously.

5.2 Mobility Results

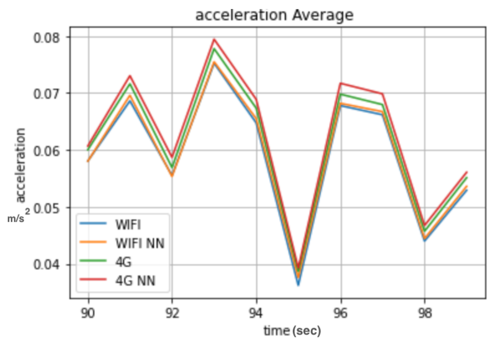
5.2.1 Acceleration

The acceleration here is defined as the difference between the current speed and last speed by the update interval.

$= \frac{(speed - lastspeed)}{updateInterval}$



(a) For car node one



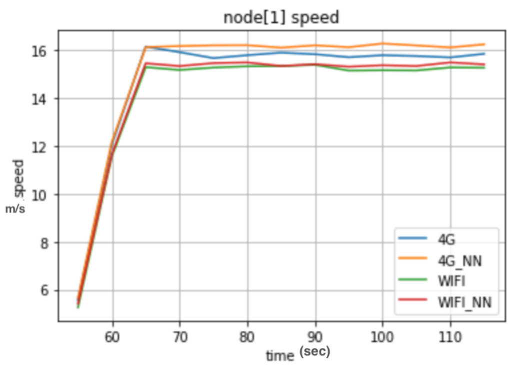
(b) For all cars

Figure 5.16: Acceleration Results in One and All Cars Scenarios

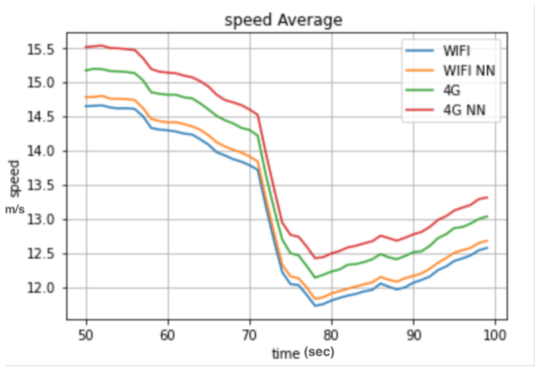
In Figure 5.16 (a), the vehicle's movement is stable in all scenarios; however, the NN could affect the acceleration by increasing it. Figure 5.16 (b) shows the average acceleration in all scenarios. We can see the NN is higher than others because the amount of handled data throughout the upper layer is lower in the NN. As a result, the vehicles could take the decision faster, and the vehicle's movement will be stable and fast.

5.2.2 Speed

The speed of vehicles based on traffic will change to avoid the traffic. Hence, the change depends on path decisions and associate destinations.



(a) For car node one



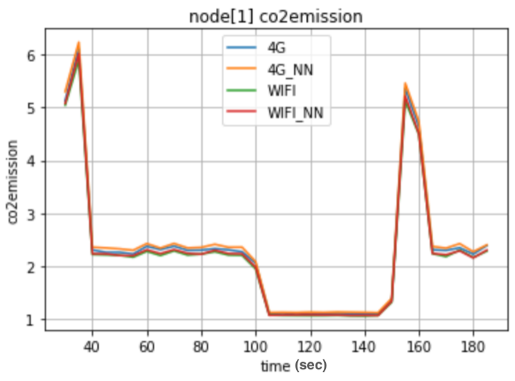
(b) For all cars

Figure 5.17: Speed Results in One and All Cars Scenarios

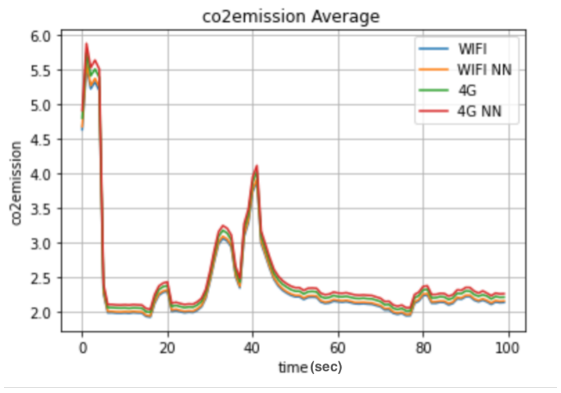
In Figure 5.17 (a), We can see the vehicle's performance in WIFI NN is lower than the 4G because, in WIFI scenarios, we will have more hops as the vehicle communicates with RSU instead of communicating directly with BS. Figure 5.17 (b) shows the average speed in all scenarios. Again, we can see the NN is higher than others because the traffic is reduced. Thus, the speed will be increased.

5.2.3 CO2 Emission

The petrol engine of the car will burn fuel and in doing so produces carbon dioxide (CO2).



(a) For car node one



(b) For all cars

Figure 5.18: CO2 Results in One and All Cars Scenarios

In Figure 5.18, they are almost the same. However, the NN got the highest CO2 emissions because, in the NN scenario, it could change the route to another street with a traffic light instead of a highway. Thus, we will have higher CO2 emissions in NN scenarios.

Chapter 6: Conclusion and Future Work

Ultra-low latency, reliability, and efficient communication in the automotive industry are essential to ensure a vehicular network's safety. However, addressing these requirements is complex and needs efficient orchestration of network functionality at different structure levels. Thus, in this work, we covered the essential requirements of vehicular networks by applying integrated technologies. First, we introduced a distributed SDN network architecture to establish and enforce the infrastructure. Second, we described the applied suitable intelligence technique in our internet of vehicle, deep reinforcement learning to improve the processing in lower latency. Finally, we presented the protocol used in the disconnection scenarios to improve the system's reliability and to keep the most critical information of the environment.

This thesis presented these technologies to find alternative routes and avoid traffic in a high mobility environment. The simulation results have demonstrated that the used technologies have reduced handling latency with high throughput, especially in the DNN scenarios. Also, using the DTN protocol has increased the reliability of the system by preserving the information to keep feeding the neural networks, so this system has increased DNN accuracy. Moreover, we have presented mobility results such as speed and acceleration to illustrate how the vehicles are smoothly moving in our system.

Future works will be dedicated to better integrating the algorithm with the OMNeT++ environment by using more realistic traffic data and comparing other proposed solutions to improve the system performance. Moreover, we could focus more on vehicular resource management by scheduling them to overcome the ultra-low latency challenges, so we could improve the system by dealing with algorithms that distribute resources efficiently between nodes.

The union of these technologies can overcome these requirements. However, in the coming years, 6G will enable a completely ubiquitous network across various devices as it will support the novelty of Vehicle to Infrastructure (V2X) communications efficiently.

Moreover, it can exchange the data through vehicle sensors using high-bandwidth and high-reliability links. Before electric vehicles travel throughout our streets, 6G will improve safety on the road and guarantee traffic efficiency in the vehicular networks to have ultra-low latency services. Hence, advanced wireless network and using AI in resources management on top of robust network infrastructure such as SDN will improve the processing to have ultra-low latency and reliability requirements in vehicular networks.

Bibliography

[1] Sherif Abdelwahab and Bechir Hamdaoui. Flocking virtual machines in quest for responsive iot cloud services. In *2017 IEEE International Conference on Communications (ICC)*, pages 1–6. IEEE, 2017.

[2] Sherif Abdelwahab, Bechir Hamdaoui, Mohsen Guizani, and Ammar Rayes. Enabling smart cloud services through remote sensing: An internet of everything enabler. *IEEE Internet of Things Journal*, 1(3):276–288, 2014.

[3] Sherif Abdelwahab, Bechir Hamdaoui, Mohsen Guizani, and Taieb Znati. Cloud of things for sensing as a service: sensing resource discovery and virtualization. In *2015 IEEE Global Communications Conference (GLOBECOM)*, pages 1–7. IEEE, 2015.

[4] Sherif Abdelwahab, Bechir Hamdaoui, Mohsen Guizani, and Taieb Znati. Cloud of things for sensing-as-a-service: Architecture, algorithms, and use case. *IEEE Internet of Things Journal*, 3(6):1099–1112, 2016.

[5] Sherif Abdelwahab, Bechir Hamdaoui, Mohsen Guizani, and Taieb Znati. Network function virtualization in 5g. *IEEE Communications Magazine*, 54(4):84–91, 2016.

[6] Sherif Abdelwahab, Sophia Zhang, Ashley Greenacre, Kai Ovesen, Kevin Bergman, and Bechir Hamdaoui. When clones flock near the fog. *IEEE Internet of Things Journal*, 5(3):1914–1923, 2018.

[7] Abdulhameed Alelaiwi. An efficient method of computation offloading in an edge cloud platform. *Journal of Parallel and Distributed Computing*, 127:58–64, 2019.

[8] Elmustafa Sayed Ali, Mohammad Kamrul Hasan, Rosilah Hassan, Rashid A Saeed, Mona Bakri Hassan, Shayla Islam, Nazmus Shaker Nafi, and Savitri Bevinakoppa. Machine learning technologies for secure vehicular communication in internet of vehicles: Recent advances and applications. *Security and Communication Networks*, 2021, 2021.

[9] Mohamed Alkalbani, Bechir Hamdaoui, Nizar Zorba, and Ammar Rayes. A blockchain-based iot networks-on-demand protocol for responsive smart city applications. In *2019 IEEE Global Communications Conference (GLOBECOM)*, pages 1–6. IEEE, 2019.

[10] Jarallah Alqahtani, Sultan Alanazi, and Bechir Hamdaoui. Traffic behavior in cloud data centers: A survey. In *2020 International Wireless Communications and Mobile Computing (IWCMC)*, pages 2106–2111. IEEE, 2020.

[11] Jarallah Alqahtani, Hassan H Sinky, and Bechir Hamdaoui. Clustered multicast source routing for large-scale cloud data centers. *IEEE Access*, 9:12693–12705, 2021.

[12] Omar Alsaleh, Bechir Hamdaoui, and Ammar Rayes. Improving quality of data user experience in 4g distributed telecommunication systems. In *2012 International Conference on Collaboration Technologies and Systems (CTS)*, pages 34–38. IEEE, 2012.

[13] Omar Alsaleh, Pavithra Venkatraman, Bechir Hamdaoui, and Alan Fern. Enabling opportunistic and dynamic spectrum access through learning techniques. *Wireless Communications and Mobile Computing*, 11(12):1497–1506, 2011.

[14] Carla Amatetti. *Study of anti-congestion algorithms for autonomous and connected vehicles.* PhD thesis, Politecnico di Torino, 2018.

[15] Jiaqi Bao, Bechir Hamdaoui, and Weng-Keen Wong. Iot device type identification using hybrid deep learning approach for increased iot security. In *2020 International Wireless Communications and Mobile Computing (IWCMC)*, pages 565–570. IEEE, 2020.

[16] Fatima Zohra Benhamida, Abdelmadjid Bouabdellah, and Yacine Challal. Using delay tolerant network for the internet of things: Opportunities and challenges. In *2017 8th International Conference on Information and Communication Systems (ICICS)*, pages 252–257. IEEE, 2017.

[17] Kamal Benzekki, Abdeslam El Fergougui, and Abdelbaki Elbelrhiti Elalaoui. Software-defined networking (sdn): a survey. *Security and communication networks*, 9(18):5803–5833, 2016.

[18] Kyle Bradford, Max Brugger, Samina Ehsan, Bechir Hamdaoui, and Yevgeniy Kovchegov. Data loss modeling and analysis in partially-covered delay-tolerant networks. In *2011 Proceedings of 20th International Conference on Computer Communications and Networks (ICCCN)*, pages 1–7. IEEE, 2011.

[19] Max Brugger, Kyle Bradford, Samina Ehsan, Bechir Hamdaoui, and Yevgeniy Kovchegov. Analytic bounds on data loss rates in mostly-covered mobile dtns. *IEEE transactions on wireless communications*, 12(7):3121–3129, 2013.

[20] Nessrine Chakchouk and Bechir Hamdaoui. Uplink performance characterization and analysis of two-tier femtocell networks. *IEEE transactions on vehicular technology*, 61(9):4057–4068, 2012.

[21] Djabir Abdeldjalil Chekired, Mohammed Amine Togou, Lyes Khoukhi, and Adlen Ksentini. 5g-slicing-enabled scalable sdn core network: Toward an ultra-low latency of autonomous driving service. *IEEE Journal on Selected Areas in Communications*, 37(8):1769–1782, 2019.

[22] Mehiar Dabbagh, Bechir Hamdaoui, Mohsen Guizani, and Ammar Rayes. Energy-efficient resource allocation and provisioning framework for cloud data centers. *IEEE Transactions on Network and Service Management*, 12(3):377–391, 2015.

[23] Mehiar Dabbagh, Bechir Hamdaoui, Mohsen Guizani, and Ammar Rayes. Exploiting task elasticity and price heterogeneity for maximizing cloud computing profits. *IEEE Transactions on Emerging Topics in Computing*, 6(1):85–96, 2015.

[24] Mehiar Dabbagh, Bechir Hamdaoui, Mohsen Guizani, and Ammar Rayes. Online assignment and placement of cloud task requests with heterogeneous requirements. In *2015 IEEE Global Communications Conference (GLOBECOM)*, pages 1–6. IEEE, 2015.

[25] Mehiar Dabbagh, Bechir Hamdaoui, Mohsen Guizani, and Ammar Rayes. Software-defined networking security: pros and cons. *IEEE Communications Magazine*, 53(6):73–79, 2015.

[26] Mehiar Dabbagh, Bechir Hamdaoui, Mohsen Guizani, and Ammar Rayes. Toward energy-efficient cloud computing: Prediction, consolidation, and overcommitment. *IEEE network*, 29(2):56–61, 2015.

[27] Mehiar Dabbagh, Ammar Rayes, Bechir Hamdaoui, and Mohsen Guizani. Peak shaving through optimal energy storage control for data centers. In *2016 IEEE International Conference on Communications (ICC)*, pages 1–6. IEEE, 2016.

[28] Fabrizio De Vita, Dario Bruneo, Antonio Puliafito, Giovanni Nardini, Antonio Virdis, and Giovanni Stea. A deep reinforcement learning approach for data migration in multi-access edge computing. In *2018 ITU Kaleidoscope: Machine Learning for a 5G Future (ITU K)*, pages 1–8. IEEE, 2018.

[29] Fabrizio De Vita, Giovanni Nardini, Antonio Virdis, Dario Bruneo, Antonio Puliafito, and Giovanni Stea. Using deep reinforcement learning for application relocation in multi-access edge computing. *IEEE Communications Standards Magazine*, 3(3):71–78, 2019.

[30] Samina Ehsan, Kyle Bradford, Max Brugger, Bechir Hamdaoui, Yevgeniy Kovchegov, Douglas Johnson, and Mounir Louhaichi. Design and analysis of delay-tolerant sensor networks for monitoring and tracking free-roaming animals. *IEEE Transactions on Wireless Communications*, 11(3):1220–1227, 2012.

[31] Abdurrahman Elmaghbub and Bechir Hamdaoui. Leveraging hardware-impaired out-of-band information through deep neural networks for robust wireless device classification. *arXiv preprint arXiv:2004.11126*, 2020.

[32] Abdurrahman Elmaghbub, Bechir Hamdaoui, and Arun Natarajan. Widescan: Exploiting out-of-band distortion for device classification using deep learning. In *GLOBECOM 2020-2020 IEEE Global Communications Conference*, pages 1–6. IEEE, 2020.

[33] Mahdi Ben Ghorbel, Bechir Hamdaoui, Rami Hamdi, Mohsen Guizani, and MohammadJavad NoroozOliaee. Distributed dynamic spectrum access with adaptive power allocation: Energy efficiency and cross-layer awareness. In *2014 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 694–699. IEEE, 2014.

[34] Mohamed Grissa, Bechir Hamdaoui, and Attila A Yavuz. Unleashing the power of multi-server pir for enabling private access to spectrum databases. *IEEE Communications Magazine*, 56(12):171–177, 2018.

[35] Mohamed Grissa, Attila A Yavuz, and Bechir Hamdaoui. Cuckoo filter-based location-privacy preservation in database-driven cognitive radio networks. In *2015 World Symposium on Computer Networks and Information Security (WSCNIS)*, pages 1–7. IEEE, 2015.

[36] Mohamed Grissa, Attila A Yavuz, and Bechir Hamdaoui. When the hammer meets the nail: Multi-server pir for database-driven crn with location privacy assurance. In *2017 IEEE Conference on Communications and Network Security (CNS)*, pages 1–9. IEEE, 2017.

[37] Mohamed Grissa, Attila A Yavuz, and Bechir Hamdaoui. Trustsas: A trustworthy spectrum access system for the 3.5 ghz cbrs band. In *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*, pages 1495–1503. IEEE, 2019.

[38] Mohamed Grissa, Attila Altay Yavuz, Bechir Hamdaoui, and Chittibabu Tirupathi. Anonymous dynamic spectrum access and sharing mechanisms for the cbrs band. *IEEE Access*, 9:33860–33879, 2021.

[39] Bechir Hamdaoui. Optimal discovery of bandwidth opportunities in spectrum agile networks. In *IEEE GLOBECOM 2008-2008 IEEE Global Telecommunications Conference*, pages 1–5. IEEE, 2008.

[40] Bechir Hamdaoui, Mohamed Alkalbani, Ammar Rayes, and Nizar Zorba. Iotshare: A blockchain-enabled iot resource sharing on-demand protocol for smart city situation-awareness applications. *IEEE Internet of Things Journal*, 7(10):10548–10561, 2020.

[41] Bechir Hamdaoui, Mohamed Alkalbani, Taieb Znati, and Ammar Rayes. Unleashing the power of participatory iot with blockchains for increased safety and situation awareness of smart cities. *IEEE Network*, 34(2):202–209, 2019.

[42] Bechir Hamdaoui, Tamara Alshammari, and Mohsen Guizani. Exploiting 4g mobile user cooperation for energy conservation: challenges and opportunities. *IEEE wireless communications*, 20(5):62–67, 2013.

[43] Bechir Hamdaoui, Abdurrahman Elmaghbub, and Siefeddine Mejri. Deep neural network feature designs for rf data-driven wireless device classification. *IEEE Network*, 2020.

[44] Bechir Hamdaoui, Pavithra Venkatraman, and Mohsen Guizani. Opportunistic exploitation of bandwidth resources through reinforcement learning. In *GLOBECOM 2009-2009 IEEE Global Telecommunications Conference*, pages 1–6. IEEE, 2009.

[45] Youssef Harrati and Abdelmounaim Abdali. Maxhopcount: Dtn congestion control algorithm under maxprop routing. *IJCSNS*, 17(5):206, 2017.

[46] Liang Huang, Suzhi Bi, and Ying-Jun Angela Zhang. Deep reinforcement learning for online computation offloading in wireless powered mobile-edge computing networks. *IEEE Transactions on Mobile Computing*, 19(11):2581–2593, 2019.

[47] Airton Ishimori, Fernando Farias, Eduardo Cerqueira, and Antônio Abelém. Control of multiple packet schedulers for improving qos on openflow/sdn networking. In *2013 Second European Workshop on Software Defined Networks*, pages 81–86. IEEE, 2013.

[48] Hongjing Ji, Osama Alfarraj, and Amr Tolba. Artificial intelligence-empowered edge of vehicles: architecture, enabling technologies, and applications. *IEEE Access*, 8:61020–61034, 2020.

[49] Koray Kavukcuoglu, D Silver, AA Rusu, J Veness, MG Bellemare, A Graves, M Riedmiller, AK Fidjel, G Ostrovski, S Petersen, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

[50] Bassem Khalfi, Bechir Hamdaoui, and Mohsen Guizani. Extracting and exploiting inherent sparsity for efficient iot support in 5g: Challenges and potential solutions. *IEEE Wireless Communications*, 24(5):68–73, 2017.

[51] Bassem Khalfi, Adem Zaid, and Bechir Hamdaoui. When machine learning meets compressive sampling for wideband spectrum sensing. In *2017 13th International Wireless Communications and Mobile Computing Conference (IWCMC)*, pages 1120–1125. IEEE, 2017.

[52] Mashael Khayyat, Ibrahim A Elgendy, Ammar Muthanna, Abdullah S Alshahrani, Soltan Alharbi, and Andrey Koucheryavy. Advanced deep learning-based computational offloading for multilevel vehicular edge-cloud computing networks. *IEEE Access*, 8:137052–137062, 2020.

[53] Zhe Li, Gwendal Simon, and Annie Gravey. Caching policies for in-network caching. In *2012 21st International conference on computer communications and networks (ICCCN)*, pages 1–7. IEEE, 2012.

[54] Le Liang, Hao Ye, and Geoffrey Ye Li. Toward intelligent vehicular networks: A machine learning framework. *IEEE Internet of Things Journal*, 6(1):124–135, 2018.

[55] Le Liang, Hao Ye, Guanding Yu, and Geoffrey Ye Li. Deep-learning-based wireless resource allocation with application to vehicular networks. *Proceedings of the IEEE*, 108(2):341–356, 2019.

[56] Yuxin Mao, Chenqian Zhou, Yun Ling, and Jaime Lloret. An optimized probabilistic delay tolerant network (dtn) routing protocol based on scheduling mechanism for internet of things (iot). *Sensors*, 19(2):243, 2019.

[57] Anum Mushtaq, Irfan Ul Haq, Muhammad Usman Imtiaz, Asifullah Khan, and Omair Shafiq. Traffic flow management of autonomous vehicles using deep reinforcement learning and smart rerouting. *IEEE Access*, 9:51005–51019, 2021.

[58] Anum Mushtaq, Asifullah Khan, Omair Shafiq, et al. Traffic flow management of autonomous vehicles using platooning and collision avoidance strategies. *Electronics*, 10(10):1221, 2021.

[59] M NoroozOliaee and Bechir Hamdaoui. Distributed resource and service management for large-scale dynamic spectrum access systems through coordinated learning. In *2011 7th International Wireless Communications and Mobile Computing Conference*, pages 522–527. IEEE, 2011.

[60] MohammadJavad NoroozOliaee, Bechir Hamdaoui, Mohsen Guizani, and Mahdi Ben Ghorbel. Online multi-resource scheduling for minimum task completion time in cloud servers. In *2014 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 375–379. IEEE, 2014.

[61] Hervé Ntareme, Marco Zennaro, and Björn Pehrson. Delay tolerant network on smartphones: Applications for communication challenged areas. In *Proceedings of the 3rd Extreme Conference on Communication: The Amazon Expedition*, pages 1–6, 2011.

[62] Chunyi Peng, Minkyong Kim, Zhe Zhang, and Hui Lei. Vdn: Virtual machine image distribution network for cloud data centers. In *2012 Proceedings IEEE INFOCOM*, pages 181–189. IEEE, 2012.

[63] Federico Poli. *Vehicular communications: from DSRC to Cellular V2X*. PhD thesis, Politecnico di Torino, 2018.

[64] Junaid Shuja, Abdullah Gani, Shahaboddin Shamshirband, Raja Wasim Ahmad, and Kashif Bilal. Sustainable cloud data centers: a survey of enabling techniques and technologies. *Renewable and Sustainable Energy Reviews*, 62:195–214, 2016.

[65] Francesco Sicuro. *EEBL safety application performance evaluation in V2V networks*. PhD thesis, Politecnico di Torino, 2018.

[66] Lion Silva, Naercio Magaia, Breno Sousa, Anna Kobusińska, António Casimiro, Constandinos X Mavromoustakis, George Mastorakis, and Victor Hugo C De Albuquerque. Computing paradigms in emerging vehicular environments: A review. *IEEE/CAA Journal of Automatica Sinica*, 8(3):491–511, 2021.

[67] Hassan Sinky and Bechir Hamdaoui. Cloudlet-aware mobile content delivery in wireless urban communication networks. In *2016 IEEE global communications conference (GLOBECOM)*, pages 1–7. IEEE, 2016.

[68] Hassan Sinky, Bassem Khalfi, Bechir Hamdaoui, and Ammar Rayes. Responsive content-centric delivery in large urban communication networks: A linknyc use-case. *IEEE Transactions on Wireless Communications*, 17(3):1688–1699, 2017.

[69] Majid Alkaee Taleghan and Bechir Hamdaoui. Efficiency-revenue optimality tradeoffs in dynamic spectrum allocation. In *2010 IEEE Global Telecommunications Conference GLOBECOM 2010*, pages 1–5. IEEE, 2010.

[70] Mim Kemal Tekin. Vehicle path prediction using recurrent neural network, 2020.

[71] Nguyen Ti Ti and Long Bao Le. Computation offloading leveraging computing resources from edge cloud and mobile peers. In *2017 IEEE International Conference on Communications (ICC)*, pages 1–6. IEEE, 2017.

[72] Chittibabu Tirupathi, Bechir Hamdaoui, and Ammar Rayes. Hybridcache: Ai-assisted cloud-ran caching with reduced in-network content redundancy. In *GLOBECOM 2020-2020 IEEE Global Communications Conference*, pages 1–6. IEEE, 2020.

[73] Asanga Udugama, Behruz Khalilov, Anas Bin Muslim, and Anna Förster. Implementation of the swim mobility model in omnet++. *arXiv preprint arXiv:1609.05199*, 2016.

[74] Pavithra Venkatraman and Bechir Hamdaoui. Cooperative q-learning for multiple secondary users in dynamic spectrum access. In *2011 7th International Wireless Communications and Mobile Computing Conference*, pages 238–242. IEEE, 2011.

[75] Pavithra Venkatraman, Bechir Hamdaoui, and Mohsen Guizani. Opportunistic bandwidth sharing through reinforcement learning. *IEEE Transactions on Vehicular Technology*, 59(6):3148–3153, 2010.

[76] Jianyu Wang, Jianli Pan, Flavio Esposito, Prasad Calyam, Zhicheng Yang, and Prasant Mohapatra. Edge cloud offloading algorithms: Issues, methods, and perspectives. *ACM Computing Surveys (CSUR)*, 52(1):1–23, 2019.

[77] Pin Wang, Ching-Yao Chan, and Arnaud de La Fortelle. A reinforcement learning based approach for automated lane change maneuvers. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 1379–1384. IEEE, 2018.

[78] Haiyong Xie, Guangyu Shi, and Pengwei Wang. Tecc: Towards collaborative in-network caching guided by traffic engineering. In *2012 Proceedings IEEE INFOCOM*, pages 2546–2550. IEEE, 2012.

[79] Yiping Xing, Rajarathnam Chandramouli, Stefan Mangold, et al. Dynamic spectrum access in open spectrum wireless networks. *IEEE journal on selected areas in communications*, 24(3):626–637, 2006.

[80] Danlei Yu and Young-Bae Ko. Ffrdv: fastest-ferry routing in dtn-enabled vehicular ad hoc networks. In *2009 11th International Conference on Advanced Communication Technology*, volume 2, pages 1410–1414. IEEE, 2009.

[81] Qing Zhao and Ananthram Swami. A survey of dynamic spectrum access: Signal processing and networking perspectives. In *2007 IEEE International Conference*

on Acoustics, Speech and Signal Processing-ICASSP'07, volume 4, pages IV–1349. IEEE, 2007.