Tuijin Jishu/Journal of Propulsion Technology
ISSN: 1001-4055
Vol. 45 No. 2(2024)

Utilizing Hybrid Deep Reinforcement Learning Approaches for Congestion Control in Vehicular Ad-Hoc Networks (VANETs)

[1]Santosh Kumar Maharana*, [2]Prashanta Kumar Patra
[1] *Biju Patnaik University of Technology, Rourkela, Odisha, India.*
[2] *SOA University, Bhubaneswar, Odisha, India.*

Abstract: — In the field of wireless communication, Vehicular Ad-Hoc Networks (VANETs) have recently garnered a lot of attention. Through the use of Vehicular Ad Hoc Networks (VANETs), automobiles are able to share vital information like location, direction, and speed with roadside equipment. However, traffic congestion is a common result of several vehicles using popular routes at once, therefore effective strategies for anticipating and avoiding congestion are necessary. The authors of this study propose enhancing traffic congestion forecasting capabilities in VANETs via the use of Hybrid Deep Reinforcement Learning (HDRL). This approach makes use of state-of-the-art deep learning methods including O-BiLSTM, Fuzzy Interface Systems, and Recurrent Capsule Networks (CapsRNN). For testing purposes before incorporating them into the deep reinforcement learning model, the SUMO platform is used to mimic traffic conditions. The methods of simulation and the approaches to programming are described in detail. Two critical metrics—mean travel time delay and mean vehicle waiting time delay—form the basis of the evaluation of the proposed method. Our method shows that these measures may be improved over time by running several simulations that take environmental input into account. The outcomes achieved with and without the integration of the CapsRNN, FIS, and O-BiLSTM algorithms are compared in a comparative study. Particularly in cases with moderate traffic density, the findings show that the deep reinforcement learning model is effective. Also, the article compares the algorithms' abilities to forecast traffic jams and concludes that CapsRNN is superior than FIS and O-BiLSTM.

Keywords: VANET, traffic congestion, neural network, HDRL, SUMO, CapsRNN, FIS, O-Bidirectional LSTM

1. Introduction

Vehicular Ad-Hoc Networks (VANETs) are a kind of mobile ad-hoc networks (MANETs) that enable intelligent communication among vehicles (V2V) and between vehicles and infrastructure (V2I). Exploring automotive technology is the first stride towards a future dominated by autonomous vehicles, providing more ease for drivers. Furthermore, VANETs provide a multitude of potential advantages, including improved transportation safety, alleviation of traffic congestion, augmentation of road capacity, and diminishment of vehicle size and parking space requirements.

Owing to the exponential growth in the quantity of automobiles traversing the roadways, traffic congestion has become an unavoidable predicament, resulting in significant squandering of time and resources. The current research in Vehicular Ad-Hoc Networks (VANETs) has focused on exploiting VANETs to address the pressing traffic problem, making it a complex and fascinating area of study. However, VANETs include intrinsic limitations and a lack of flexibility in efficiently identifying congested routes and redirecting vehicles to other channels. For instance, congestion may arise abruptly and then dissipate rapidly, making rerouting unnecessary. However, there are instances when traffic congestion occurs after a vehicle has already selected a certain route, leaving no other practical alternatives. Relying only on conventional VANET techniques may be inadequate in very dynamic scenarios. Moreover, there is a pervasive expectation in the transportation industry for groundbreaking advancements, aiming to enhance safety, efficiency, and reliability on a global scale.

In order to tackle these problems and limitations, it is essential to develop a robust traffic prediction model. Rather of just relocating vehicles when a congestion location is detected, it is beneficial to consistently assess and predict if the congestion persists as a vehicle approaches that place. This method mitigates abrupt fluctuations in traffic patterns that arise after a vehicle has already been redirected. Developing a powerful algorithm that can accurately tackle this intricate issue is essential for accurate and efficient traffic prediction.

This research proposes the use of a machine learning technique known as hybrid deep reinforcement learning to address this challenge. This approach combines the techniques of deep learning and reinforcement learning (RL). Deep learning is a significant advancement in artificial intelligence (AI) that use several layers to extract intricate information and uncover patterns that may be hard for humans to see. On the other hand, reinforcement learning has the capability to autonomously acquire knowledge and improve itself by using past mistakes or encounters. Hybrid deep reinforcement learning offers a strong foundation for tackling complex decision-making issues that were previously beyond the reach of computers. This system not only demonstrates exceptional proficiency in gaining the necessary abilities for doing intricate tasks, but it also significantly improves performance in several areas via a multitude of iterations. Deep reinforcement learning is gaining recognition as an effective method for addressing the inherent challenges of traffic prediction models, with the potential to provide successful solutions. Moreover, given the substantial impact of transportation on the general standard of daily existence, the use of these contemporary technologies is becoming imperative.

The efficacy of our proposed technique is shown by simulations conducted using the SUMO traffic simulator, which faithfully replicates actual traffic circumstances. To optimize the efficiency of our prediction model, we iteratively execute the simulation, using input data from the generated traffic scenario. Within the context of deep reinforcement learning, the agent is an essential component that plays a pivotal role in making choices based on the rewards it gets. This study entails simulating and comparing three distinct neural network algorithms (CapsRNN, FIS, and O-Bidirectional LSTM) to ascertain the most suitable one for our model. It is well acknowledged that increasing the number of simulations enhances the accuracy of identifying congested routes. Consequently, any vehicle has the ability to effectively evade traffic congestion, thereby decreasing the amount of time spent traveling. The evaluation of the effectiveness of our approach relies on two critical metrics: the average transit time delay and the average waiting time delay. The simulation results unequivocally demonstrate that our proposed technique significantly reduces both the mean transit time delay and the mean waiting time delay.

2. Related Works

Recently, a multitude of varied research have been conducted to improve urban transportation [4], [14], [16], [19], [20], [21], [25], [26], [27], [29]. This section explores the relevant research on Vehicular Ad Hoc Networks (VANET), as well as the use of traffic simulators and deep reinforcement learning for VANET.

Over the last several years, there has been a significant emphasis placed on VANET routing protocols. This is being done for a variety of reasons, including the reduction of traffic congestion, the reduction of accidents, and the enhancement of the efficiency and reliability of packet delivery. The simulation described in [20] is used to assess the performance of topology-based routing protocols such as AODV, AOMDV, DSDV, and DSR. With the lowest average time delay and the highest average throughput, it is evident that DSR is the most effective method. In spite of this, DSDV is superior than the other approaches in terms of the percentage of packets that are delivered.

An investigation of hybrid deep learning algorithms for the goal of congestion detection and control is carried out in [1], which compares and contrasts these aforementioned methodologies. The information on the patterns of traffic is acquired, and the vehicles are sent along the routes that are the most efficient. A total of around 8,000 manual counts were carried out on the roadway over the course of a period of twelve hours for the purpose of this investigation. We acquired the data with the use of Kaggle. The forecasting of traffic bottlenecks is accomplished by the use of a hybrid deep learning approach.

The paper [4] examines the efficacy of position-based routing algorithms in dynamic environments. Each protocol has an own set of parameters that determines the necessary amount of support and performance. For example, some protocols like GSR often need supplementary infrastructure, such as street maps. Conversely, GPSR and B-MFR are appropriate for interstate travel, but other protocols are more suitable for urban environments. Both IGRP and RBVT exhibit exceptional delivery rates, representing a significant improvement.

Paper [11] introduces a novel position-based routing algorithm known as the Efficient Routing Algorithm (ERA). The system uses predictions of future vehicle locations inside the SDN Internet of Vehicles to generate the most optimal and dependable routes. ERA has superior packet delivery rates when compared to protocols like GPSR and AODV.

Recent research have extensively examined VANET, namely its integration with deep learning techniques. The research undertaken by [29] investigates the use of VANET and artificial neural networks in autonomous cars to enhance security measures against malicious attacks such as denial of service (DOS), impersonation, and jamming. Deep learning improves the security of VANETs, hence reducing the likelihood of collisions and malicious attacks.

Deep learning algorithms are used for traffic prediction in VANET [11], [23]. The work [11] introduces a prognostic model that employs real-time data, such as speed, direction, and velocity, to approximate the positions of vehicles in dynamic scenarios. The ERA improves communication by offering dependable pathways for the delivery of data packets. The research [23] focuses on using deep convolutional neural networks to predict real-time traffic flow in metropolitan regions.

Concurrently, researchers are now studying the use of deep learning to mitigate traffic congestion [19]. The SUMO software is used to simulate various traffic scenarios, allowing vehicles to adjust their routes in response to congestion encountered on their journeys to their destinations.

Deep reinforcement learning, a combination of artificial neural networks and reinforcement learning, is widely acknowledged as a strong and reliable framework [9], [30]. It enables agents to learn the most optimal behaviors in complicated situations by associating present actions with future rewards. Previous studies [3], [24] have examined the use of this technology in developing intelligent agents that are deployed at Road Side Units (RSUs) to ensure Quality of Service (QoS) in vehicular networks.

Later on, [12] proposes the use of reinforcement learning to enhance vehicle trajectories in SUMO. This study demonstrates the use of reinforcement learning models to teach automobiles to independently navigate towards destinations while avoiding congested roads. Meanwhile, a study examines the use of deep reinforcement learning to control traffic lights in large grid networks. This technique employs a signal-control system that incorporates a diverse range of various actions. As a result, it effectively reduces queue lengths and waiting times for automobiles during periods of traffic congestion.

As deep reinforcement learning algorithms progress in handling more intricate tasks, it is expected that they will improve their dependability in uncertain real-world situations and exhibit the capacity to investigate a diverse array of potential actions. This accomplishment demonstrates the capacity to generate significant breakthroughs in several domains, hence paving the way for more effective artificial intelligence solutions.

3. Methods

a. Proposed Algorithm

This section provides a detailed description and visualization, in the form of a flowchart diagram (Figure 1), of our proposed method. The technique is designed to forecast traffic patterns in a Vehicular Ad-Hoc Network (VANET) utilizing a hybrid deep reinforcement learning model.

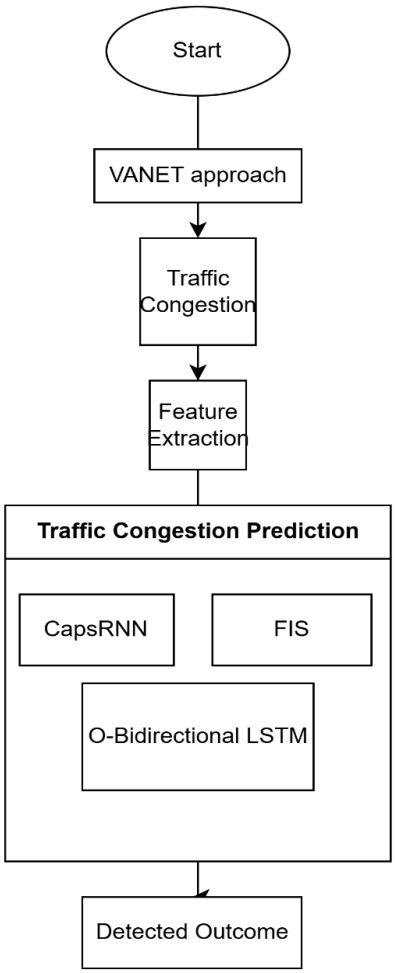

Figure1.Flowchart Diagram of our Proposed Algorithm

Algorithm

The many stages included in the procedure are:

Step 1: (data collecting)

The data is obtained from OpenStreetMap via the process of traffic scenario generation.

Step 2: (feature extraction)

Data obtained is used to extract statistical features, higher order statistical features, correlation-based features, and database features.

Step 3: (forecast of traffic congestion)

The novel hybrid deep learning technique, which incorporates Recurrent capsule networks (CapsRNN), Fuzzy Interface System (FIS), and Optimized Bi-LSTM (O-Bidirectional Long short Memory), is used to anticipate traffic congestion.

Step 4: Simulation

The TRACI interface is used by SUMO to simulate real-world traffic.

3.2 Data Collection: Traffic Scenario Creation from OpenStreetMap

SUMO (Simulation of Urban Mobility) is used to simulate and manage large road networks in order to recreate road traffic situations. First, a designated region on an actual map is selected and imported into SUMO to create the traffic scenario, which is then combined with a deep reinforcement learning model.

This project employs the use of Open Street Map (OSM), a digitized street map, to precisely identify the boundaries and intersections of roads, resulting in very accurate results. OSM, a globally collaborative open-source platform, is created by a community of mappers to create street-level maps. Users have the ability to search for and choose specific regions, as well as different street features like lanes, primary roads, highways, sidewalks, railroads, underground passages, traffic signals, bridges, and junctions. In order to streamline the simulation process, the city map, which includes the main roadways, is extracted from OSM and converted into a map format. An OSM file is a text file encoded in XML format that contains information on nodes, streets, and tags (object properties).

Figure 2 presents a flowchart that shows the process of generating SUMO configuration files using the city map obtained from OSM. The process of creating a map may be outlined in a step-by-step manner as follows:

- The map.osm file from Open Street Map was imported for the selected geographic region.
- The map.net.xml file was generated using the NETCONVERT command-line program.
- The route file map.rou.xml was generated using the randomTrips.py Python tool.
- The PLOYCONVERT command-line application was then used to produce the map.poly.xml file, integrating further data from typemap.xml.
- Afterwards, the SUMO configuration file (map.sumocfg) was created using the previously stated files.
- The SUMO configuration file encapsulates the traffic scenario that has been developed.

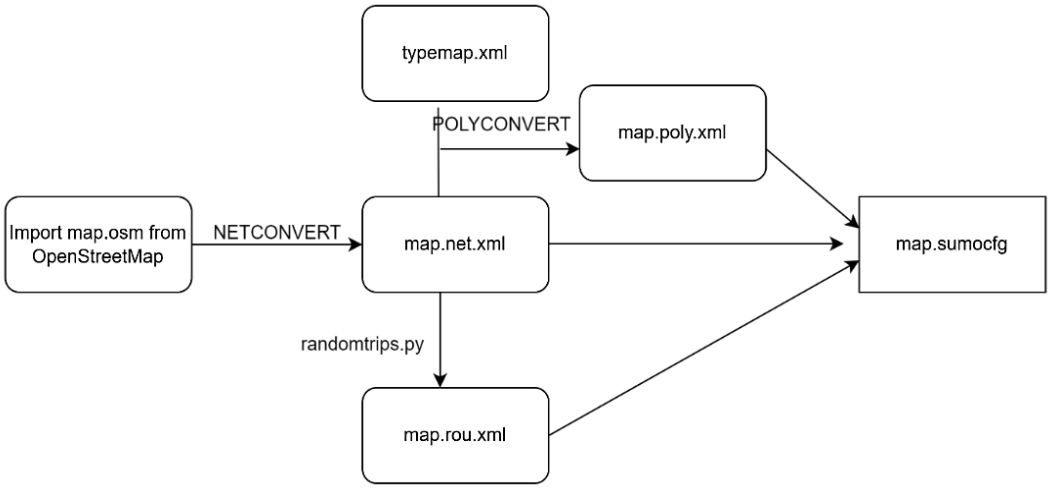

Figure 2. Creating SUMO configuration file for real world traffic

The SUMO simulator often uses a default routing method called DuaRouter, which is a specialized functionality inside SUMO that creates a vehicle-routing file (map.rou.xml). In order to use DuaRouter, it is necessary to create a road network (map.net.xml) and define traffic needs, such as trip and flow definitions. The demand definitions, obtained from the origin and destination edges, assist in calculating the most efficient path for dynamic user assignment (DUA). However, when using deep reinforcement learning with SUMO, the default routing approach is replaced with a superior and smarter strategy acquired by the agent via the deep reinforcement learning algorithm.

By using the TraCI library, the SUMO simulator may be controlled, and establishing a connection to this library facilitates the implementation of a deep learning technique [27]. Python is used as the programming language to control both TraCI and deep reinforcement learning in this project. PyCharm, an adequate integrated development

environment (IDE) for Python programming, enables the seamless integration of the deep reinforcement learning model into SUMO.

3.3 Statistical Feature Extraction

Statistical features

The four statistical features considered in this study for each dataset are mean or average, standard deviation, skewness and kurtosis.

Mean or average

The mean or average ($\bar{X}$) value of a collection of N numbers ($X1,X2\ldots XN$) may be quantified using the following formula.

$$\bar{X} = \frac{\sum_{i=1}^{N} Xi}{N} \quad (1)$$

Standard deviation

The standard deviation is a commonly used descriptive statistical measure that indicates the spread of data points around the mean. It represents the deviation of each data point from the average value. A small standard deviation in a dataset suggests that the data points are tightly grouped around the mean value. On the other hand, a large standard deviation indicates that the data points are dispersed across a broader range. The formula for calculating the standard deviation (SD) of a dataset of size $N(X1,X2\ldots XN)$ with a mean of $\bar{X}$ is as follows.

$$SD = \sqrt{\frac{\sum_{i=1}^{N}(Xi-\bar{X})^2}{N-1}} \quad (2)$$

Skewness

Skewness is the degree to which a distribution deviates from a perfectly symmetrical normal distribution. The skewness value might be positive, zero, or negative (Fig. 2). In datasets with negative skewness, the distribution has a longer left tail, whereas in datasets with positive skewness, the right tail is more extended. In the case of data that is not skewed, both tails exhibit symmetry. The provided formula may evaluate the skewness of a given dataset ($X1,X2\ldots XN$)

$$\mu_3 = \frac{\sum_{i=1}^{N}(Xi-\bar{X})^3}{(N-1)XSD^3} \quad (3)$$

Kurtosis

Kurtosis measures the extent to which the probability distribution of a real-valued random variable deviates from a normal distribution in terms of tail weight. It aids in determining if the data have heavy-tailed or light-tailed properties in comparison to a normal distribution. The formula below quantifies the kurtosis ($\beta 2$) of a dataset ($X1,X2\ldots XN$) with a mean and standard deviation of $\bar{X}$ and SD, respectively.

$$\beta_2 = \left\{\frac{N(N+1)}{(N-1)(N-2)(N-3)}\sum_{i=1}^{N}\left(\frac{X_i-\bar{X}}{SD}\right)^4\right\} - \frac{3(N-1)^2}{(N-2)(N-3)} \quad (4)$$

3.4 Traffic Congestion Detection:

When there is an occurrence of traffic congestion, a concise and clear procedure, as shown in Figure 4, takes place in the following manner: Upon joining the road, every car is allocated a random origin and destination. The vehicle's data is sent to a traffic congestion prediction model in the deep reinforcement learning module to forecast future traffic patterns and detect routes that are not crowded. If congestion is identified, cars are sent to alternate routes; otherwise, they continue on their originally chosen course. After all vehicles have arrived at their destinations, the average travel time delay and average waiting time delay are calculated.

At first, the software may not choose the best route, which would minimize both travel time delay and waiting time delay. Consequently, the program gets performed on several occasions. By repeatedly executing the program,

the model's ability to forecast crowded pathways and reduce travel time delay and waiting time delay is improved, since it keeps the data of actions conducted. This repeating procedure continues until the time delay ceases to decrease, signifying that it has attained its minimal value.

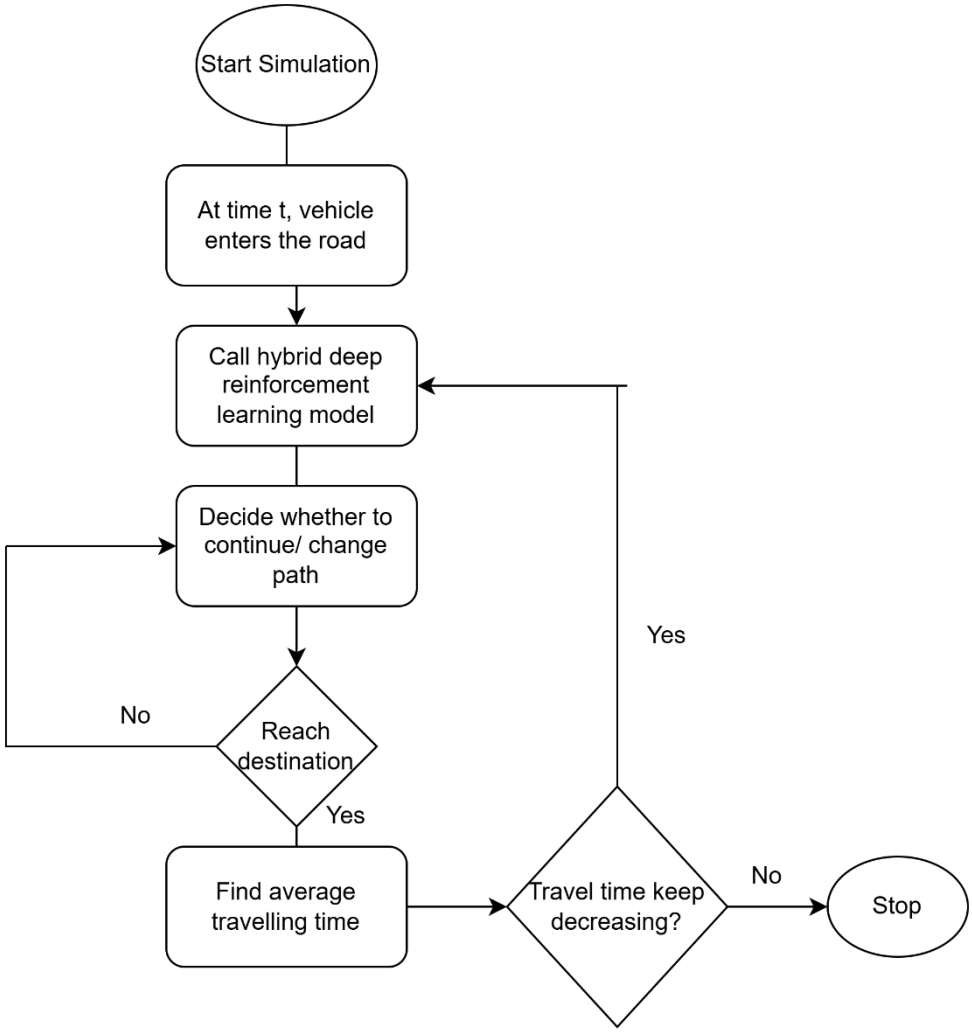

Figure3. Flowchart of traffic congestion prediction concept

3.4.1 HYBRID Deep Reinforcement Learning Implementation

This research is based on the premise that there is communication between vehicles and infrastructures (V2I), as well as between vehicles (V2V) when they are on the road. The transmission of road traffic data between automobiles and roadside equipment is often advantageous for deep reinforcement learning. Given that the scenario has been implemented in SUMO, it is possible to easily get traffic data.

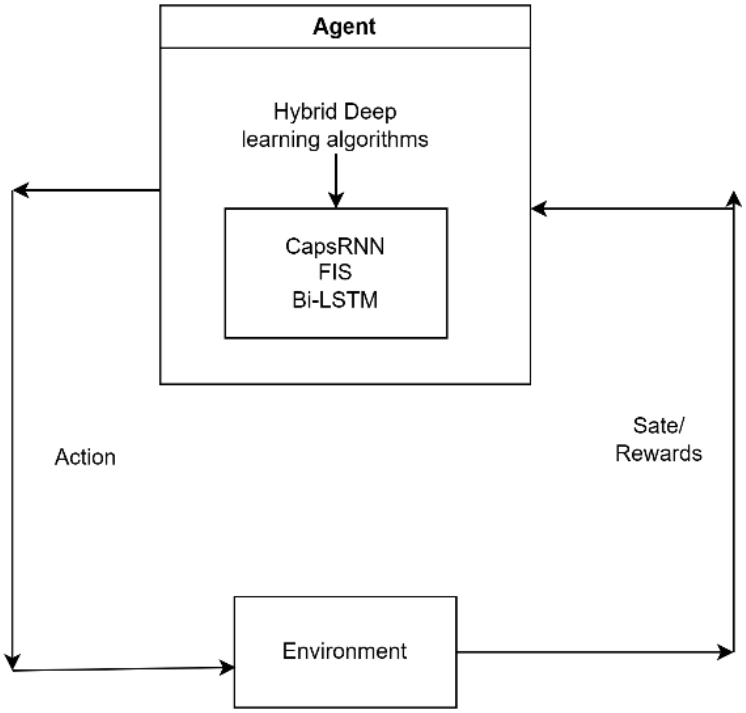

Figure 4. Hybrid Deep reinforcement learning

Fundamental ideas that are common to all deep reinforcement learning models include agents, environments, states, actions, and rewards. Figure 4 illustrates the process of deep reinforcement learning. The role of the environment component is to convert an action performed in a certain state into a new state and its related reward. Afterwards, an agent modifies these resulting values and subsequently converts them into the subsequent action. The agent strives to do behaviors in the environment in order to maximize the rewards. The environment refers to the specific road network built by SUMO in which the agent functions.

To create an environmental model for deep reinforcement learning, one must use the TraCI module in PyCharm to establish a connection with SUMO and gather road network data, such as edges, lanes, and nodes. After acquiring this data, NumPy, a library specifically built for performing advanced mathematical operations on arrays, is used to construct multidimensional arrays and matrices that accurately depict the coordinates of roads and junctions, as well as their corresponding velocities.

The agent's primary responsibility is to make judgments by taking into account incentives and observations. Figure 4 depicts the internal structure of the agent, which includes the construction of neural networks. These networks consist of hybrid deep learning algorithms, including CapsRNN, FIS, and Bi-LSTM.

3.4.1.1 SARSA-learning technique

The SARSA algorithm employs an On-Policy method, where the current action done from the presently used policy is used to learn the Q-value.

The update statements for the SARSA learning approach may be precisely specified as follows:

$$Q(s_t, a_t) = Q(s_t, a_t) + \alpha(r(s_t, a_t) + \gamma\, Q(s_{t+1}, a_{t+1}) - Q(s, a_t)) \quad (5)$$

In SARSA, the update relies on the current action, current state, received reward, next state, and next action, which are written as State Action Reward State Action (s, a, r, s', a').

3.4.1.2 Algorithms of Hybrid deep learning Used in the Proposed Model

Keras, an open-source neural network library, is used for creating neural network layers. The reward function is assessed by considering the aggregate count of vehicles and the count of cars that are waiting on certain roadways. The agent chooses acts that result in greater rewards. Using the TraCI library is crucial for running the program with the deep reinforcement learning model in SUMO. This research investigates and compares three different hybrid deep learning models in order to determine the most efficient one for analyzing traffic scenarios.

Recurrent capsule networks (CapsRNN)

The input layer, recurrent layer, and capsule layer are the three main parts.

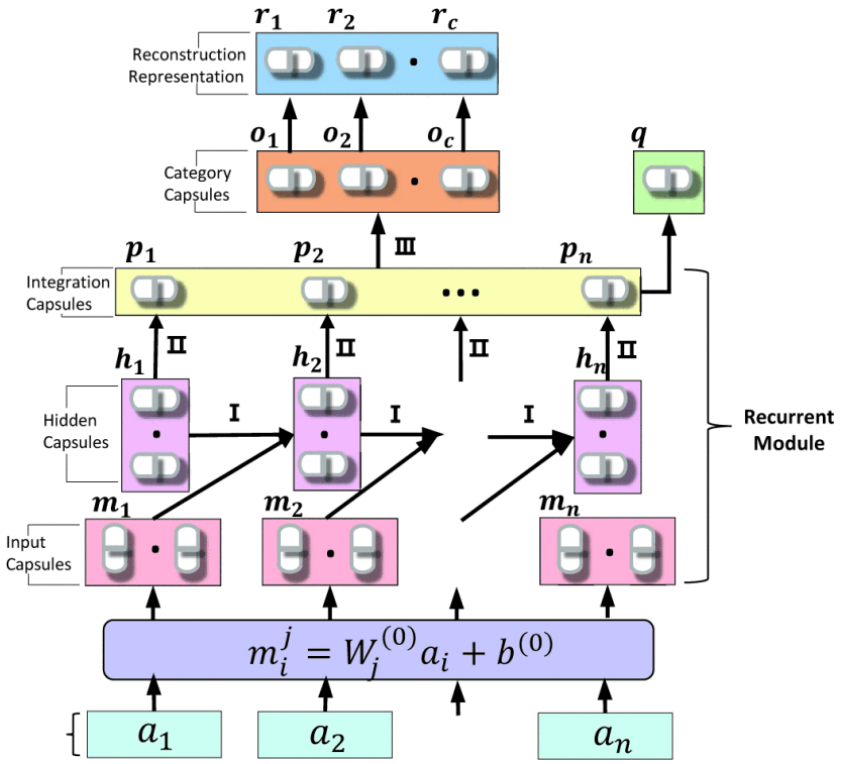

Fig. 5 CapsRNN architecture

Fuzzy Inference System (FIS):

A fuzzy logic system consists of a fuzzy inference system (FIS) that utilizes fuzzy set theory, IF-THEN rules, and fuzzy reasoning to calculate precise outputs that correspond to specific inputs. Notable features of FIS include: - Accurate acquisition of values from actions.

- The use of fuzzy membership functions to transform precise values into fuzzy values.
- The fuzzy outcomes are generated by applying IF-THEN rules from the fuzzy rule base.
- Development of algorithms to transform fuzzy outputs into accurate/definite values.

Functional Blocks in FIS:

The formation of FIS is made easier by five functional components:

1. The rule set is defined using fuzzy IF-THEN rules.
2. Membership functions define the degree of membership of fuzzy sets used in the rules.
3. Fuzzy rule application by the decision-making unit.
4. The fuzzification interface unit is responsible for transforming crisp values into fuzzy quantities.
5. The defuzzification interface unit converts fuzzy values into precise quantities.

Operation of FIS:

The functioning of the fuzzy inference system may be divided into the following steps:

1. The fuzzification unit facilitates the implementation of different fuzzification approaches for varied purposes.
2. Clear and precise inputs are transformed into imprecise inputs, creating a knowledge base derived from the rule-based database.
3. The defuzzification unit transforms imprecise inputs into precise outputs

Optimized Bi-LSTM (O-Bidirectional Long Short-Term Memory):

This is a powerful sequence model that is often used in tasks related to natural language processing (NLP). The Bi-LSTM model consists of two LSTM layers, where one layer processes the input in the forward direction and the second layer processes it in the backward way. Bi-LSTM improves the model's comprehension of sequence links by taking into account both previous and future contexts.

Architecture:

- The sequence is processed in both the forward and backward directions by two LSTMs that operate in a unidirectional fashion.

-A combination of probabilities derived from both forward and backward LSTMs is produced as the final output.

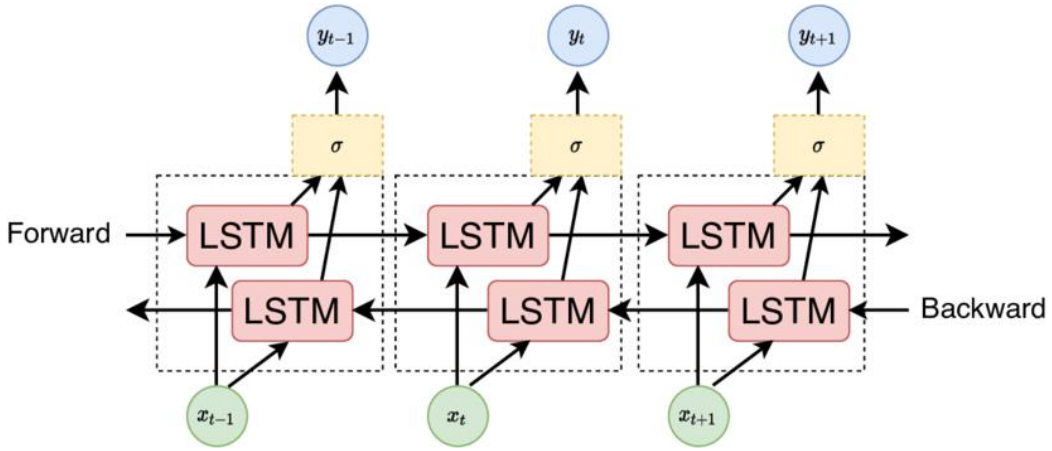

Fig. 6 Bi-LSTM Architecture

3.5 Using TraCI interface for Simulation

We construct the traffic simulation and the map.sumocfg file using the required procedures and command-line tools before we interface with TraCI. Listed below are the following steps:

1. TraCI.start() is how the SUMO simulator is started by the TraCI library.
2. Using the traci.vehicle.getIDList() and traci.edge.getIDList() methods, we can acquire the list of vehicle and edge IDs, respectively.
3. At certain points in time, the deep reinforcement learning model starts to recall the list vehicle's behaviors when it hits the road.
4. Traffic-clogged roadways choose whether to merge lanes or redirect to less congested areas. The traci.vehicle.rerouteTraveltime() and traci.vehicle.updateBestLanes() methods are used to reroute and change lanes, respectively, in this stage.
5. The vehicles then use the list of edge IDs acquired in the previous stage to choose new routes or lanes, also known as edges. Traffic simulation ends after all cars have finished making judgments and have arrived at their destinations.
6. When cars reach the next edges, the traci.edge.getTraveltime() and traci.edge.getWaitingTime() functions provide the journey time and waiting time, respectively. The average travel time and waiting time are determined after the simulation ends.

7. The deep reinforcement learning model assesses the vehicle's choices using the SARSA-Learning algorithm and learns from experience based on the reward values before the SUMO simulation ends. You may get the total number of driving cars and waiting vehicles using the traci.edge.getLastStepVehicleNumber() and traci.edge.getLastStepHaltingNumber() methods, respectively (see section 4.3 for more information).
8. The final program run count is 15, which is enough to ensure the model operates properly and stabilizes the waiting and travel durations. The procedure starts again from the beginning if the number of programs executed is fewer than 15.

3.5.1 Simulation Parameters

Table 1 Simulation Parameters

Parameters	Environment
Traffic Simulator	SUMOv1.2.0
Geographical Information System (GIS)	OpenStreetMap (OSM)
Deep Learning Module	TensorFlowv2.0
Computer Language	Pythonv3.12
Integrated Development Environment (IDE)	PyCharm

4. Results

In this section, the simulation results are illustrated with 96% confidence interval.

a. Average Travelling Time Delay

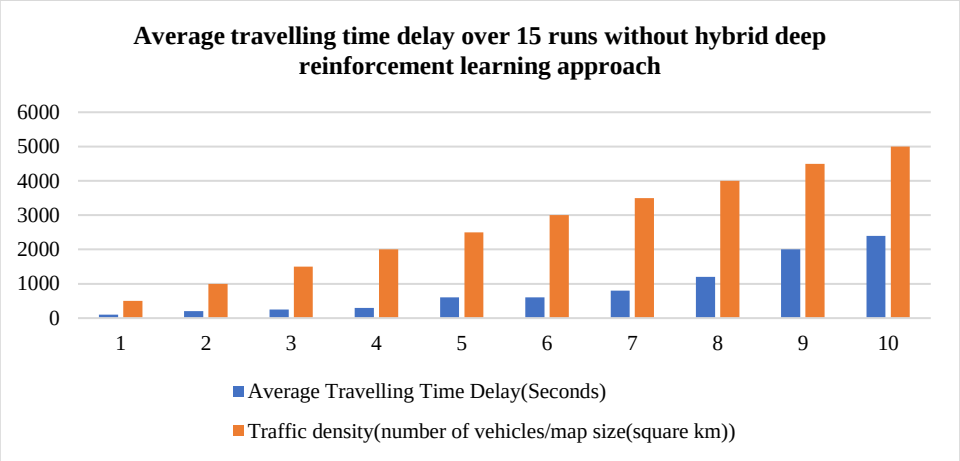

Fig 7. Average Travelling Time Delay of algorithms without hybrid deep reinforcement learning approach over 15 runs

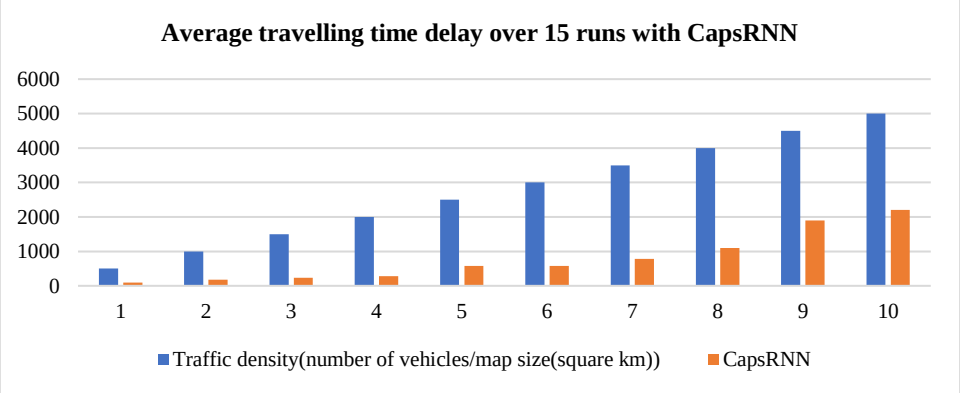

Fig. 8. Travelling Time Delay of algorithms with Caps RNN over 15 runs

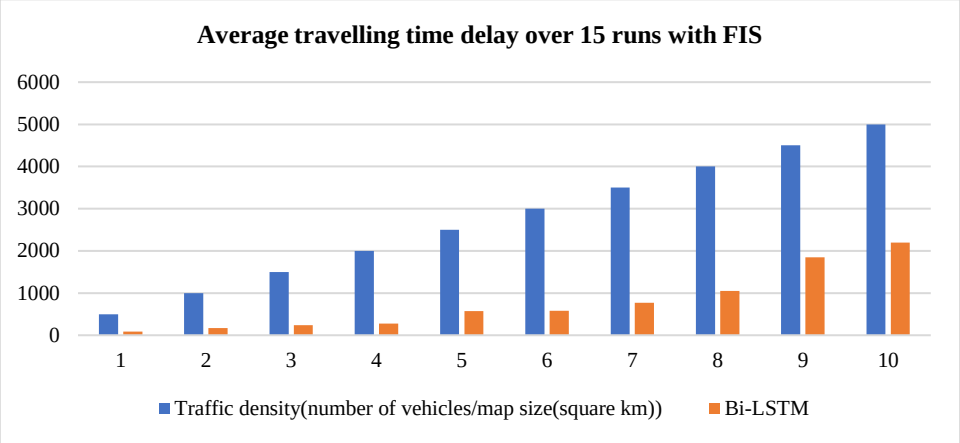

Fig. 9 Travelling Time Delay of algorithms with FIS over 15 runs

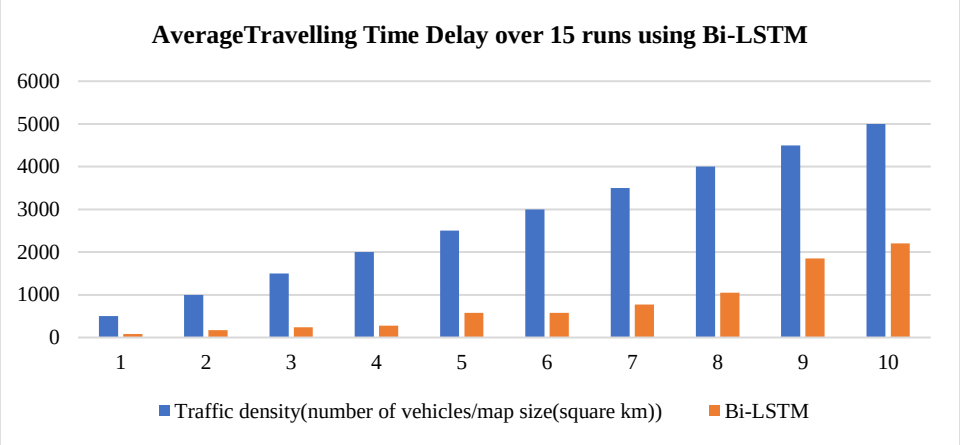

Fig.10 Travelling Time Delay of algorithms with FIS over 15 runs

At traffic densities of 3,500 to 4,500 vehicles/km2, the deep reinforcement learning model seems to perform better. The consistent pattern with the average journey time delay is readily apparent in the data.

5. Conclusions and Future Works

This research uses VANET-based deep reinforcement learning to forecast traffic jams. Through vehicle to-infrastructure (V2I) and vehicle-to-vehicle (V2V) communication, VANET allows machines to study traffic data such as vehicle velocities, positions, directions, and patterns of movement in order to figure out how to decrease congestion. The goal of deep reinforcement learning is to solve difficult problems by learning from experience and interacting with real-world data. A SUMO-based traffic simulation must be established prior to building a deep reinforcement learning model. The SUMO road network is based on real-world road layouts obtained by importing regions from OpenStreetMap (OSM). Afterwards, the paths that cars will take and other map elements, such traffic signals, are established. After then, the SUMO configuration file takes charge of the traffic simulation, which is subsequently interfaced with the Python programming language and SUMO's TraCI library to combine deep learning methods.

The average travel time delay and waiting time delay both increase with repeated runs of this traffic prediction model as cars learn to choose routes based on incentives from earlier runs. More specifically, when traffic density is modest, the deep reinforcement learning model is shown to be beneficial when comparing simulation results with and without three distinct deep learning algorithms: CapsRNN, FIS, and Bi-LSTM.

More situations, like accidents and barricades, which can cause congestion, may be included to the traffic prediction model in future work to make it more efficient and accurate. To make the model more realistic and

useful, it may be enhanced by adding vehicle-to-pedestrian (V2P) communication and other kinds of vehicles, such buses and motorbikes.

Refrences

[1] Ahamed VMN, Prakash A, Ziyath M (2023) TCC-HDL: A Hybrid Deep Learning Based Traffic Congestion Control System for VANET. Indian Journal of Science and Technology 16(32): 2548-2559. https://doi.org/10.17485/IJST/v16i32.1319.

[2] Akhter S., Ahsan, M. N., Quaderi, S. J. S., Forhad, M. A. A., Sumit, S. H., & Rahman M. R. (2020) "A SUMO Based Simulation Framework for Intelligent Traffic Management System", *Journal of Traffic and Logistics Engineering*, Vol. 8, No. 1, pp.1-5

[3] Atallah, R., Assi, C., & Khabbaz, M. (2017) "Deep reinforcement learning-based scheduling for roadside communication networks*", The 15th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt 2017)*, Paris, France.

[4] Batish, S., Chahal, M., &Sofat, S. (2015) "Comparative studyof position-based routing protocols in VANET*", Asian Research Publishing Network* (*ARPN*), Vol.10, No.15, pp. 6414–6418.

[5] Choi R. Y., Coyner, A. S., Kalpathy-Cramer, J., Chiang, M. F., & Campbell, J. P. (2020) "Introduction to Machine Learning, Neural Networks, and Deep Learning", *Translational Vision Science & Technology (TVST)*, Vol. 9, No. 2, pp.1-12.

[6] Choudhary, A. (2019) "A Hands-On Introduction to Deep Q-Learning using OpenAI Gym in Python*"*,https://www.analyticsvidhya.com/blog/2019/04/introduction-deep-q-learning-python/.

[7] Contributors, OpenStreetMap "OpenStreetMap", https://www.openstreetmap.org/. GeofabrikGmbH: Karlsruhe, Germany.

[8] Contributors, SUMO Simulator "SUMOUserDocumentation",https://sumo.dlr.de/docs/.

[9] Fran¸cois-Lavet, V., Henderson, P., Islam, R., Bellemare, M. G., & Pineau, J. (2018) "Anintroduction to deep reinforcement learning", *Foundations and Trends in Machine Learning, Cornell University*, Vol.11, No. 3-4.

[10] Huang, S., (2020) "What is Big O Notation Explained: Space and Time Complexity", https://www.freecodecamp.org/news/big-o-notation-why-it-matters-and-why-it-doesnt.

[11] Jibran, M. A., Abbass, M. T., Rafiq, A., & Song, W.-C. (2020) "Position prediction for routing in software defined internet of vehicles", *Journal of Communications*, Vol.15, No.20, pp.157-163.

[12] Koh, S. S., Zhou, B., Yang, P., Yang, Z., Fang, H., & Feng, J. (2018) "Reinforcement learning for vehicle route optimization in SUMO*", The 2018 IEEE 20th International Conference on High Performance Computing and Communications; IEEE 16th International Conference on Smart City; IEEE 4th International Conference on Data Science and Systems, Exeter, United Kingdom*, pp.1468- 1473.

[13] Krajzewicz, D., Erdmann, J., Behrisch, M., & Bieker, L. (2012) "Recent development and applications of SUMO - Simulation of Urban MObility", *International Journal on Advances in Systems and Measurements*, Vo.5, No. 3 & 4, pp.128-138.

[14] Kumar, R., & Dave, M. (2011) "AcomparativestudyofvariousroutingprotocolsinVANET" *InternationalJournalofComputerScience Issues (IJCSI)*, Vol.8, Issue4, No.1, pp.643-648.

[15] Kwong, Y., Bolong, N., Kiring, A., Yang, S. S., Tze, K., & Teo, K. (2011) "Q-learning based traffic optimization in management of signal timing plan", *International Journal of Simulation: Systems, Science and Technology (IJSSST)*, Vol.12, No. 3, pp.29-35.

[16] Liang, W., Li, Z., Zhang, H., Wang, S., &Bie, R. (2015)"Vehicularadhocnetworks: Architectures, research issues, methodologies, challenges, and trends",*International Journal of Distributed Sensor Networks*, Vol. 11, No. 8, pp.1-11.

[17] Lopez, P. A., Behrisch, M., Bieker-Walz, L., Erdmann J., Flötteröd, Y., Hilbrich, R., Lücken, L., Rummel, J., Wagner, P., & Wießner, E. (2018) "Microscopic Traffic Simulation using SUMO", *International Conference on Intelligent Transportation Systems (ITSC)*, Maui, Hawaii, USA,pp.2575-2582.

[18] Mehta, A. (2019) "A Comprehensive Guide to Types of Neural Networks", https://www.digitalvidya.com/blog/types-of-neural-networks/.

[19] Perez-Murueta, P., G´omez-Espinosa, A., Cardenas, C., & Gonzalez-Mendoza, M. J. (2019) "Deep learning system for vehicular re-routing and congestion avoidance", *Journal of Applied Sciences: special issue Artificial Intelligence Applications to Smart City and Smart Enterprise*, Vol.9, No. 13. pp.1-14

[20] Rakkesh, S., Weerasinghe, A., & Ranasinghe, R. (2016) "Adecentralized vehiclere-routing approach using vehicular ad-hoc network", *The 16th International Conference on Advances in ICT for Emerging Regions (ICTer2016)*, Negombo, Sri Lanka, pp.201-207.

[21] Sanwal, R., & Kumar, V. (2013 "Simulation based evaluation of proactive and reactive routing protocols in realistic vehicular network", *International Conference on Recent Trends in Computing and Communication Engineering - RTCCE 2013*, Hamirpur, India, pp. 127-131.

[22] Singh, S., Saraswat, A., &Yadav, S. (2019)"Traffic simulationintegrationusingSUMOsimulator", *International Journal of Scientific Research and Review*, Vol.7, No.2, pp.22–26.

[23] Sun, S., Wu H., & Xiang, L. (2020) "City-Wide Traffic Flow Forecasting Using a DeepConvolutional Neural Network", *Multidisciplinary Digital Publishing Institute (MDPI)*, Vol.20, No. 2, pp.1-15.

[24] Tan, T., Bao, F., Deng, Y., Jin, A., Dai, Q., & Wang, J. (2020) "Cooperative deep reinforcement learning for large-scale traffic grid signal control", *In IEEE Transactions on Cybernetics*, Vol.50, Issue 6, pp. 2687-2700.

[25] SaifAl-Sultan, G. Calandriello, Y. Mak, TracyAnn Kosa, Stephen Marsh, H. Kim, J. Grover, M. Gaur, C. Lochert, B. Scheuermann, C. Wewetzer, A. Luebke, L. Nassar, M. Kamel, F. Karray, Ramu Panayappan, S. A. Khayam, J.J. Haas (2015) "Security and privacy in vehicular ad hoc network (VANET): A survey*", The IJCA Proceeding on National Conference on Advances in Computing Communication and Application (ACCA2015)*, pp.1-3.

[26] Wang, S., Djahel, S., & McManis, J. (2015) "An adaptive and VANETs-based next road rerouting system for unexpected urban traffic congestion avoidance", *The IEEE Vehicular Networking Conference (VNC2015)*, Kyoto, Japan, pp.196-203.

[27] Wedel, J. W., Sch¨unemann, B., &Radusch, I.(2009) "V2X-based trafficcongestion recognition and avoidance*", The 10th International Symposium on Pervasive Systems, Algorithms, and Networks 2009*, Kaohsiung, Taiwan, pp.637-641.

[28] Wu, C., Parvate, K., Kheterpal, N., Dickstein, L., Mehta, A., Vinitsky, E., et al. (2017) "Framework for control and deep reinforcement learning in traffic", *The IEEE 20th International Conference on Intelligent Transportation Systems (ITSC2017)*, Yokohama, Japan, pp. 155-163.

[29] Ydenberg, A., Heir, N., & Gill, B. (2018) "Security, SDN, and VANET technology of driver-less cars", *The IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC2018)*, Las Vegas, NV, USA, pp.313-316.

[30] "A Beginner's Guide to Deep Reinforcement Learning", (n.d.) https://pathmind.com/wiki/deep-reinforcement-learning.

[31] Ngo TT, Huynh-The T, Kim DS. A novel VANETs-based traffic light scheduling scheme for greener planet and safer road intersections. *IEEE Access*. 2019; 7:22175–22185. Available from: https://doi.org/10.1109/ACCESS.2019.2891250.

[32] Jindal V, Bedi P. An improved hybrid ant particle optimization (IHAPO) algorithm for reducing travel time in VANETs. *Applied Soft Computing*. 2018; 64:526–535. Available from: https://doi.org/10.1016/j.asoc.2017.12.038.

[33] Ata A, Khan MA, Abbas S, Khan MS, Ahmad G. Adaptive IoT Empowered Smart Road Traffic Congestion Control System Using Supervised Machine Learning Algorithm. *The Computer Journal*. 2021;64(11):1672–1679. Available from: https://doi.org/10.1016/j.jksuci.2018.10.011.

[34] Zheng H, Chang W, Wu J. Traffic flow monitoring systems in smart cities: Coverage and distinguishability among vehicles. *Journal of Parallel and Distributed Computing*. 2019; 127:224–237. Available from: https://doi.org/10.1016/j.jpdc.2018.07.008.

[35] Khatri S, Vachhani H, Shah S, Bhatia J, Chaturvedi M, Tanwar S, et al. Machine learning models and techniques for VANET based traffic management: Implementation issues and challenges. *Peer-to-Peer Networking and Applications*. 2021;14(3):1778–1805.

[36] D'andrea E, Marcelloni F. Detection of traffic congestion and incidents from GPS trace analysis. *Expert Systems with Applications*. 2017; 73:43–56. Available from: https://doi.org/10.1016/j.eswa.2016.12.018.
[37] Jobaer S, Zhang Y, Hussain MAI, Ahmed F. UAV-Assisted Hybrid Scheme for Urban Road Safety Based on VANETs. *Electronics*. 2020;9(9):1499. Available from: https://doi.org/10.3390/electronics9091499.
[38] Rashid MM, Datta P. Performance Analysis of Vehicular Ad Hoc Network (VANET) Considering Different Scenarios of a City. *International Journal of Computer Applications*. 2017;(10):162. Available from: https://doi.org/10.5120/ijca2017913329.
[39] Lakshmanaprabu SK, Shankar K, Rani SS, Abdulhay E, Arunkumar N, Ramirez G, et al. An effect of big data technology with ant colony optimization -based routing in vehicular ad hoc networks: Towards smart cities. *Journal of Cleaner Production*. 2019; 217:584–593. Available from: https://doi.org/10. 1016/j.jclepro.2019.01.115.
[40] Rath M, Pati B, Pattanayak BK. Mobile agent-based improved traffic control system in VANET. 2019. Available from: https://link.springer.com/chapter/ 10.1007/978-981-10-8797-4_28.
[41] Rizwan A, Karras DA, Dighriri M, Kumar J, Dixit E, Jalali A, et al. Simulation of IoT-based Vehicular Ad Hoc Networks (VANETs) for Smart Traffic Management Systems. *Wireless Communications and Mobile Computing*. 2022; 2022:1–11. Available from: https://doi.org/10.1155/2022/3378558.
[42] Elhoseny M, Shankar K. Energy efficient optimal routing for communication in VANETs via clustering model. 2020. Available from: https://doi.org/10. 1007/978-3-030-22773-9.
[43] Qureshi KN, Abdullah AH, Kaiwartya O, Iqbal S, Butt RA, Bashir F. A Dynamic Congestion Control Scheme for safety applications in vehicular ad hoc networks. *Computers & Electrical Engineering*. 2018; 72:774–788. Available from: https://doi.org/10.1016/j.compeleceng.2017.12.015.
[44] Jain R. A congestion control system based on VANET for small length roads. 2018. Available from: https://doi.org/10.48550/arXiv.1801.06448.
[45] Liu X, Jaekel A. Congestion Control in V2V Safety Communication: Problem, Analysis, Approaches. *Electronics*. 2019;8(5):540. Available from: https://doi.org/10.3390/electronics8050540.
[46] Mohanty A, Mahapatra S, Bhanja U. Traffic congestion detection in a city using clustering techniques in VANETs. *Indonesian Journal of Electrical Engineering and Computer Science*. 2019;13(3):884. Available from: https://ijeecs.iaescore.com/index.php/IJEECS/article/view/13156.
[47] Sharma S, Pandey R. Accident detection, avoidance and prevention using intelligent transportation system. *International Journal of Computer Applications*. 2018;182(7):10–12. Available from: https://www.ijcaonline.org/archives/volume182/number7/sharma-2018-ijca-917639.pdf.
[48] Ravikumar K, Vishvaroobi T. Congestion control in vehicular ad hoc networks (VANET) using meta-heuristic techniques. *International Journal of Computer Science Trends and Technology*. 2017;5(4):66–72. Available from: http://www.ijcstjournal.org/volume-5/issue-4/IJCST-V5I4P12.pdf.
[49] Abdelatif S, Derdour M, Ghoualmi-Zine N, Marzak B. VANET: A novel service for predicting and disseminating vehicle traffic information. *International Journal of Communication Systems*. 2020;33(6):e4288. Available from: https://doi.org/10.1002/dac.4288.
[50] Mallah RA, Quintero A, Farooq B. Distributed Classification of Urban Congestion Using VANET. *IEEE Transactions on Intelligent Transportation Systems*. 2017;18(9):2435–2442. Available from: https://doi.org/10.48550/arXiv.1904.12685.
[51] Choe C, Ahn J, Choi J, Park D, Kim M, Ahn S. A Robust Channel Access Using Cooperative Reinforcement Learning for Congested Vehicular Networks. *IEEE Access*. 2020; 8:135540–135557. Available from: https://doi.org/10.1109/ACCESS.2020.3011568.
[52] Ullah A, Yaqoob S, Imran M, Ning H. Emergency Message Dissemination Schemes Based on Congestion Avoidance in VANET and Vehicular FoG Computing. *IEEE Access*. 2019; 7:1570–1585. Available from: https://doi.org/10.1109/ACCESS.2018.2887075.