Leveraging RL for Efficient Collection of Perception Messages in Vehicular Networks

Chaima Zoghlami, Rahim Kacimi, Riadh Dhaou

Leveraging RL for Efficient Collection of Perception Messages in Vehicular Networks

Chaima Zoghlami, Rahim Kacimi, and Riadh Dhaou
IRIT, Université de Toulouse, CNRS, Toulouse INP, UT3, Toulouse, France
{*chaima.zoghlami, rahim.kacimi, riadh.dhaou*}*@irit.fr*

***Abstract*—Cooperative messages play a vital role in vehicle-to-everything (V2X) applications by enhancing situational awareness, supporting collision avoidance and improving traffic efficiency. Additionally, they contribute to Vulnerable Road Users (VRU) safety by increasing environment perception. The purpose of this paper is to introduce a novel Q-Learning technique that can improve the selection of cooperative messages' type, size and frequency. The methodology is based on leveraging the diversity of existing messages in vehicular networks to determine the best message type with the appropriate size while adjusting its transmission frequency according to the environmental context in order to efficiently manage network resources. In addition to alleviating the network overload and decreasing the number of messages sent simultaneously, our method could result in significant energy savings when applied to VRUs when they are identified by Connected and or Autonomous Vehicles (CAV).**

***Index Terms*—V2X communications, Reinforcement Learning, VRU's safety, MEC.**

I. Introduction

Vehicular networks are an emerging field that integrates vehicles and communication technologies to enable seamless communication between cars as well as vulnerable road users (VRUs) and roadside infrastructure. Different communication messages exist including VRU Awareness Messages (VAM) for VRUs, Cooperative Awareness Messages (CAM), Decentralized Environmental Notification Messages (DENM) and Basic Safety Messages (BSM) for vehicles, and Collective Perception Messages (CPM) for vehicles and infrastructure [1]. Taking advantage of the diversity of existing messages in vehicular networks can be a key enabler to optimizing network resource management and addressing scalability problems while meeting safety application requirements. However, the matter of choosing the optimal message type based on network conditions and environmental context is yet to be addressed to ensure optimal performance and efficiency of vehicular networks and particularly road safety applications [2]. Moreover, the consideration of VRUs as active participants within safety applications by their exchanging continuous awareness messages with their environment raises the question of how to design an effective and fair communication framework that considers the limited battery resources of VRUs while ensuring their safety [3]. On top of that, the system should be scalable and able to support the large number of communicating road users to avoid overloading the network. All the aforementioned challenges pushed us to conceive a solution that leverages artificial intelligence and machine learning [4]. Reinforcement learning (RL) techniques are a promising solution to address the challenges faced in vehicular networks [5], particularly regarding load balancing and network resource management. In this paper, we suggest a frequency-adjusting RL algorithm, that relies on smart clustering, to streamline the process of determining which message type would be most advantageous for transmission in vehicular networks. The remainder of this paper is organized as follows: in the following section, we delve into the relevant literature. In section III, the utilized system model is outlined. Details of the RL algorithm can be found in section IV. In section V, we analyze and present simulation results for performance evaluation purposes. Finally, section VI provides our future directions and concludes the paper.

II. Related work

The current regulations for generating collective perception messages have been found to create an excessive amount of messages, each reporting on only a few detected objects. This results in increased communication overhead and decreased reliability of V2X communications and perception capabilities. An algorithm is proposed in [6] that reorganizes how information about detected objects is transmitted, thereby reducing the number of collective perception messages per second. It aims to decrease communication load and overhead while enhancing both V2X communications reliability and cooperative perception by modifying the content of CPMs using prediction. In [7], [8], the authors put forth a plan for optimizing communication for CAVs. If one vehicle receives updated information about an object from another vehicle, it won't need to rebroadcast that same information. This would decrease redundancy and lessen the burden of communication.

In addition to the aforementioned prediction and redundancy mitigation techniques, Q-learning is a commonly used reinforcement learning method that can be employed to address network scalability issues and reduce network overload. For instance, [9] explores how reinforcement learning can be utilized as a substitute for the existing optimization technique in managing network resources in vehicular networks. A collaborative edge computing framework is developed in [10] to reduce service latency and improve reliability. The learning algorithm can predict network traffic demands based on network performance metrics (such as latency, packet delivery ratio (PDR), channel busy rate (CBR), etc.) to orchestrate the radio resources efficiently and solve congestion problems.

Authors of [11] use RL to determine transmission parameters according to present channel conditions, offering an adaptive remedy for congestion control. The work showcases how RL techniques can create an appropriate reward system that balances the conflicting goals of congestion control and recognizing surrounding circumstances. To the best of our knowledge, there exists a wide range of message types that facilitate the communication of vehicles' relevant information. However, while regulations are in place for each type of message individually, there is a need for specific guidelines addressing their collective utilization due to the similar data they share. For instance, the following questions are intriguing our curiosity:

1) Under what circumstances should a CPM be generated in lieu of a CAM?
2) Is it feasible to generate both a CPM and a CAM concurrently?
3) When deciding between the CPM and CAM, does the size of the CPM take precedence over the CAM's size for transmission?
4) Can this selection between CPM and CAM be customized based on the specific situation?
5) Can one CPM totally replace the transmission of other CAMs and VAMs for VRUs if they are detected by the CPM-generating vehicle?
6) To what extent does the number of detected objects impact this decision, considering that a high number of objects contributes to a heavier message size?

Therefore, we believe that this paper represents an initial effort to propose the utilization of various existing messages with an adaptive approach to answer the aforementioned questions. The aim of this work, built upon our prior research [12] and inspired by the previously mentioned works, is to employ reinforcement learning techniques for improving resource allocation by leveraging the range of available awareness messages. This involves identifying the appropriate message type with optimal size and adjusting transmission frequency based on network conditions in order to optimize network resource management to solve scalability issues without degrading safety application requirements.

III. System Model

In this section, the reinforcement learning algorithm for the joint adaptation of message type, size and frequency is introduced.

A. Q-Learning Framework

The Q-learning update equation is given by:

$$\begin{aligned} Q_t(s_t, a_t) \leftarrow (1-\alpha) Q_{t-1}(s_t, a_t) \\ +\alpha \left[r_t + \gamma \min_{a \in A} Q(s_{t+1}, a) \right] \end{aligned} \tag{1}$$

where $Q_t(s_t, a_t)$ is the Q-value [13] for state-action pair (s_t, a_t), r_{t+1} is the reward obtained after taking action a_t in state s_t and transitioning to state s_{t+1}, α is the learning rate, and γ is the discount factor. The discount factor γ , which falls between 0 and 1, is used to weigh the importance of immediate versus future rewards. The learning rate α, also within the range of 0 to 1, determines how much weight should be given to new knowledge as opposed to old information (e.g. when α equals to 1 only the latest new information will be considered).

B. Overview of Q-Learning-Based Adaptive Algorithm

As shown in Fig. 1, the framework of reinforcement learning consists of the agent interacting with the environment in a centralized architecture inspired by [14]. The details of this proposed framework are outlined below:

Optimization function: In this work, a centralized RL-based message type and size with frequency adaptation is designed to handle the scalability problem by minimizing the network overload while not impacting the performance of the road safety application. The constrained minimization problem is defined as follows:

$$\begin{aligned} &\min \sum_{i=1}^{N} b_{i \to mec} \\ &\text{s.t.} \begin{cases} \sum_{i=1}^{N} th_{i \to mec} * b_{i \to mec} \leq C \\ \sum_{i=1}^{N} d_{i \to mec} * b_{i \to mec} \leq D \\ b_{i \to mec} \in \{0, 1\} \end{cases} \end{aligned} \tag{2}$$

where: N is the total number of road users; $b_{i \to mec}$ is a binary decision variable that indicates whether a vehicle i sends a message to the MEC server or no; $th_{i \to mec}$ is the individual throughput sent from a vehicle i to the MEC server; C is the maximum aggregated throughput that can be handled by the MEC server, taking into account the network bandwidth and processing capacity; $d_{i \to mec}$ represents the maximum allowable delay for transmitting a message from a vehicle i to the server; D represents the maximum allowable delay for transmitting a message, taking into account the QoS requirements of the network.

The objective function is the total network overhead obtained by summing up the number of messages sent by all vehicles and VRUs. The first constraint limits the aggregated throughput which is the sum of all the individual throughputs of vehicles, CAVs, and VRUs, considering the used periodicity, ensuring that the network capacity is not exceeded. The second constraint limits the delay for transmitting a message from any road user to the server, ensuring that the QoS requirements are met. The third constraint limits the values of $b_{i \to mec}$ to either 0 or 1, indicating whether a VRU/vehicle will send a message (VAM/CAM if it is not perceived by a CAV in the neighborhood, CPM if it is identified as a reporting CAV), or not in case its data is reported in a CPM by another reporting vehicle. Therefore, the proposed algorithm tends to find the best message type, size and frequency configuration to balance the network load considering the environment context.

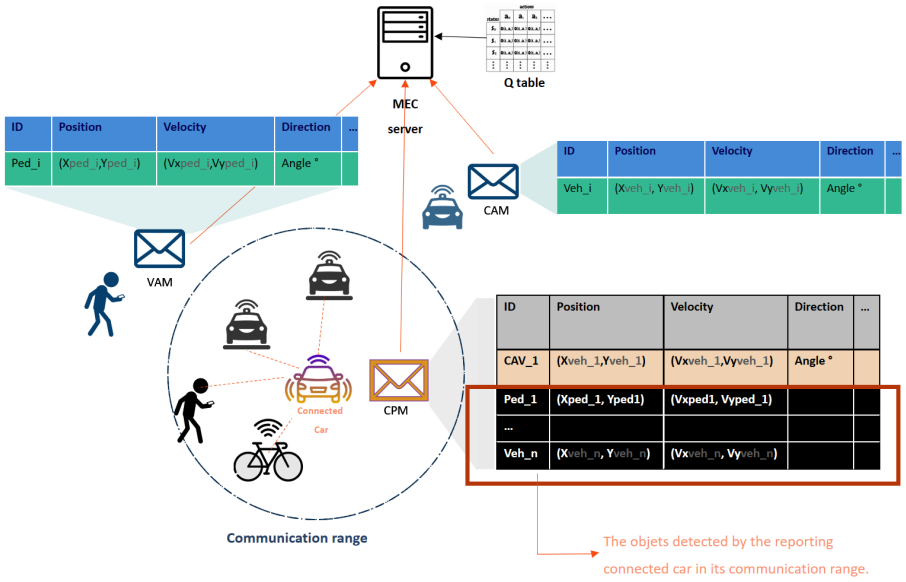

Fig. 1: An illustration of the RL framework for efficient message collection.

It should be noted that this model is still simplified and presupposes that the mode of communication used involves vehicle to infrastructure (V2I) and VRU to infrastructure (V2VRU).

Agent: One central MEC server that takes advantages of the global view of the network with access to traffic history, road users messages, infrastructure sensors and cameras collected data. It can capture the network state to take actions based on a policy, which is a mapping from the state space S to the action space A.

Environment: The overall vehicular network system containing RSU, the MEC server, vehicles and vulnerable road users communicating via their smartphones. This environment takes at time t the state s_t and calculates from the action taken by the agent the new state s_{t+1} and the corresponding reward, which is transmitted to the agent and so on. This succession of events is a step. At each step, the agent observes the state s_t from S and accordingly takes action a_t from A.

State: A collection of information that uniquely identify the situation of the environment. In our case, it is defined as a 3-tuple including a combination of three surrounding context information each one with three values to represent low, medium, and high variation categories: *(i) the velocity variation:* is the average velocity of all the road users in the coverage area of the MEC server divided by the maximum allowed velocity, *(ii) the neighboring rate:* is the average neighbors divided by the maximum number of perceived vehicles/VRU to be included in a CPM message, and *(iii) the network density:* is the current number of road users divided by the maximum supported number by the MEC server.

Actions: All the possible agent actions for each state when interacting with the environment. The action space is discretized and indexed. On the application layer, at each step, the agent takes an action, consisting of a two-dimensional matrix that combines the period at which cooperative messages are sent, and a threshold representing the minimum number of approximate road users a vehicle must have to be designed as a reporting vehicle as represented in Fig. 1. The threshold is used to group vehicles and/or VRU according to their proximity in their communication range, where the elements inside the same group won't send either CAM or VAM because the reporting selected CAV will send a CPM instead containing the data (relative position, speed, etc.) of its neighbors. The other vehicles or VRU that do not belong to any group will send a CAM/VAM. Thus, the threshold is used to adjust the message size by defining the minimum number of perceived objects to include in a CPM.

The solution search space exploration is ensured by choosing the periodicity and the number of perceived objects that have a minimum Q-value, following the epsilon-greedy strategy. Thus, given a probability value of epsilon, ϵ in [0, 1] and a random number r in [0, 1] generated in each learning round, the action a_t is selected as:

$$a_t = \begin{cases} \text{Random action}, & \text{if } r > 1 - \epsilon \\ \text{argmin}_a \, Q_t(s, a), & \text{otherwise} \end{cases} \tag{3}$$

Reward: The reward function that guides the learning, it should be consistent with the learning objectives. The agent selects the best threshold and period parameters according to the overall network context information while respecting the safety application constraints. The goal is to minimize the network load, considering the environmental context information, without degrading the safety application requirements by reducing simultaneous message sending while providing necessary road user data through CPM or CAM for a reliable collision avoidance algorithm used by the MEC server.

$$reward_t = w_1 * N_{th} + w_2 * N_d \tag{4}$$

$$N_d = S_p / Max_p \quad (5)$$

$$N_{th} = A_{th} / Max_{th} \quad (6)$$

Where S_p is the selected periodicity and Max_p is the maximum period that corresponds to 1000 ms. A_{th} is the aggregated throughput which is the sum of all individual throughput, Max_{th} is the maximum throughput if the maximum number of vehicles handled by the same MEC server send individual CAM in each selected periodicity. $w_1 \in [0, 1]$ represents the weight of the normalized aggregated throughput N_{th}, and w_2 represents the weight of the normalized average end-to-end delay N_d. For safety applications, we can set w_2 relative higher than w_1, since applications require lower delay time. For non-safety applications, w_1 can be adjusted higher.

Episode: An episode represents a period of trial when an agent makes decisions and gets feedback from its environment. It ends when the simulation time is reached.

Hyperparameters: are variables that control the performance of the agent during training. *(i) The learning rate:* is a hyperparameter that controls how many new experiences are counted in learning at each step. *(ii) The discount factor:* determines the level of importance given by the reinforcement learning agent to the rewards that are expected in the far future, as compared to those that can be received immediately.

IV. Adaptive Q-learning Algorithm

Our proposed method evolves through two stages; the training stage and the test stage. The environment simulation contains routes populated with vehicles and VRUs. We model a VRU as a vehicle with lower velocity and able to communicate through VAM. To simplify matters and since our focus is solely on the essential components of these messages, we assume that VAM and CAM share identical sizes due to their matching mandatory fields [2]. With the frequency selection that corresponds to a certain network state, the simulator can provide the next state and the reward to the agents. In the training stage, the Q-learning policy used in each action selection is random at the beginning and gradually improved with the updated Q-networks, as described in Algorithm 1.

In the testing stage, the actions of selecting messages' transmission frequencies are chosen with the minimum Q-value given by the trained Q-networks, based on which the evaluation is obtained.

V. Performance Analysis

In this section, we present simulation results to demonstrate the performance of our proposed adaptive approach.

A. Performance metrics

Our Q-learning algorithm for Cooperative Messages Type, Size and Frequency Adaptation was assessed based on several factors including message latency, aggregated throughput (as detailed in section III-B) in terms of number of CAM and CPM, and the quantity of messages transmitted to the MEC server. This evaluation included a comparison between outcomes with and without reinforcement learning using ETSI standard like fixed 10 Hz beaconing rate [3].

Algorithm 1 Q-Learning Training Algorithm

Input: α, ϵ, γ, simulation environment
Output: Q-table
Training `Algorithm()`**:**
  Initialize the model:
    Q(s,a)=0 for all s ∈ S, a ∈ A.
    Start environment simulator and generate vehicles and VRUs.

  for *each episode* **do**
    Initialize S
    for *each step* **do**
      - Capture the required parameters from the environment to compute the current state.
      - Determine the neighboring vehicles/VRU and identify the reporting vehicles.
      - Select the periodicity and the neighboring threshold using the ϵ-greedy strategy.
      - Capture the next state and the reward generated by the environment based on the selected action.

      - Save the (reward, old-state, action, new-state).
      - Update the Q-table using:
      $Q_t(s_t, a_t) \leftarrow (1 - \alpha) Q_{t-1}(s_t, a_t) + \alpha \left[r_t + \gamma \min_{a \in A} Q(s_{t+1}, a) \right]$
    end
  end
  return Q-table

B. Simulation Setup

TABLE I: Experiment setup

Parameter	Value
Training time	∼2 days
SUMO play ground size in m²	2500*2500
CAM size (3GPP Model) [15]	190 Byte
CPM fixed part size in Byte [16]	121 Byte
Size per included perceived object [16]	35 Byte
Learning rate	0.9
Discount Factor	0
epsilon	0.3
Communication range (C-V2X)	500 m

Table I illustrates the crucial aspects of our simulation to provide better comprehension. We developed the simulation model with Python and SUMO simulator [17], incorporating a road network from VANET project in San Francisco covering an area of 2500 m x 2500 m inspired by [14] for improved reliability. As we only focus on the immediate reward, we set the discount factor to zero. To maintain balance between exploration and exploitation, we employed ϵ-greedy policy while utilizing MEC server for hosting the learning algorithm linked to an RSU at the center point of the map. Due to its resource-intensive nature, Grid5000 [18] was utilized to execute the training algorithm, and we allocated computing for the weekend, otherwise the job is killed. In this experiment, we set the maximum number of road users communicating simultaneously supported by one MEC server to 1000. For each network state, we have four values of periodicity 100, 400, 700 and 1000 ms and five values of thresholds 5, 7, 10, 12 and 15 perceived objects. For delay sensitive safety

Algorithm 2 Q Learning Joint Cooperative Messages Type, Size and Frequency Adaptation Testing Algorithm

Input: Q Table, simulation environment
Output: Evaluation results
Testing `Algorithm()`**:**
Initialize the model:
Q(s,a)=0 for all s ∈ S, a ∈ A.
Load the Q-network model.
Start environment simulator and generate vehicles and VRUs.

- Load the Q-Table.
- Start environment simulator and generate vehicles and VRUs.
for *each step* **do**
- Get current neighbouring rate;
- Get current network density;
- Get current average speed;
- Compute the current state using the same indexing function in the training;
- Map the current state to QTable states;
- Select the frequency by choosing the action with the largest Q-value;

end
return Evaluation results

TABLE II: States space

State	Velocity Variation (%)	Neighboring rate (%)	Vehicular density (%)
0	[0, 33[	[0, 33[	[0, 33[
1	[0, 33[	[0, 33[	[33, 66[
2	[0, 33[	[0, 33[	[66, 100]
6	[0, 33[	[33, 66[	[0, 33[
...	...	...	...
26	[66, 100]	[66, 100]	[66, 100]

applications, we can choose lower periods to minimize the delay. To reduce complexity in the state space, we establish 3 tiers for each element of the 3-tuple state. These tiers comprise low, medium and high levels as represented by table II. The assumption is that the MEC server possesses comprehensive network knowledge and is aware of the vehicles capable of transmitting a CPM.

C. Simulation Results

Fig. 2 plots the state variations in each simulation step when Fig. 3 shows the selected period by the RL algorithm during the testing scenario example. The algorithm's decision-making process in selecting a periodicity is impacted by the reward function assigned to each state. In order to minimize latency and the aggregated throughput, the algorithm aims to determine an appropriate threshold value that represents the number of perceived objects to include in a CPM.

As shown in Fig. 4, by decreasing the number of sending vehicles/VRUs and incorporating neighboring vehicles' data using a 35-byte field per detected object in CAVs CPM

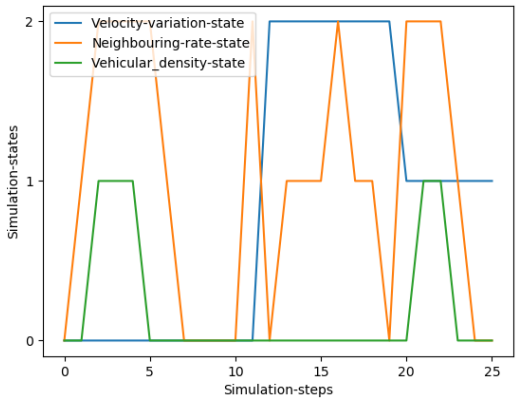

Fig. 2: The testing scenario states.

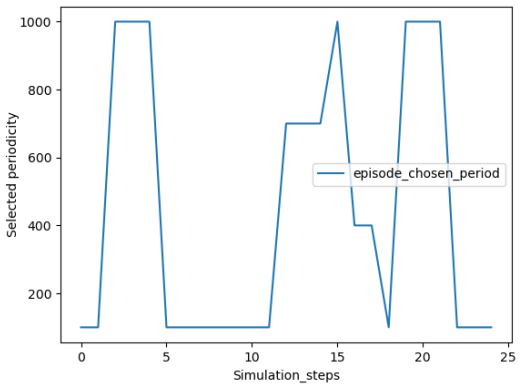

Fig. 3: Selected periodicity.

messages, significant improvements in throughput have been achieved. Furthermore, adjusting message periodicity based on network conditions rather than individually sending CAM every 100 ms has resulted in reduced throughput.

For instance, consider the steps from 0 to 5. In this case, the decrease in overall throughput, compared to a fixed beaconing rate of 10 Hz with each user sending an individual message every 100 ms, can be attributed to the use of high period and also the presence of a high neighboring rate state. This state suggests that there is a significant number of CAVs

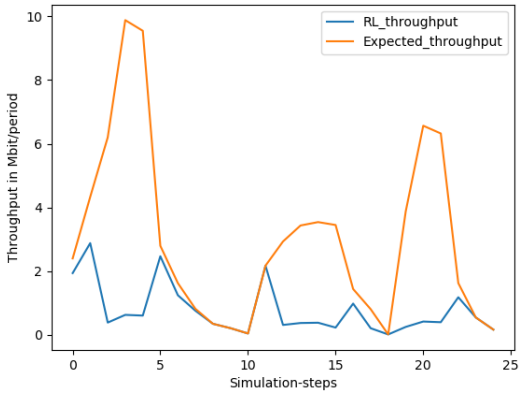

Fig. 4: RL vs expected throughput using 10 Hz fixed beaconing.

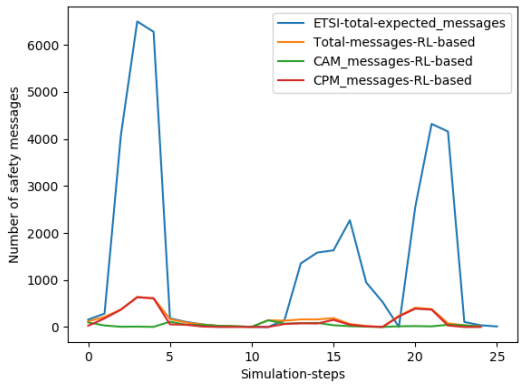

Fig. 5: The number of messages in CAM and CPM using RL along with the anticipated number of messages using 10 Hz fixed beaconing.

transmitting periodic messages containing data from other users. This is confirmed in Fig. 5 where the number of CPMs is the highest compared to CAMs in this steps' interval. Moreover, the utilization of a high period can be attributed to the low velocity variation state. Additionally, since the goal is to decrease the aggregated throughput, the algorithm tends to prioritize selecting longer periods. Furthermore, the RL algorithm can be beneficial for battery-dependent VRUs by reducing the number of CAMs where the VRUs no longer need to transmit their information since it is reported by nearby CAVs in a CPM.

VI. Discussion

The proposed RL algorithm aims to prioritize sending CPM whenever possible, so its performance depends on the number of reporting CAVs that are capable of transmitting CPM in the network. In our implementation, we ensured that each object's data is not redundant if it is detected simultaneously by multiple reporting CAVs. However, in reality, redundancy control becomes more complex if it has to be done at the level of each reporting CAV. A potential resolution involves the MEC server identifying redundant data for the same object and sending a message to a reporting vehicle, instructing it not to include that data in the next transmission.

VII. Conclusion

In this article, we put forward a Q-Learning method for combined Cooperative Messages Type, Size and Frequency Adaptation. The aim is to determine the suitable message type with optimum size and regulate transmission frequency according to network conditions in order to reduce network overload. Our technique has potential benefits for VRUs by reducing their energy consumption when detected by reporting CAVs. To evaluate our approach, we carried out tests using an actual vehicular trace dataset and found that the algorithm satisfies the optimization function objectives. We believe that resorting to reinforcement learning instead of relying exclusively on heuristics or algorithms presents noteworthy benefits in terms of execution time. When envisioning an algorithm that evaluates network modifications whenever it receives a message, one encounters heightened complexity, resulting in additional latency during execution. Nevertheless, by pretraining our model and leveraging a Q-table that provides the MEC server with the optimal action for each corresponding network state, we can reach significant advantages such as low complexity, execution time, and computational resources. In future works, we aim to enhance the reward function by factoring in other network parameters. Furthermore, our research endeavors to examine the reliability of collision detection algorithms by comparing an exclusive utilization of CAMs with our suggested method which incorporates CPM containing the relative kinematics of detected objects.

References

[1] A. Fabio, P. Giovanni, and S. Alessandro. V2x communications applied to safety of pedestrians and vehicles. *Journal of Sensor and Actuator Networks*, 9(1), 2020.

[2] C. Zoghlami et al. 5g-enabled v2x communications for vulnerable road users safety applications: a review. *Wireless Networks*.

[3] E. Alemneh et al. An energy-efficient adaptive beaconing rate management for pedestrian safety: A fuzzy logic-based approach. *Pervasive and Mobile Computing*, 2020.

[4] A. Boualouache and T. Engel. A survey on machine learning-based misbehavior detection systems for 5g and beyond vehicular networks. *IEEE Communications Surveys Tutorials*, 2023.

[5] M. Farzanullah and T. Le-Ngoc. Platoon leader selection, user association and resource allocation on a c-v2x based highway: A reinforcement learning approach, 2023.

[6] G. Thandavarayan et al. Generation of cooperative perception messages for connected and automated vehicles. *2019 IEEE Transactions on Vehicular Technology*.

[7] G. Thandavarayan et al. Redundancy mitigation in cooperative perception for connected and automated vehicles. In *2020 IEEE 91st Vehicular Technology Conference*.

[8] G. Thandavarayan et al. Analysis of message generation rules for collective perception in connected and automated driving.

[9] L. Liang, H. Ye, and G-Y. Li. Toward intelligent vehicular networks: A machine learning framework. *IEEE Internet of Things Journal*, 2019.

[10] M. Li and et al. Deep reinforcement learning for collaborative edge computing in vehicular networks. *IEEE Transactions on Cognitive Communications and Networking*, 2020.

[11] Xiaofeng Liu et al. A q-learning based adaptive congestion control for v2v communication in vanet. In *2022 International Wireless Communications and Mobile Computing (IWCMC)*.

[12] C. Zoghlami et al. Dynamics of cooperative and vulnerable awareness messages in v2x safety applications. In *2022 International Wireless Communications and Mobile Computing (IWCMC)*.

[13] Richard S Sutton et al. *Reinforcement learning: An introduction*. MIT press, 2018.

[14] C. Wang and et al. Sdcor: Software defined cognitive routing for internet of vehicles. *IEEE Internet of Things Journal*, 2018.

[15] 3GPP. Study on lte-based v2x services (v14.0.0, release 14). *document*, 2016.

[16] ETSI. Intelligent transport systems (its); vehicular communications; basic set of applications; analysis of the collective perception service (cps); release 2. *TECHNICAL REPORT*, 2019.

[17] P. A. Lopez et al. Microscopic traffic simulation using sumo. 2018.

[18] Franck Cappello, F Desprez, and D Margery. Grid5000, 2010.