Deep Q-Network Based Routing in a Simplified Vehicular Network: A Proof-of-Concept Study

Bianca S. de C. da Silva, Ana Cecília S. Fernandes, and Samuel B. Mafra

***Abstract*—In this paper, we present a simplified Deep Q-Network (DQN) approach designed to optimize routing decisions in Vehicular Ad Hoc Networks (VANETs). The proposed framework models the environment as a grid-based scenario where each node represents a possible vehicle position, and obstacles emulate dynamic network constraints such as traffic congestion or link failures. Through Reinforcement Learning (RL), the agent autonomously learns to identify the most efficient communication paths, balancing exploration and exploitation to minimize route length and avoid blocked regions. Simulation results demonstrate that the proposed DQN-based agent successfully converges toward the optimal path, achieving stable reward values and smooth convergence over training episodes. The visual results confirm that the learned trajectory closely follows the shortest feasible route, validating the effectiveness of DQN in dynamic vehicular routing problems.**

***Index Terms*—Vehicular Ad Hoc Networks, Deep Reinforcement Learning, Dynamic Routing, Intelligent Transportation Systems.**

I. Introduction

VANETs have emerged as a cornerstone technology for enabling Intelligent Transportation Systems (ITS), supporting applications ranging from collision avoidance and traffic flow optimization to infotainment and autonomous driving. In such networks, vehicles communicate directly with each other or with Roadside Unitss (RSUs) to exchange safety-critical information in real time [1]. However, the dynamic nature of vehicular environments, characterized by high node mobility, frequent topology changes, and variable link quality, poses significant challenges to efficient and reliable routing.

Traditional routing protocols, such as Ad hoc On-Demand Distance Vector (AODV) and Dynamic Source Routing (DSR), rely on deterministic route discovery mechanisms that often fail to adapt quickly enough to the rapidly changing VANET topology [2]. These methods typically incur high overhead and latency when frequent route rediscovery is required. Consequently, they struggle to maintain optimal communication paths under dynamic conditions, resulting in increased packet loss, routing instability, and reduced network efficiency [3].

To overcome these limitations, recent advances in Machine Learning (ML) and, more specifically, in RL, have introduced new possibilities for data-driven, adaptive decision making in wireless networks. By continuously interacting with the environment and learning from feedback, RL agents can gradually identify optimal actions that maximize long-term performance metrics such as throughput, latency, and energy efficiency [4]. Within the broad family of RL algorithms, Deep Reinforcement Learning (DRL) methods, which integrate Neural Networks (NNs) into the learning process, have shown particular promise for handling high-dimensional state spaces common in complex networked environments [5].

In VANETs, DRL has been explored to enhance route stability, spectrum allocation, and resource optimization. Algorithms such as DQN, Double DQN, and Dueling DQN have been successfully employed in adaptive decision-making tasks involving continuous state-action spaces, where deterministic algorithms often fail. These Deep Learning (DL)-based agents are capable of generalizing across large, dynamic topologies, enabling self-organizing and context-aware routing strategies [6, 7].

Nevertheless, most existing studies apply DQN-based routing directly to large-scale or highly realistic simulation environments such as Network Simulator (NS)-3 or Simulation of Urban MObility (SUMO), where the underlying complexity often obscures the fundamental learning dynamics of the agent. Moreover, such frameworks demand high computational cost and intricate mobility models, which can hinder reproducibility and the interpretability of the learning process [8, 9]. Therefore, there is a need for a simplified yet representative vehicular environment where the operation of a DRL agent can be studied in isolation, providing clear insights into convergence, reward evolution, and routing behavior.

In this context, this work explores the use of a DQN to learn dynamic routing strategies in a simplified VANET scenario. The environment is modeled as a grid-based or graph-based structure in which each node represents a potential vehicle position or intersection, and edges emulate communication links. Through trial and error, the DQN agent learns to select optimal paths that minimize cumulative transmission cost and avoid blocked or disconnected regions [10, 11]. This minimal yet interpretable approach enables a controlled evaluation of how DRL can adapt to vehicular communication dynamics without relying on prior topological knowledge.

The main contributions of this work can be summarized as follows:

- A lightweight simulation environment representing VANET routing as a grid-based dynamic topology problem, suitable for RL experiments and educational reproducibility.
- The application of a DQN agent to autonomously learn optimal routes without prior knowledge of network structure, balancing exploration and exploitation in a controlled vehicular context.
- A comprehensive visualization and analysis of the learned paths and training performance, demonstrating the DQN's ability to converge toward efficient and stable routing policies.

The remainder of this paper is organized as follows. Section II reviews related work on RL-based routing in vehicular and wireless networks. Section III presents the proposed method and environment modeling. Section IV details the experimental setup and simulation parameters, while Section V discusses the obtained results. Finally, Section VI concludes the paper and outlines directions for future research.

II. Related Work

Routing optimization in VANETs has been an active research topic for more than a decade, evolving from traditional topology-based protocols toward intelligent, data-driven methods. Early routing algorithms such as AODV, DSR, and Greedy Perimeter Stateless Routing (GPSR) rely on static or deterministic heuristics for path discovery and maintenance. While these approaches are effective under moderate mobility, they face significant challenges in highly dynamic vehicular environments, where route failures and frequent topology reconstructions lead to increased latency and packet loss [2, 3].

The limitations of conventional routing have motivated the adoption of ML techniques to enhance adaptability. While supervised and unsupervised methods can predict link quality or vehicle density, they depend on labeled data and prior mobility information [1]. In contrast, RL allows agents to learn optimal actions through continuous interaction with the environment, optimizing long-term network performance without explicit supervision.

A. *Reinforcement Learning for VANET Routing*

Several studies have proposed RL-driven approaches for vehicular communication networks. In [4], the authors introduced a Q-learning-based method for dynamic next-hop selection, where each node adjusts its forwarding decision based on local rewards. Although the approach improved delivery ratio, it suffered from scalability issues due to the discrete state–action mapping. Similarly, the work in [5] demonstrated the potential of tabular RL for adaptive routing but highlighted its poor generalization capacity in large and dynamic topologies.

Recent advances have evolved from traditional, tabular RL methods toward DRL, which integrates NN into the learning process to approximate value functions in continuous or high-dimensional environments. For instance, Zhang et al. [10] proposed a DQN-based routing algorithm for vehicular networks where each node acts as an agent that learns optimal routes based on connectivity and delay. The approach achieved improved adaptability but required significant computational resources. Similarly, Fatemidokht et al. [3] employed a deep RL framework for delay-sensitive communication in Internet of Vehicles (IoV) scenarios, achieving higher throughput under variable traffic densities.

In another line of work, double DQN and dueling architectures have been employed to stabilize training and reduce overestimation bias. Li et al. [12] implemented a double DQN routing scheme capable of dynamically selecting relay vehicles while minimizing route switching overhead. Meanwhile, Wang et al. [13] proposed a dueling DQN strategy that separates value and advantage functions to enhance decision accuracy in heterogeneous vehicular environments. These methods demonstrate the growing maturity of DRL as a viable alternative to heuristic routing.

TABLE I
Summary of Reinforcement Learning-Based Routing Approaches in VANETs

Ref.	Method	Objective	Environment
[10]	DQN	Delay-aware adaptive routing	SUMO + NS-3
[3]	Deep RL	Delay-tolerant routing under mobility	IoV topology
[12]	Double DQN	Stable relay selection and link adaptation	Urban simulation
[13]	Dueling DQN	Reduce overestimation in heterogeneous VANETs	Hybrid model
This work	**DQN**	**Learn optimal path in simplified VANET**	**Custom environment**

B. *Simplified and Interpretable Environments*

Despite progress, most of the aforementioned works rely on large-scale simulators (e.g., NS-3, SUMO) and complex mobility traces, which, while realistic, complicate interpretability and reproducibility. Furthermore, such setups make it difficult to isolate the agent's decision-making process from external factors like signal fading, mobility randomness, or vehicle density.

A simplified and interpretable environment allows researchers to examine the fundamental learning behavior of DRL-based routing agents in a controlled setting. This approach is essential for proof-of-concept validation, algorithm debugging, and educational demonstration. To the best of the authors' knowledge, few studies have focused on minimal graph-based vehicular routing environments where the DQN's convergence and exploration–exploitation dynamics can be analyzed transparently. This paper contributes to filling that gap by presenting a lightweight vehicular routing model in which the agent learns to reach a destination node through dynamic link exploration, serving as a foundation for future large-scale implementations. Table I summarizes representative studies that employ reinforcement learning techniques for routing in VANETs, including both traditional and deep variants such as Q-learning, DQN, Double DQN, and Dueling DQN.

III. Proposed Method and Environment Modeling

This section presents the proposed framework for dynamic routing in a simplified vehicular network using a DQN agent. The objective is to design an interpretable, lightweight environment that captures the core characteristics of vehicular communication topologies while allowing reproducible RL experiments. The proposed environment, denoted as Mini-VANET, models intersections and vehicle communication links as nodes and edges of a graph. Within this framework, the DQN agent learns optimal routing policies through interaction with the environment.

A. Environment Design

The proposed MiniVANETEnv is constructed as an undirected weighted graph $G = (V, E)$, where $V = \{v_0, v_1, \ldots, v_K\}$ represents the set of nodes, K is the number of nodes and $E \subseteq V \times V$ denotes bidirectional communication links. Each edge (v_i, v_j) corresponds to a possible wireless connection between two vehicles or between a vehicle and an infrastructure unit. The weight w_{ij} associated with each edge represents the cost or distance of transmission between the two nodes.

At each episode, the agent starts at a source node v_s and aims to reach a destination node v_d within a limited number of steps. The state at time t is defined as a one-hot vector $\mathbf{s}_t \in \mathbb{R}^{|V|}$, where the active element indicates the current node position. The action space $\mathcal{A}$ corresponds to the set of possible next-hop nodes. The environment transition function $T(v_i, a_t)$ returns the next node v_j if the selected action represents a valid edge. Otherwise, the state remains unchanged and a penalty is applied.

The reward function r_t is designed to encourage shortest-path exploration while penalizing invalid or inefficient transitions, as follows:

$$r_t = \begin{cases} +10, & \text{if } v_t = v_d \\ -5, & \text{if } (v_t, a_t) \notin E \\ -1, & \text{otherwise.} \end{cases} \tag{1}$$

Episodes terminate when the destination is reached or when a maximum number of steps $T_{\max}$ is exceeded. This simple but effective formulation promotes convergence toward efficient routing paths while discouraging loops and disconnected transitions.

B. Deep Q-Network Agent

The routing decision is learned through a DQN, which approximates the optimal action–value function $Q^*(s, a)$ defined as [14]

$$Q^*(s, a) = \mathbb{E}\left[r_t + \gamma \max_{a'} Q^*(s', a')\right], \tag{2}$$

where $\gamma \in [0, 1)$ is the discount factor, r_t is the reward, and s' is the next state.

The proposed DQN employs a fully connected NN with two hidden layers of 64 neurons each, activated by Rectified Linear Unit (ReLU) functions, and an output layer of size $|\mathcal{A}|$ corresponding to all possible actions. The network parameters θ are updated by minimizing the Mean Square Error (MSE)

$$L(\theta) = \mathbb{E}_{(s,a,r,s')}\left[(y_t - Q(s, a; \theta))^2\right], \tag{3}$$

where $y_t = r_t + \gamma \max_{a'} Q(s', a'; \theta^-)$ and θ^- represents the target network weights periodically synchronized with θ to improve training stability.

To balance exploration and exploitation, the ϵ-greedy policy is adopted

$$a_t = \begin{cases} \text{random action}, & \text{with probability } \epsilon \\ \arg\max_a Q(s_t, a; \theta), & \text{otherwise,} \end{cases} \tag{4}$$

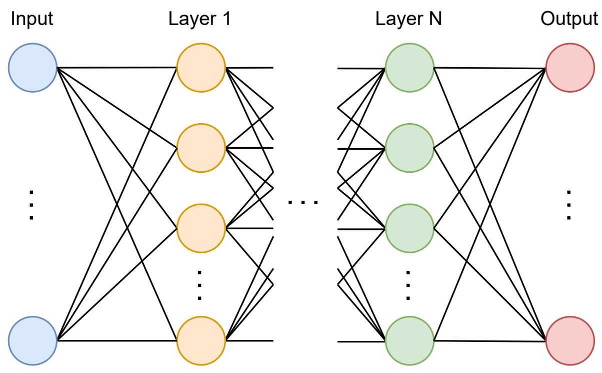


Fig. 1. Architecture of the proposed DQN agent. The network consists of an input layer encoding the node state, two hidden layers with 32 neurons and ReLU activation, and an output layer representing the possible routing actions.

where ϵ decays exponentially after each episode to encourage exploitation as training progresses.

The learning algorithm can be summarized as follows:

1) Initialize replay memory $\mathcal{M}$ and network weights θ.
2) For each episode:
    a) Reset environment to initial state s_0.
    b) For each time step t:
        - Select action a_t using ϵ-greedy policy.
        - Execute a_t in the environment, observe (r_t, s_{t+1}).
        - Store transition (s_t, a_t, r_t, s_{t+1}) in $\mathcal{M}$.
        - Sample a minibatch from $\mathcal{M}$ and update θ via gradient descent.

The overall structure of the proposed DQN model is illustrated in Fig. 1, highlighting the interaction between the input state, hidden layers, and the output action space.

C. Implementation Details

The model is implemented using TensorFlow 2.0, employing the Adam optimizer with a learning rate $\alpha = 10^{-3}$, discount factor $\gamma = 0.9$, and replay buffer size of 2000 transitions. Each training episode consists of $T_{\max} = 10$ steps. The exploration rate ϵ decays from 1.0 to 0.01 with a decay factor of 0.995. The entire training process spans 100 episodes.

An illustration of the environment topology and the agent's decision process is shown in Fig. 2. The MiniCity 3×3 VANET consists of interconnected urban intersections where each node represents a vehicle or potential relay, and the gray lines indicate bidirectional communication links along the roads.

IV. Experimental Setup and Metrics

This section describes the experimental configuration adopted for training and evaluating the proposed DQN-based vehicular routing framework. All simulations were carried out using Python 3.10, TensorFlow 2.12, NumPy, and NetworkX libraries on a workstation equipped with an Intel Core i7 CPU and 16 GB of RAM. The proposed environment and agent implementation are fully reproducible and designed to run efficiently on standard hardware without the need for specialized GPUs.

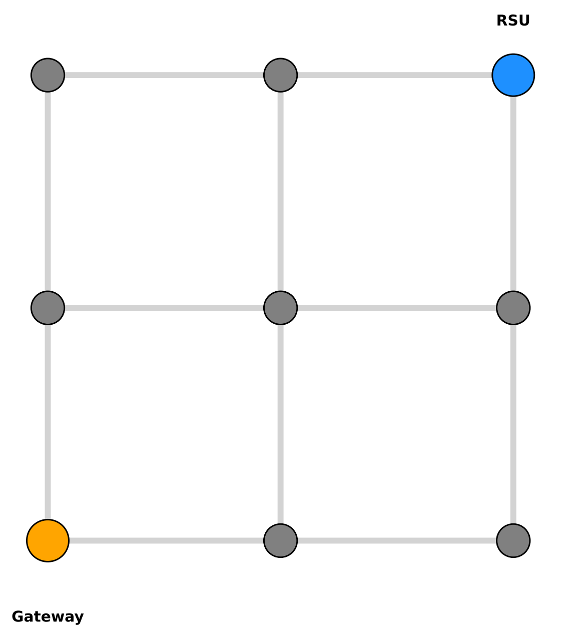


Fig. 2. Topology of the MiniCity 3×3 VANET environment used for DQN training. Each intersection represents a vehicle node or relay, and gray lines indicate the bidirectional communication links forming the urban grid.

A. Simulation Parameters

Table II summarizes the key hyperparameters and environment settings used throughout the experiments. The environment topology consists of 9 nodes interconnected through weighted edges that represent the communication cost or distance between vehicles and roadside units. Node v_0 serves as the source, while node v_9 represents the destination. Each episode begins with the agent positioned at the source node and terminates when the destination is reached or the maximum number of steps $T_{\max} = 10$ is exceeded.

The DQN agent employs two hidden layers of 32 neurons each, using ReLU activation, and a linear output layer corresponding to the number of possible actions. The model is trained using the Adam optimizer with a learning rate of 10^{-3}. The discount factor γ is set to 0.9 to balance immediate and future rewards, while the exploration rate ϵ follows an exponential decay schedule:

$$\epsilon_{t+1} = \max(\epsilon_{\min}, \epsilon_t \cdot \epsilon_{\text{decay}}), \tag{5}$$

where $\epsilon_{\text{decay}} = 0.995$ and $\epsilon_{\min} = 0.01$. The replay buffer stores up to 2000 state transitions, from which random minibatches of 32 samples are drawn at each learning step.

B. Evaluation Metrics

The learning performance of the proposed DQN agent is assessed using both quantitative and qualitative metrics. The total reward per episode represents the cumulative sum of rewards obtained during each training episode and serves as an indicator of convergence and policy stability. A steady increase in this value followed by saturation suggests that the agent has successfully learned an optimal policy. The convergence curve, which illustrates the evolution of average rewards over episodes, provides further insight into training efficiency and stability. As shown in Fig. 3, the agent progressively transitions from random exploration to consistent exploitation behavior. Finally, the learned optimal path demonstrates the agent's ability, after training, to operate deterministically ($\epsilon = 0$) and identify the most efficient route from the source to the destination within the network topology.

TABLE II
SIMULATION PARAMETERS AND HYPERPARAMETER CONFIGURATION

Parameter	Value
Number of nodes ($\lvert V\rvert$)	9
Maximum steps per episode ($T_{\max}$)	10
Learning rate (α)	0.001
Discount factor (γ)	0.9
Batch size	32
Replay buffer size	2000
Initial exploration rate (ϵ_0)	1.0
Minimum exploration rate ($\epsilon_{\min}$)	0.01
Exploration decay factor (ϵ_{decay})	0.995
Hidden layers	2 (64 neurons each)
Activation function	ReLU
Optimizer	Adam
Number of training episodes	300

C. Environment Modeling

The vehicular environment was designed as a simplified urban topology, referred to as the MiniCity 3×3 VANET, composed of nine interconnected intersections arranged in a grid pattern. Each intersection represents a potential vehicle or relay node capable of forwarding data packets, while the gray road segments denote bidirectional communication links among them. The Gateway acts as the source node, and the RSU represents the final destination of the transmitted information. The environment design emphasizes simplicity and interpretability, allowing for a controlled analysis of the agent's learning behavior without the additional complexity of large-scale simulations.

An illustration of the environment topology is shown in Fig. 2. In this environment, the DQN agent interacts with the grid by sequentially selecting the next node to forward packets, receiving feedback in the form of rewards based on connectivity validity, transmission distance, and goal proximity. This design enables a clear visualization of the RL process while maintaining a realistic abstraction of vehicular network routing scenarios. The same topology is later used in Section V to evaluate the agent's ability to discover optimal routes after training.

V. Results and Discussion

This section presents and analyzes the results obtained from the proposed DQN-based routing framework. The performance evaluation focuses on the agent's ability to learn efficient routing policies, its convergence behavior during training, and the interpretability of the resulting paths within the simplified vehicular network.

A. Learning Behavior and Convergence Analysis

The total reward accumulated across training episodes is shown in Fig. 3. During the initial episodes, the agent explores the environment randomly due to the high exploration rate ($\epsilon \approx 1.0$), leading to low or even negative cumulative rewards. As training progresses, the exploration rate gradually decreases, allowing the agent to exploit its learned Q-values and converge toward more consistent routing decisions. After approximately 50 episodes, the total reward stabilizes, indicating that the DQN has successfully identified a near-optimal policy. The upward trend followed by a plateau confirms that the learning process achieved a balance between exploration and exploitation. Occasional reward fluctuations reflect the inherent stochasticity of the ϵ-greedy exploration mechanism and the random sampling of minibatches from the replay buffer. This convergence pattern aligns with expectations for small-scale DRL environments, demonstrating that even with a limited network size and modest computational complexity, the agent can efficiently approximate the optimal action–value function $Q^*(s, a)$ through experience replay and gradient-based updates.

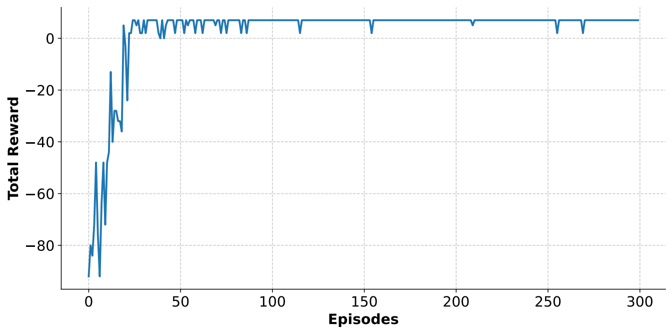


Fig. 3. Evolution of total reward per episode during DQN training.

B. Optimal Routing Path Evaluation

After the training phase, the performance of the DQN agent was evaluated within the same MiniCity 3×3 VANET environment illustrated in Fig. 4. The learned policy converged to the shortest and most reliable route between the Gateway and the RSU, shown by the red path in the figure. Although multiple shortest paths exist in the grid topology, the agent consistently selected the central route (0–3–4–7–8). This preference arises from the higher connectivity of the intermediate nodes, which reduces the likelihood of invalid transitions and maximizes the expected cumulative reward. Hence, the DQN successfully learned an optimal and robust routing strategy purely through interaction with the environment, without requiring global topology knowledge.

C. Comparison with Related Works

While several prior studies [10, 12, 13] have applied advanced DRL variants such as Double DQN or Dueling DQN to large-scale vehicular networks, their implementations typically require complex mobility traces (SUMO, NS-3) and substantial computational resources. In contrast, the present work provides a simplified yet rigorous environment that isolates the learning process, allowing transparent interpretation of the agent's decision-making dynamics.

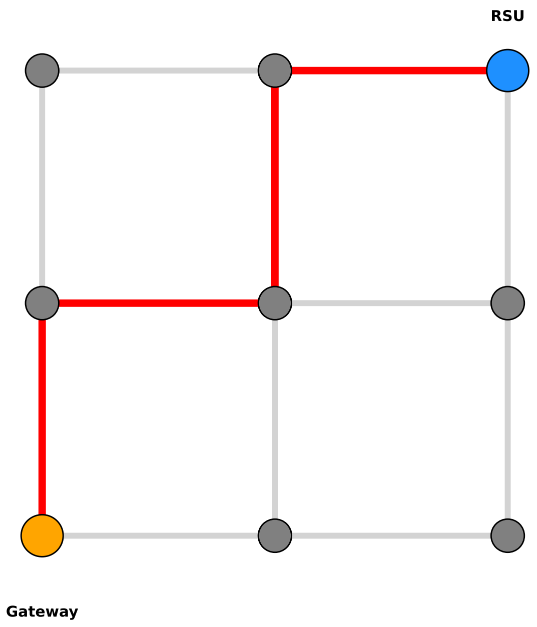


Fig. 4. Learned optimal routing path in the MiniCity 3×3 VANET environment. Each intersection represents a vehicle or a potential relay, while gray lines denote road segments forming the communication graph. The red path indicates the optimal route learned by the DQN agent from the Gateway (source) to the RSU (destination).

Although the proposed model does not incorporate mobility, interference, or traffic flow, its reduced complexity offers valuable insight into how DRL agents form routing strategies. The proof-of-concept results confirm that a basic DQN architecture with two hidden layers can achieve consistent convergence and policy stability, which serves as a foundation for future extensions toward multi-agent coordination, mobility modeling, and larger graph structures.

D. Discussion and Implications

The findings highlight the trade-off between model interpretability and environmental realism. Complex simulators enable high-fidelity performance evaluation but obscure the agent's internal learning process. Conversely, simplified environments such as MiniVANET provide clear visualization of Q-value evolution and decision paths, facilitating algorithmic debugging and conceptual understanding.

Moreover, the lightweight implementation presented here demonstrates that meaningful reinforcement learning behavior can be reproduced on standard computational hardware without specialized Graphics Processing Units (GPUs). This makes the proposed framework particularly suitable for academic instruction, algorithm prototyping, and baseline validation before deploying more sophisticated DRL architectures in large-scale vehicular network simulators.

Overall, the results confirm that the DQN agent effectively learns optimal routing policies through experience, validating the feasibility of DRL as a viable and interpretable solution for next-generation vehicular communication systems.

VI. Conclusion and Future Work

This paper presented a proof-of-concept framework for adaptive routing in vehicular networks using a DQN agent trained within a lightweight graph-based environment. The proposed MiniVANET model abstracted vehicular communication as an interconnected topology of nodes and links, allowing the agent to autonomously learn efficient routing paths without prior topological knowledge. Experimental results demonstrated that the DQN achieved stable convergence and consistently identified the optimal path between source and destination, validating the effectiveness of the reward design and learning strategy. Unlike large-scale simulators, the simplified environment provided interpretability, reproducibility, and computational efficiency, making it an ideal platform for analyzing the learning dynamics of reinforcement-based routing. As future work, this study can be extended toward larger and more realistic vehicular networks incorporating node mobility, link variability, and traffic density; the adoption of advanced DRL architectures such as Double or Dueling DQN and Graph NN; and the exploration of multi-agent coordination and cross-layer optimization to address scalability challenges in next-generation Vehicle-to-Everything (V2X) and Sixth Generation (6G) vehicular communication systems.

Acknowledgment

The preferred spelling of the word "acknowledgment" in America is without an "e" after the "g". Avoid the stilted expression "one of us (R. B. G.) thanks ...". Instead, try "R. B. G. thanks...". Put sponsor acknowledgments in the unnumbered footnote on the first page.

References

[1] Saif Al-Sultan, Moath M Al-Doori, Ali H Al-Bayatti, and Hussien Zedan. A comprehensive survey on vehicular ad hoc network. *Journal of network and computer applications*, 37:380–392, 2014.

[2] Nabeel Akhtar, Oznur Ozkasap, and Sinem Coleri Ergen. Vanet topology characteristics under realistic mobility and channel models. In *2013 IEEE Wireless Communications and Networking Conference (WCNC)*, pages 1774–1779, 2013. doi: 10.1109/WCNC.2013.6554832.

[3] Hamideh Fatemidokht, Marjan Kuchaki Rafsanjani, Brij B Gupta, and Ching-Hsien Hsu. Efficient and secure routing protocol based on artificial intelligence algorithms with uav-assisted for vehicular ad hoc networks in intelligent transportation systems. *IEEE Transactions on Intelligent Transportation Systems*, 22(7):4757–4769, 2021.

[4] Kai Arulkumaran, Marc Peter Deisenroth, Miles Brundage, and Anil Anthony Bharath. Deep reinforcement learning: A brief survey. *IEEE Signal Processing Magazine*, 34(6):26–38, 2017. doi: 10.1109/MSP.2017.2743240.

[5] Leslie Pack Kaelbling, Michael L Littman, and Andrew W Moore. Reinforcement learning: A survey. *Journal of artificial intelligence research*, 4:237–285, 1996.

[6] Yuxi Li. Deep reinforcement learning: An overview. *arXiv preprint arXiv:1701.07274*, 2017.

[7] Shengbo Eben Li. Deep reinforcement learning. In *Reinforcement learning for sequential decision and optimal control*, pages 365–402. Springer, 2023.

[8] Vladi Kolici, Tetsuya Oda, Evjola Spaho, Leonard Barolli, Makoto Ikeda, and Kazunori Uchida. Performance evaluation of a vanet simulation system using ns-3 and sumo. In *2015 IEEE 29th International Conference on Advanced Information Networking and Applications Workshops*, pages 348–353. IEEE, 2015.

[9] Chitraxi Raj, Urvik Upadhayaya, Twinkle Makwana, and Payal Mahida. Simulation of vanet using ns-3 and sumo. *International Journal of Advanced Research in Computer Science and Software Engineering*, 4(4), 2014.

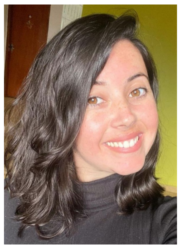
[10] Xu Wang, Sen Wang, Xingxing Liang, Dawei Zhao, Jincai Huang, Xin Xu, Bin Dai, and Qiguang Miao. Deep reinforcement learning: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, 35(4):5064–5078, 2024. doi: 10.1109/TNNLS.2022.3207346.

[11] Rashmi Mishra, Akhilesh Singh, and Rakesh Kumar. Vanet security: Issues, challenges and solutions. In *2016 international conference on electrical, electronics, and optimization techniques (ICEEOT)*, pages 1050–1055. IEEE, 2016.

[12] Z. Li, L. Wang, and X. Liu. Dynamic routing for vanets using double deep q-networks. *IEEE Internet of Things Journal*, 8(12):9783–9794, 2021. doi: 10.1109/JIOT.2021.3058120.

[13] Y. Wang, J. Li, J. Sun, and Y. Zhou. Dueling deep q-network for routing optimization in vehicular ad hoc networks. *IEEE Transactions on Intelligent Vehicles*, 7(4):1125–1136, 2022. doi: 10.1109/TIV.2021.3108865.

[14] Ziming Gao, Yuan Gao, Yi Hu, Zhengyong Jiang, and Jionglong Su. Application of deep q-network in portfolio management. In *2020 5th IEEE International Conference on Big Data Analytics (ICBDA)*, pages 268–275, 2020. doi: 10.1109/ICBDA49040.2020.9101333.



Bianca S. de C. da Silva was born in Santa Rita do Sapucaí, Minas Gerais, Brazil, in 1998. She received the B.S. degree in Control and Automation Engineering in 2020 and the M.Sc. degree in Telecommunications Engineering in 2025, both from the National Institute of Telecommunications (INATEL), Santa Rita do Sapucaí, where she is currently pursuing a Ph.D. degree in Telecommunications Engineering. In 2023, she supported field technicians with remote site integration at Ericsson-INATEL.

Ana Cecília S. Fernandes was born in Campo Belo, Minas Gerais, Brazil. She received the B.S. degree in Computer Engineering in 2024 from the National Institute of Telecommunications (INATEL), Santa Rita do Sapucaí, where she is currently pursuing the M.Sc. degree in Telecommunications Engineering. She works as a Systems Specialist at INATEL, focusing on Artificial Intelligence and Machine Learning applications.