


Article

An Integrated DQN and RF Packet Routing Framework for the V2X Network

Chin-En Yen ¹, Yu-Siang Jhang ², Yu-Hsuan Hsieh ², Yu-Cheng Chen ², Chunhui Kuo ³ and Ing-Chau Chang ^{2,*} 

¹ Department of Early Childhood Development and Education, Chaoyang University of Technology, Taichung 413310, Taiwan; ceyen@cyut.edu.tw

² Department of Computer Science and Information Engineering, National Changhua University of Education, Changhua 50007, Taiwan; m0994014@mail.ncue.edu.tw (Y.-S.J.); s0954010@mail.ncue.edu.tw (Y.-H.H.); s1022038@mail.ncue.edu.tw (Y.-C.C.)

³ Department of Mathematics, State University of New York at Geneseo, Geneseo, NY 14454, USA; ckuo@geneseo.edu

* Correspondence: icchang@cc.ncue.edu.tw; Tel.: +886-4-723-2105 (ext. 8427)

Abstract: With the development of artificial intelligence technology, deep reinforcement learning (DRL) has become a major approach to the design of intelligent vehicle-to-everything (V2X) routing protocols for vehicular ad hoc networks (VANETs). However, if the V2X routing protocol does not consider both real-time traffic conditions and historical vehicle trajectory information, the source vehicle may not transfer its packet to the correct relay vehicles and, finally, to the destination. Thus, this kind of routing protocol fails to guarantee successful packet delivery. Using the greater network flexibility and scalability of the software-defined network (SDN) architecture, this study designs a two-phase integrated DQN and RF Packet Routing Framework (IDRF) that combines the deep Q-learning network (DQN) and random forest (RF) approaches. First, the IDRF offline phase corrects the vehicle's historical trajectory information using the vehicle trajectory continuity algorithm and trains the DQN model. Then, the IDRF real-time phase judges whether vehicles can meet each other and makes a real-time routing decision to select the most appropriate relay vehicle after adding real-time vehicles to the VANET. In this way, the IDRF can obtain the packet transfer path with the shortest end-to-end delay. Compared to two DQN-based approaches, i.e., TDRL-RP and VRDRT, and traditional VANET routing algorithms, the IDRF exhibits significant performance improvements for both sparse and congested periods during intensive simulations of the historical GPS trajectories of 10,357 taxis within Beijing city. Performance improvements in the average packet delivery ratio, end-to-end delay, and overhead ratio of the IDRF over TDRL-RP and VRDRT under different numbers of pairs and transmission ranges are at least 3.56%, 12.73%, and 5.14% and 6.06%, 11.84%, and 7.08%, respectively.

Keywords: vehicular ad hoc networks; vehicle-to-everything; software-defined network; random forest; deep reinforcement learning; deep Q-learning; integrated DQN and RF packet routing framework



Citation: Yen, C.-E.; Jhang, Y.-S.; Hsieh, Y.-H.; Chen, Y.-C.; Kuo, C.; Chang, I.-C. An Integrated DQN and RF Packet Routing Framework for the V2X Network. *Electronics* **2024**, *13*, 2099. <https://doi.org/10.3390/electronics13112099>

Academic Editor: Young-Joo Suh

Received: 22 April 2024

Revised: 19 May 2024

Accepted: 26 May 2024

Published: 28 May 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Due to the Ultra-Reliable Low Latency Communication (URLLC), Massive Machine Type Communication (mMTC), and Enhanced Mobile Broadband (eMBB) capabilities enabled by 5G networks [1], automotive industry manufacturers are currently aiming to integrate 5G technology into the development of autonomous driving systems. However, a rapid increase in the number of these vehicles has led to the swift evolution and large-scale deployment of vehicular ad hoc networks (VANETs) [2], presenting even more significant challenges in this area. There are four primary types of V2X (vehicle-to-everything) technology [3]: vehicle-to-vehicle communication (V2V), vehicle-to-infrastructure communication (V2I), vehicle-to-network communication (V2N), and vehicle-to-pedestrian communication (V2P). This paper focuses on V2V and V2I communication. V2I technology allows vehicles

to relay packets to other vehicles through Roadside Units (RSUs) in the network. Typically, there are physical wired connections between RSUs and the network infrastructure, and IEEE 802.11p [4] is adopted for communication between vehicles. Despite this, a loss of signal strength is caused by buildings or obstacles in the environment, leading to unstable communication. Therefore, a framework is required to gather information about the current state of the V2X network to determine the optimal source-to-destination packet relay path.

Software-defined networking (SDN) is a new technology [5] that allows dynamic changes in the network. SDN is divided into two main components within the network environment: the Control Node (CN) and the Data Node (DN). The CN controls the vehicles when SDN is applied to V2X, and the DN is responsible for communication between vehicles, RSUs, and SDN/OpenFlow switches. RSUs are installed at each intersection to collect information about the vehicles. By utilizing the SDN architecture, the CN can obtain the statuses of all vehicles through RSUs and make optimal decisions on packet forwarding paths in the V2X environment (Figure 1). However, vehicle communication is often unstable due to the high vehicle mobility, insufficient reliability of the inter-vehicle links, and inadequate infrastructure. Therefore, a well-designed packet routing protocol for V2X is crucial.

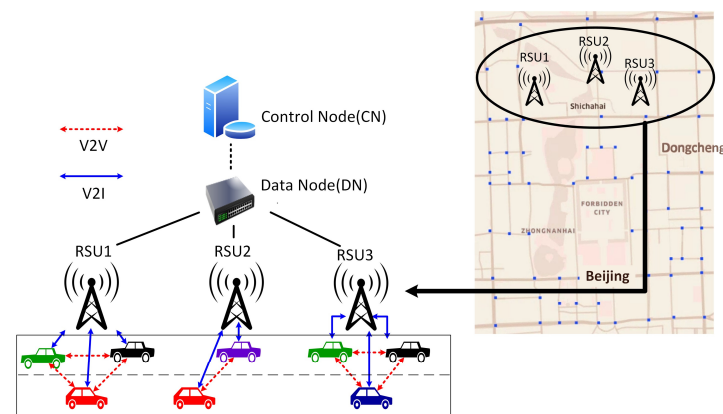


Figure 1. The SDN networking architecture used in this study.

Recently, the integration of artificial intelligence has optimized routing decision-making [6,7]. The three main components of reinforcement learning (RL) [8,9] are environment, decision, and reward. Based on the current state of the environment, the agent selects an action and receives a corresponding reward. Consequently, the environment transitions into a new state. Using Q-learning [10] as an example, the values generated from performing actions in different states are recorded and stored in the Q-table during the reinforcement learning process. Having an excessive number of executed actions and their associated values can lead to excessive occupation of the storage space in the Q-table.

To reduce the size of the Q-table in Q-learning, we proposed the Software-Defined Directional QGRID (SD-QGRID) routing architecture [11], which uses the grid-based concept to partition the entire environment into uniformly sized grids. By deploying the SDN CN and enabling all vehicles to exchange HELLO messages, a V2X centralized control is established. Hence, the SD-QGRID routing algorithm improves the computation method employed by the Q-table and determines the macroscopic packet transmission direction between grids by considering the real-time locations of vehicles and their historical trajectory records involving movement to adjacent grids. Using fuzzy Q-learning, ITAR-FQ [12], reduces potential signal propagation obstruction by buildings, which may lead to rapid signal decay. ITAR-FQ consists of two components: RTAP-RE and RDP-FQ. RTAP-RE is designed to handle traffic information and estimate the road quality (RQ) through a road assessment. On the other hand, RDP-FQ is responsible for exchanging traffic information at intersections between adjacent roads. However, accurate vehicle trajectory data and the issue of end-to-end delay have not been considered in experiments involving ITAR-FQ. As

a result, vehicles carry packets for extended periods, impeding their ability to efficiently transmit them to the destination node through V2X networks. Additionally, the algorithm does not promptly update the optimal routing paths when new vehicles join the network. QAGR [13] optimizes the routing scheme for global message forwarding by categorizing the environment into two parts, aerial UAVs and ground vehicles, with UAVs being responsible for the collection of global road traffic information. Vehicles maintain a fixed-size Q-table and converge towards the reward function proposed by the author. A search of the Q-table filtered by global routing paths allows route requests to be relayed to the optimal nodes. PFQ-AODV [14] incorporates Fuzzy Logic and learns the optimal routing path through the Q-learning method, thereby addressing the challenge of route selection in VANET environments. In summary, these Q-learning-based routing protocols suffer from two apparent problems. First, they do not utilize trajectory data from new vehicles in real-time or consider the reliability of inter-vehicle links, end-to-end delay, vehicle travel directions, or the latest packet forwarding paths. The absence of past historical data for new vehicles presents a challenge when training the Q-learning model, as the optimal routing paths for these vehicles cannot be updated promptly. Second, the excessive number of executed actions and generated values in the Q-table can lead to significant storage space consumption during Q-learning.

Deep reinforcement learning (DRL) [15,16] addresses the issues encountered in complex environments by substituting the Q-table used in Q-learning with neural network layers. TDRL-RP [17] introduces a software-defined trust-based DRL framework. Since the vulnerability of vehicle infrastructure communication to attacks from malicious nodes results in degradation of the overall network performance, the algorithm uses DRL on top of SDN to learn the optimal routing paths in the VANET environment. Its objective is to enhance the network's scalability, flexibility, and self-learning capabilities. Compared to the initial neural network approach, faster DRL convergence can be achieved by integrating a CNN model with SDN. Regarding the packet forwarding strategy, a neighbor behavioral trust module is employed with TDRL-RP. However, this still does not account for the latest forwarding paths, end-to-end delay, or vehicle travel direction. VRDRT [18] integrates DRL into the VANET environment. Due to the high vehicle mobility and consequent rapid changes in the communication links and network topology, algorithm design must address packet transmission delay and stability issues in the networks. The algorithm emphasizes the establishment of routing paths for traffic and vehicle density predictions to increase the packet forwarding probability, reduce the transmission delay, and minimize the packet-carrying instances. Both VRDRT steps are implemented using DRL. The first step explores the optimal paths from the current path, and the second searches for potential routing paths. A RSU collects and stores traffic information from all vehicles in each road segment. It uses DRL to predict road traffic conditions and calculate delays and destinations. However, the algorithm does not account for the latest packet forwarding paths or the reliability of inter-vehicle links.

This study adopted the SDN architecture to improve the performance of the AI training phase and to allow real-time optimal decision-making. The proposed algorithm is divided into offline and real-time phases. The offline phase includes several algorithms that correct the GPS trajectory, determine the actual vehicle movement trajectories, add information on vehicle arrival, and estimate the average vehicle speed on the road to rectify historical trajectory data. Next, we proposed the first routing strategy of the proposed Integrated DQN and Random Forest (RF) [19,20] Packet Routing Framework (IDRF), known as IDRF_DQN. The strategy exclusively utilizes the DQN for optimal relay node selection and addresses the excessive storage consumption and searching delays in Q-learning caused by numerous action choices. IDRF_DQN solely relies on the DQN to select the vehicle with the highest delay weight as the packet relay node and does not consider the movement directions of the target vehicle or the newly joined vehicles with lower end-to-end path delays during packet transmission. Thus, it may lack the ability to determine the shortest end-to-end delay path, or the packet may fail to meet the target vehicle based on

the real-time VANET environment. Conversely, if the routing scheme solely relies on RF to select an intersection for packet forwarding without considering which relay node is optimal, some vehicles may experience long delays when carrying packets. Furthermore, if the communication time between vehicles is too short, incomplete packet transmission may occur.

To address the abovementioned issues, we designed a second routing strategy within the IDRf framework, IDRf_DQN_RF. The algorithm utilizes a combination of the DQN and RF approaches. The delay weights of relay nodes computed by the DQN and the intersection transfer probabilities calculated by RF are recalculated as their transfer delay weights to determine a new relay path, addressing the problem of exclusively relaying to the vehicle with the highest delay weight determined by the DQN. Furthermore, we proposed an algorithm to estimate vehicle link stability, determine the feasibility of packet transmission with neighboring vehicles, and calculate the minimum road segment delay. This approach prevents relays from carrying a packet for a long time or incomplete packet transmission due to inadequate vehicle communication time.

However, the IDRf_DQN_RF approach still does not account for the vehicles not included in the training process. In this study, we defined vehicles that are not incorporated in DQN training as “real-time new vehicles”. The packet transmission path generated by new vehicles may be better. Therefore, in this study, during the real-time phase, we calculated the end-to-end delay and vehicle link reliability based on vehicle information. In combination with offline RF and DQN models, we established a comprehensive IDRf routing strategy to attain the shortest end-to-end delay based on the optimal packet transmission path for real-time vehicles based on the current VANET environment. The following main contributions to the two IDRf phases are made:

- In the offline phase, real-world moving trajectory data from Beijing taxis are employed to assess the use of the vehicle trajectory continuity algorithm to correct vehicle GPS trajectories, add data on vehicle arrival at intersections, and estimate the average vehicle speed for a road segment. The algorithm can rectify and analyze vehicle trajectories, resulting in the attainment of accurate information and better routing decisions.
- The SDN architecture, which utilizes historical vehicle information to train a packet relay node model using the DQN, is adopted when designing the IDRf_DQN framework. The IDRf_DQN_RF framework, which combines IDRf_DQN with RF, is proposed to train a vehicle intersection traversal model using historical trajectory information. These two frameworks adopt the DQN to reduce excessive storage consumption and searching delays during Q-learning.
- In the real-time phase, we propose algorithms to estimate the vehicle link stability, determine whether complete packet transmission can be carried out between the packet-carrying vehicle and the relay ones, and calculate the minimum segment delay, which is the delay when transmitting packets from one road segment to the next. The optimal packet transmission path can be found when considering both real-time and non-real-time vehicles.
- We propose and adopt an integrated IDRf framework that combines IDRf_DQN_RF with real-time DQN-determined packet routing and exception-handling mechanisms. When new vehicles enter the network in real time, their information is added to the segment delay calculation. Then, the framework determines whether these new real-time vehicles can establish new packet relay paths by combining the relay node delay choices from the trained DQN model and the intersection relay probabilities computed through RF. Consequently, this integration enables the shortest end-to-end delay packet relay path for real-time vehicles between the source and the destination to be determined.

2. Related Work

2.1. The Deep Q-Learning Network for Deep Reinforcement Learning

The DQN model for DRL was utilized in this study to address the issues encountered in complex environments by substituting the Q-table used in Q-learning with neural network layers (Figure 2). Two crucial components of DRL are experience replay and fixed Q-targets. Experience replay is a replay mechanism that employs a replay memory to store the different strategies (S_t, a_t, S_{t+1}, r_t) experienced in the same state during learning. In DRL, experience replay acts as a memory storage repository that controls the maximum number of entries in the replay memory, thereby avoiding the unbounded growth that can occur in the Q-table. During each DQN update, experiences from the replay memory can be randomly sampled for learning rather than sequential selection, which can lead to a high level of similarity among the chosen samples, potentially causing the DQN to converge to local optima during the learning process. Utilizing replay memory with random sampling breaks the correlations between data in the dataset. There is no need for frequent interactions with the environment; new parameters can be obtained through training using previous experiences (Figure 3). During training, neural networks are typically trained in batches, a process known as batch learning in machine learning. This approach offers advantages such as increased speed and accuracy in learning (Figure 4).

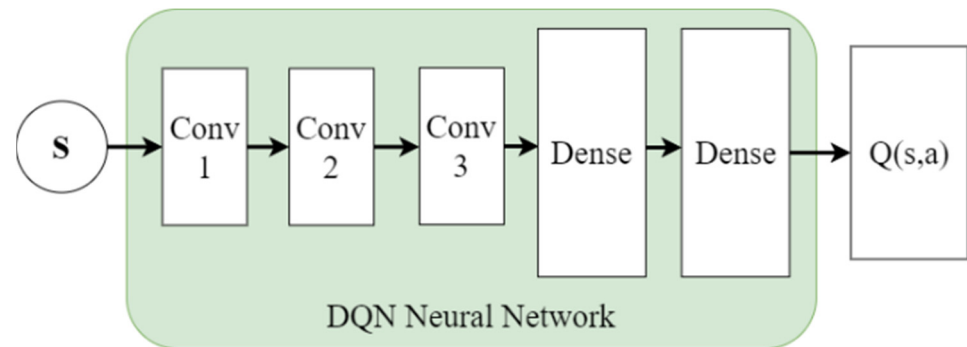


Figure 2. The DQN network architecture diagram.

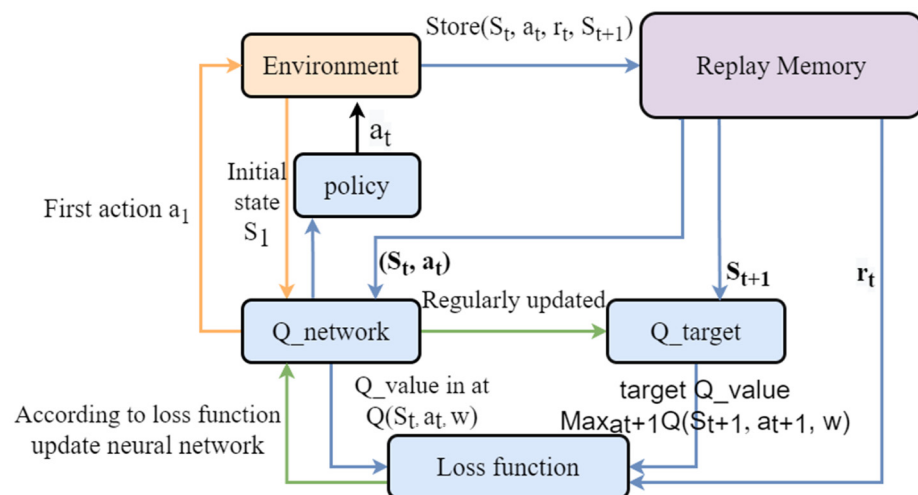


Figure 3. The DQN learning process diagram.

Fixed Q-targets are primarily employed to address the target value instability within the Q-network. Unlike the Q-network, the target values in fixed Q-targets do not change with each update during training. Instead, they are updated at regular intervals or after a certain number of training steps. In the fixed Q-target update process, the Q-network's neural network parameters are copied to the fixed Q-targets to ensure that the parameters

of the fixed Q-targets remain relatively stable. During Q-learning, the action with the maximum initial value at each state is selected. However, the primary focus for neural networks is to optimize the loss function through the selection of appropriate parameters during training, which requires a training dataset with inputs (x) and labels (y). In Q-learning, the state corresponds to the input (x), and value (Q-value) learning serves as the label (y). The objective is to make the estimated Q-values approach the target Q-values, leading to the definition of a loss function (Equation (1)). This formulation allows the use of neural networks for training. A loss function is introduced to incorporate this process into the Q-learning update formula and convert it into the DQN update formula (Figure 5).

$$Loss\ function = (\gamma \times \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}, w) - Q(s_t, a_t, w))^2 \tag{1}$$

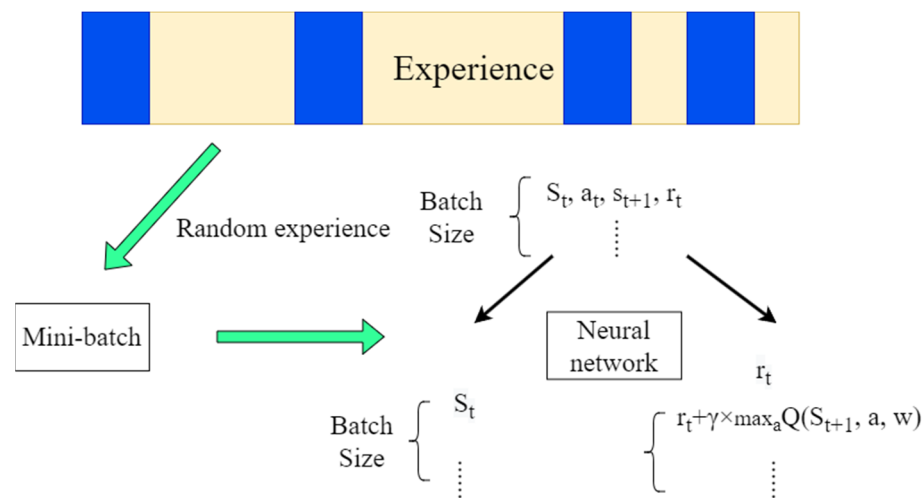


Figure 4. Batch learning for replay memory (the blue blocks are randomly sampled experiences among all stored in the replay memory).

$$Q^{new}(s_t, a_t, w) \leftarrow (1 - \alpha) \times Q^{old}(s_t, a_t, w) + \alpha \times (r_t + \gamma \times \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}, w))$$

Weights
Learning rate
Learning Value

Discount Factor
Estimate the best value in the future to optimize the Loss function

Reward

Figure 5. The core DQN formula.

2.2. Machine Learning Techniques Applied to Networking Issues

The literature contains many examples of the application of machine learning techniques to networking issues. For example, in [20], a two-layer machine-learning detection model for detecting malware on the Android operating system is proposed. The structure of the model is based on ensemble learning using an RF with extra trees and a bagging technique. All features are considered when splitting a node, thereby reducing uncertainty and potential bias. With the aid of RF, this method outperforms techniques employed in previous work and is more accurate when classifying attack types on the same dataset. The Temporal Deep Q-Learning Network (tDQN) [21] represents a self-learning reinforcement-based model that prevents a surge in network traffic on IoT devices using SDN. It designs a model that utilizes DQN to account for dynamic network traffic and fluctuations and is pre-trained with an LSTM model to prevent over-fitting. This algorithm was tested in three scenarios, including mild, medium, and heavy traffic. It outperformed traditional networks regarding throughput, delay, jitter, packet delivery ratio, and packet loss. Additionally, it was shown to auto-balance traffic fluctuations. This approach has been tested on email, HTTP, FTP, and several other applications. Further, in [22], a method is proposed to measure the trustworthiness of driverless cars using deep neural network (DNN) models. Being

feedforward networks, DNNs represent complex functions, such as traffic management, more efficiently. The study utilized traffic data, both real-time and historical, to calculate the trust scores of vehicles. The traffic data used in the study were generated through SUMO, with a map of Melbourne used to simulate real-world scenarios. The proposed method also detects compromised components such as cameras, radar, and LiDAR. However, it assumes that the historical data are correct and does not include correcting GPS positions.

2.3. Learning-Based VANET Routing Research for Packet Forwarding

This study focused on learning-based VANET routing research for packet forwarding. Here, we present the six works most relevant to our proposed IDRF approach. Since signal propagation can be hindered by tall urban buildings, causing rapid signal decay, the intersection-based traffic-aware routing protocol with fuzzy Q-learning (ITAR-FQ) was proposed [12]. The framework comprises two main components: the real-time traffic aware process and road evaluation (RTAP-RE) and the routing decision process with fuzzy Q-learning (RDP-FQ). The RDP-FQ indicates the traffic information from adjacent roads at intersections under the operation of the RTAP-RE without utilizing actual vehicle trajectory data, neglecting end-to-end delay issues and real-time updates regarding optimal routing paths upon the entry of new vehicles. In QAGR [13], a global message-forwarding routing scheme that categorizes the environment into aerial UAVs and ground vehicles is introduced. UAVs collect global road traffic information, and global routing paths are calculated using the Fuzzy Logic and DFS algorithms. Subsequently, these paths are relayed to ground request vehicles. Vehicles maintain a fixed-size Q-table and converge toward the reward function proposed by the author. A search of the Q-table filtered by global routing paths allows route requests to be relayed to the optimal nodes. However, the proposed approach does not incorporate the actual vehicle trajectory data, thereby neglecting the end-to-end delay issue, the inter-vehicle link reliability, and the real-time update of optimal routing paths upon the entry of new vehicles. In [14] the PFQ-AODV was proposed. This is based on ad hoc on-demand distance vector routing (AODV) and integrates Fuzzy Logic to constrain Q-learning and determine the optimal routing path. The aim of this method is to address the routing selection challenges caused by the VANET environment. The available bandwidth (BWF), node mobility (MF), and link quality (LQF) are considered in Fuzzy Logic. However, this process does not incorporate the actual vehicle trajectory data, inter-vehicle link reliability, vehicle traveling direction, or real-time updates of optimal routing paths upon the entry of new vehicles.

In [15], a software-defined trust-based DRL framework, TDRL-RP, is proposed. Due to the high mobility of vehicles and the scarcity of infrastructure in VANETs, communication can be affected by malicious node attacks, causing a decline in the overall network performance. TDRL-RP integrates DRL into an SDN infrastructure, aiming to learn the optimal trusted routing path in the VANET and enhance the overall network scalability, adaptability, and self-learning capabilities. DRL is deployed using SDN, and the addition of the CNN in DRL results in faster convergence than in the original neural network. Regarding forwarding strategies, a neighbor behavior trust evaluation module is introduced. However, issues related to new vehicles, end-to-end delays, or vehicle traveling directions have not been addressed. In VRDRT [18], the incorporation of DRL into VANET is introduced. The high mobility of vehicles leads to rapid changes in communication links and network topology, making it crucial to address issues related to longer transmission delays and stability. Traffic prediction and road vehicle density are emphasized during the routing path establishment process. The objective is to enhance the packet forwarding probability, reduce transmission delays, and optimize packet-carrying scenarios. The VRDRT is divided into two steps, both implemented using DRL. However, the issues of new-added vehicles, connection reliability, and the adequacy of packet forwarding in regions with higher vehicle density are not considered. To reduce the state space size and accelerate the Q-learning convergence speed, IV2XQ [23] was developed. This contains an intersection-based V2X routing protocol using Q-learning and sets the intersection as the state. The Q-learning algorithm is trained with

historical traffic flow on road segments in an urban scenario. The greedy algorithm was chosen as the model's packet-forwarding strategy. When a vehicle node carrying a packet passes an RSU, it transfers the packet information to the RSU. Based on this information, the RSU determines the next hop intersection for relay using Q-learning, and the source node then uses a greedy method to transfer the packet to the target intersection. The IV2XQ approach learns from historical traffic information in VANET environments. It can select another non-congested road segment by monitoring the real-time link status. Experimental results indicate that the proposed approach outperforms QGrid, GPSR, and RAVP since it can forward packets to road segments with the optimal vehicle density.

Table 1 presents a comparative overview of learning-based research on VANET routing protocols. For most studies [12–14], vehicular data were artificially generated. Even for studies that utilize historical data, the real-time addition of new vehicles was not addressed. Studies employing historical data [17,18] did not address whether the historical data aligns accurately with the actual vehicle travel paths due to GPS localization offset issues. Please note that although TDRL-RP uses the SDN trust-based DRL framework, VRDRT emphasizes traffic predictions and the road vehicle density with DRL, and IV2XQ considers both historical and real-time vehicular information, these three approaches do not correct historical vehicular trajectory information or integrate DRL and RF for real-time vehicle routing decision-making. However, the proposed IDRf approach applies the SDN architecture to integrate DRL and RF to first correct historical vehicular trajectory information and then execute real-time vehicle routing decision-making to determine the shortest end-to-end delay packet relay path between the source and destination. This is done by considering both historical and real-time vehicular information, the end-to-end delay, the inter-vehicle link stability, the vehicle movement direction, and the newly generated packet forwarding path due to adding new vehicles. Approaches adopted in the proposed IDRf approach are discussed in the next section.

Table 1. Comparative analysis of learning-based VANET routing research.

Feature	Work	ITAR-FQ [12]	QAGT [13]	PFQ- AODV [14]	TDRL-RP [17]	VRDRT [18]	IV2XQ [23]	Proposed IDRF
Applies the SDN architecture.		No	No	No	Yes	No	No	Yes
Adopts the DRL model		No	No	No	Yes	Yes	No	Yes
Considers end-to-end delay		No	No	Yes	No	Yes	Yes	Yes
Considers the inter-vehicle link stability.		Yes	No	No	Yes	No	Yes	Yes
Considers the newly generated packet forwarding path due to the entry of new vehicles.		No	No	No	Yes	No	No	Yes
Considers the vehicle's movement direction.		Yes	Yes	No	No	Yes	Yes	Yes
Integrates DRL and RF for real-time vehicle routing decision-making.		No	No	No	No	No	No	Yes

3. Design of the IDRf System

V2X is a VANET in which vehicles are interconnected with other vehicles, people, and infrastructure as nodes. Compared to other point-to-point networks, V2X exhibits higher mobility characteristics, enhancing its scalability and flexibility. However, maintenance of the V2X network topology remains a significant challenge due to its dynamic nature. The primary nodes in V2X are vehicles, either traveling along road segments or at intersections. Within road segments, vehicles can have a maximum of two travel directions (forward and reverse). However, due to the high mobility of vehicles, the establishment of communication links with vehicles in the reverse direction is often more challenging. In [24],

the density of vehicles at intersections was shown to generally be higher than that within road segments. Moreover, the density tends to increase as vehicles approach intersections. This indirectly signifies that there are more transmitting nodes at intersections, offering a better option for relay decisions. To select a relay node that can successfully transmit packets at intersections, the neighbor vehicles present at the intersection must be known, and pairs of neighbor vehicles during the communication time must be chosen to allow complete packet transmission. In this study, the following assumptions about the VANET environment were made:

1. There are RSUs at each intersection, and physical network lines interconnect these RSUs.
2. On each road segment connected at intersections, vehicles periodically transmit vehicle information to an RSU.

Figure 6 depicts the IDRf platform proposed in this study, which can be divided into two phases.

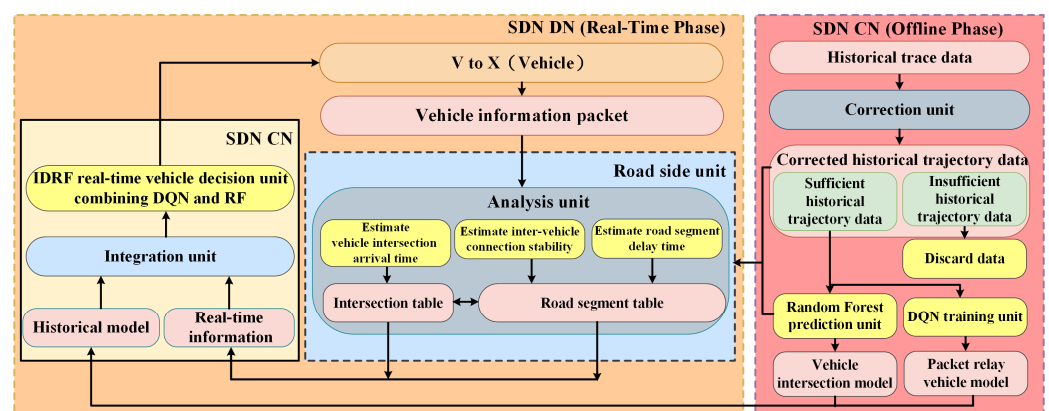


Figure 6. The IDRf system architecture.

- Phase 1 (Offline Phase):

The SDN CN performs offline execution of the correction unit to rectify actual vehicle trajectory data, as presented in Section 3.1. It primarily corrects the coordinate offsets through GPS positioning to align them with the actual road. Subsequently, the rectified trajectories are categorized as sufficient data and insufficient data. Trajectory data classified as insufficient data are discarded, and the corresponding vehicle is considered to be an unknown vehicle with no historical trajectory information during the real-time phase. Trajectory data categorized as sufficient data undergoes the RF prediction unit process. The rectified trajectory data trains the Vehicle Intersection Model using the RF prediction unit. The corrected historical trajectory data feeds into the analysis unit during the real-time phase. Simultaneously, the DQN training unit undergoes DQN training to generate the Packet Relay Vehicle Model. During the real-time phase, the integration unit utilizes the historical model containing the trained Vehicle Intersection and Packet Relay Vehicle Models.

- Phase 2 (Real-Time Phase):

The SDN DN periodically transmits vehicle information to intersection RSUs through wireless communication. The RSUs perform the analysis unit process to estimate the vehicles' times of arrival at the intersection, the inter-vehicle link reliability, and the road segment delay based on real-time information transmitted by vehicles. This information is organized into the segment table for the corresponding road segment and the intersection table for the respective intersection. These two tables represent the real-time vehicle information in the integration unit of the SDN DN during the real-time phase. Finally, by combining the relay node delay decided by the DQN and the intersection relay probability calculated by RF in the offline phase, the real-time vehicle decision unit computes the

segment delay. It determines the packet forwarding path with the shortest end-to-end delay from the source to the destination when a new vehicle enters the V2X network in real time.

3.1. Offline Phase

This study employed vehicle trajectory data from the Beijing city taxi dataset, sourced from the Microsoft platform [25–27]. The dataset encompasses GPS trajectories of 10,357 taxis within Beijing city on 2–8 February 2008. It comprises approximately 15 million recorded points, spanning a cumulative distance of 9 million kilometers. First, we defined a specific region within Beijing city and removed data points from outside the area. The map used in this study encompasses latitudes 39.89851 to 39.94923 and longitudes 116.36701 to 116.41253. The dimensions of this map are a length (L) of 3880 m and a width (W) of 5636 m. Since the original trajectory data do not include information about specific road segments and intersections traveled by vehicles, we annotated the coordination of 72 intersections within the region as $\{I_i, I_j, \dots, I_x\}$. Each road segment is formed by two adjacent intersections labeled $\{S_i, S_j, \dots, S_x\}$, making 110 road segments in total (Figure 7). Moreover, we incorporated the intersection and road segment serial numbers into the original trajectory data. The modified information is presented as follows (Figure 8):

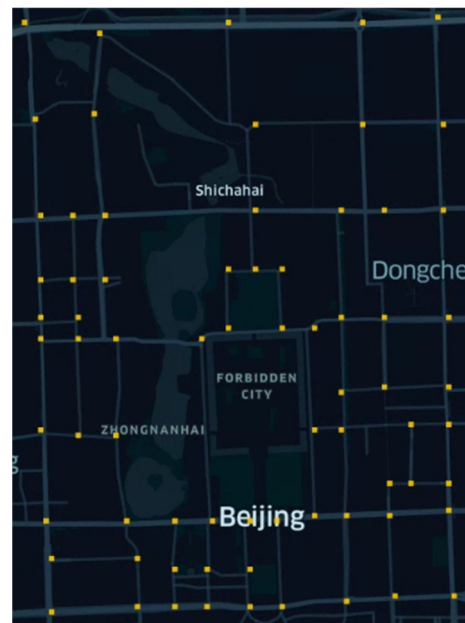


Figure 7. The Beijing city map used (yellow dots are intersections).

```
{
  1 // vehicle serial number
  2008-02-02, 15:36:08 // date, time
  116.51172 // longitude
  39.92123 // latitude
  Ii // road segment serial number
  Sj // intersection serial number
}
```

Figure 8. The modified trajectory information.

3.1.1. Correction Unit for Correcting the Vehicle Trajectory Data

Visualization of the vehicle trajectory data from Figure 7 into a 2D format (Figure 9) reveals that, due to GPS positioning issues, the reported latitudes and longitudes from

the vehicles may not accurately align with the actual roads. Furthermore, the trajectory data may not include vehicle passing intersections, causing inaccurate decision-making regarding the original actual vehicle trajectories, leading to packet forwarding failures. The literature has four approaches to the correction of vehicle trajectory data [28–31]. Most approaches require relevant real-time information from other reference stations or sensors to correct GPS locations and cannot be applied to the offline phase of the proposed IDRF. Therefore, in this study, vehicle trajectory data were corrected by calibrating known GPS trajectory information using accurate map data [28]. By identifying the parts of the trajectory that best match the roads or features on the map, the positions of the historical GPS trajectory can be adjusted before training the AI model. In the following text, we present a series of corrections to the vehicle trajectories through the vehicle trajectory continuity algorithm. First, the distance between two intersections of a road segment and the offset distance of the recorded vehicle coordinates away from the actual road segment are calculated. Then, the Haversine formula [32] is employed to compute the perpendicular distance between the recorded vehicle coordinates and the road segment. It is necessary to find the vertical coordinate points perpendicular to the road segments during the correction process.

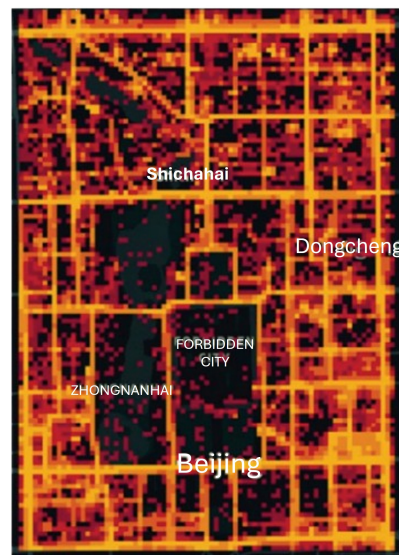


Figure 9. Visualization of vehicle trajectory data from Beijing city (the red points represent the trajectory data, and different shades of orange represent varying densities of vehicle trajectory data).

During the correction process, there may be a few instances where a vehicle's coordinates coincidentally have equal perpendicular distances to two road segments. In such cases, we refer to the preceding trajectory information, denoted as T_i^1 , and the subsequent trajectory information, denoted as T_{i+2}^1 . Based on the vehicle's serial number, trajectory update timestamp (constrained within a 5-min window), and the road segments on which the vehicle is situated at T_i^1 and T_{i+2}^1 , the correction unit can determine which road segment is correct. If discontinuity between road segments is found in the actual vehicle trajectory, the correction unit randomly selects one of the two road segments, and the selected segment is then designated as the corresponding trajectory segment. As illustrated in Figure 10, assume vehicle v_1 with the i th trajectory point T_i^1 is located on road segment 5. During the correction process T_{i+1}^1 , if T_{i+1}^1 has an identical distance to road segments 6 and 9 and T_{i+2}^1 is located on road segment 10, the correction unit can determine that the coordinates of T_{i+1}^1 should be on road segment 9. If the time difference between two adjacent trajectory points, such as T_{i+2}^1 and T_{i+3}^1 , exceeds 5 min, there is no need to compute the continuity between them.

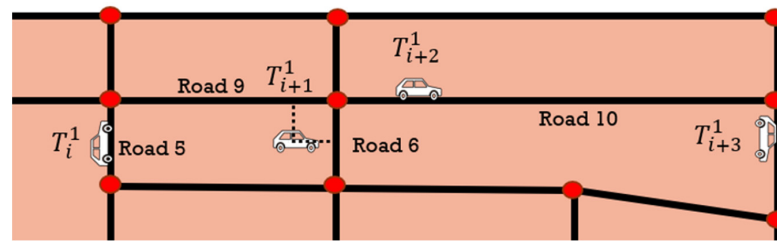


Figure 10. The road segment correction is based on the vehicle trajectory continuity algorithm (the red dots are intersections and dotted lines are perpendicular distances to two road segments).

3.1.2. Adding the Vehicle Arrival Time at the Intersection into Vehicle Trajectory Data

When a vehicle passes through an intersection, the timestamp of its arrival is not always recorded precisely. However, we can use the actual vehicle trajectory information to infer when the vehicle arrived at the intersection. After calculating the vehicle’s arrival time at the intersection, timestamps are added to the original vehicle trajectory information (Figure 9). Assuming that vehicle v_1 passes the trajectory point T_T^1 of a road segment at time T , its coordinates at that moment are $(T_{T_x}^1, T_{T_y}^1)$ and it belongs to road segment i . As vehicle v_1 passes the trajectory point $T_{T'}^1$ of the subsequent road segment at time T' , its coordinates at that moment become $(T_{T'_x}^1, T_{T'_y}^1)$ and it belongs to road segment j . Assuming that intersection R is the shared junction for road segments i and j and that none of its trajectory information is available, we calculate the time at which vehicle v_1 passes intersection R . Assuming the coordinates of intersection R are (I_x, I_y) the total distance traveled between two trajectory points, T_T^1 and $T_{T'}^1$, is first calculated by fd using Equation (2). The total travel time for vehicle v_1 is equal to $ft = T' - T$. The time tr needed for vehicle v_1 to arrive at intersection R from the trajectory point at time T is proportional to the distance between this trajectory point and intersection R divided by the distance between the two trajectory points (Equation (3)). Therefore, the time taken for vehicle v_1 to reach the intersection is calculated by $T + ft \times tr$. After correcting, we insert the arrival time into the original trajectory data (Figure 11).

$$fd = \sqrt{(T_{T_x}^1 - I_x)^2 + (T_{T_y}^1 - I_y)^2} + \sqrt{(T_{T'_x}^1 - I_x)^2 + (T_{T'_y}^1 - I_y)^2} \tag{2}$$

$$tr = \frac{\sqrt{(T_{T_x}^1 - I_x)^2 + (T_{T_y}^1 - I_y)^2}}{fd} \tag{3}$$

Current dataset

Vehicle number	Date	Time	longitude	latitude	Road segment	Intersection
1	9/22	10:00	2	3	5	
1	9/22	10:08	5	4	9	
1	9/22	10:15	8	4	10	

↓

Added dataset

Vehicle number	Date	Time	longitude	latitude	Road segment	Intersection
1	9/22	10:00	2	3	5	
1	9/22	10:02	2	4		6
1	9/22	10:08	5	4	9	
1	9/22	10:15	8	4	10	

Figure 11. Insertion of the vehicle’s arrival time at the intersection (the red frame is the arrival time inserted to the current dataset).

3.1.3. Estimation of the Average Vehicle Speeds for Each Road Segment within Different Periods

After rectifying and classifying the actual vehicle trajectories, we can obtain comprehensive vehicular information, including the time and coordinates at which the vehicle is at the road segment and intersection. Therefore, we can calculate the speed for each vehicle on every road segment. In terms of the calculation method, we partition one day into eight time periods [33], focusing on those with relatively higher traffic densities: 07:00~09:00, 12:00~13:00, 17:00~19:00, and 22:00~23:00. Next, utilizing historical trajectory data, we calculate the estimated average speeds for each road segment within different periods (Figure 12).


	Road segment 1	Road segment 2	Road segment 3	Road segment 4	Road segment 5			Road segment 110
Period 1	60km/h	75km/h	70km/h	65km/h	68km/h			62km/h
Period 2	50km/h	62km/h	48km/h	45km/h	30km/h			38km/h
Period 3	62km/h	73km/h	72km/h	68km/h	70km/h			60km/h
Period 4	48km/h	64km/h	50km/h	43km/h	35km/h			36km/h
Period 5	65km/h	70km/h	68km/h	72km/h	65km/h			65km/h
Period 6	48km/h	50km/h	52km/h	48km/h	35km/h			32km/h
Period 7	68km/h	64km/h	68km/h	74km/h	66km/h			70km/h
Period 8	49km/h	52km/h	55km/h	45km/h	40km/h			45km/h

Figure 12. Estimation of the average vehicle speeds for road segments.

3.1.4. Utilizing Random Forests for the Prediction Unit

The following rules are given to classify the dataset into sufficient and insufficient data and partition them into training, validation, and test sets. The corrected vehicle trajectory data, categorized as sufficient data, are used to train the RF prediction and DQN models.

1. On the rectified vehicle trajectories from the last seven days, taxis with a total timestamp count of fewer than 50 entries are categorized as vehicles with insufficient data. Other vehicles are classified as vehicles with sufficient data.
2. Sufficient data recorded on 2–5 February 2008 are assigned to the training set.
3. Sufficient data for 6–7 February 2008 are assigned to the validation set.
4. Sufficient data recorded on 8 February 2008 are assigned to the test set.

In this study, for the prediction unit, regression RF, a supervised learning method, was employed. RF comprises decision trees and the bagging algorithm [34]. The decision tree uses a greedy approach to obtain distinct classification outcomes for each group. A tree is generated based on training data, and predictions are made for new samples according to the learned rules. The distance between the predicted result \hat{V}_x and the original target V_x is computed as the mean squared error (Equation (4)).

$$Avg_{error} = \frac{1}{n} \sum_{i=1}^n (V_x - \hat{V}_x)^2 \tag{4}$$

The bagging algorithm is applied to a training set $S = [V_1, V_2, \dots, V_x]$. For a training set with V_x data points, each datum is sampled with a probability of $\frac{1}{V_x}$, and the likelihood of not being sampled is $1 - \frac{1}{V_x}$. The probability of a datum not being selected during V_x sampling times is $(1 - \frac{1}{V_x})^{V_x}$. As $V_x \rightarrow \infty$, $(1 - \frac{1}{V_x})^{V_x} \rightarrow \frac{1}{e} \cong 0.368$, meaning that approximately 36.8% of the data in the training set S remain unsampled after undergoing random sampling. Since these data points were not involved in the training process, they are used to assess the model’s generalization capability. This study obtained

M independent training sets by repeatedly sampling N training data points in M rounds. The prediction approach is based on decision trees. By placing the M independent training sets into decision trees separately, the structure of the RF can be formed (Figure 13).

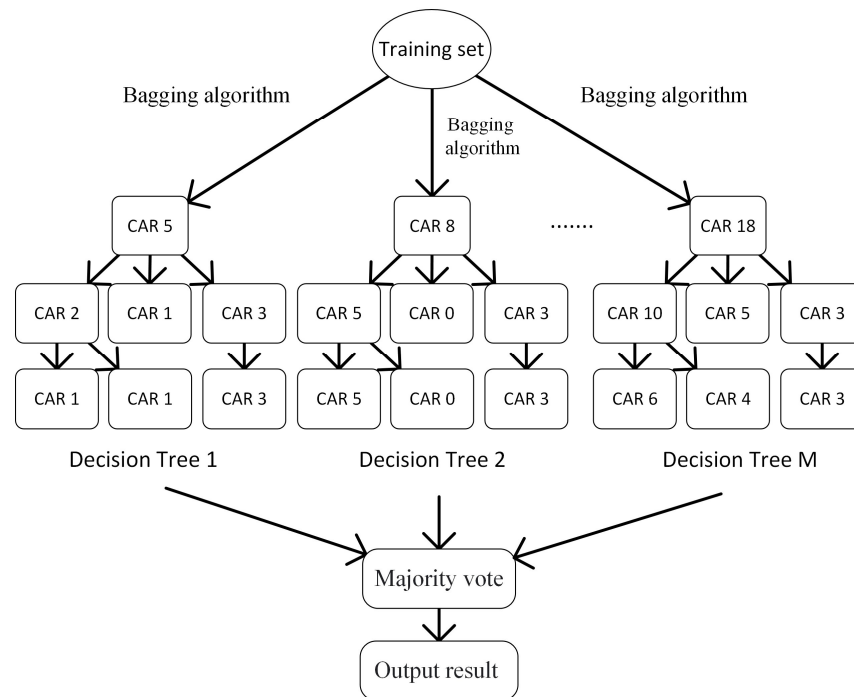


Figure 13. The RF model utilized in this study.

The dataset was used in conjunction with the RF method. Trajectories typically lack regularity after the passenger gets in the taxi, increasing the prediction difficulty. However, given the characteristics of RF, which reduce inter-dataset correlations, a majority voting approach to predicting taxi movement trajectories is better suited for the dataset used in this study. From Figure 11, we can obtain the times at which vehicles enter intersections or road segments. The road segments are set as predictive targets. The rectified vehicle trajectory dataset, vehicle movement direction, and the set of target intersections are denoted as $X = [(V_1, S_1, speed_1, angle_1), (V_2, S_2, speed_2, angle_2), \dots, (V_n, S_n, speed_n, angle_n)]$, $Y = [0 \text{ (forward)}, 1 \text{ (left turn)}, 2 \text{ (right turn)}]$, and $Z = [I_1, I_2, \dots, I_{72}]$, respectively. The RF works as follows: Initially, the training set X is divided into M independent training sets using the bagging algorithm. Then, these M independent training sets are processed within decision trees. Based on each independent training set at every node, a splitting threshold is chosen from the designated label Y. This threshold is employed to partition the data at each node. We can obtain M decision trees and train with these M decision trees. After the completion of training, the mean squared error (MSE) is computed separately for M decision trees. For the training set X, predictions can be made through a majority vote by M decision trees. Our objective was to use the estimated average vehicle speeds of various road sections over different periods to predict the next intersection I_6 , the intersection after the next I_7 , and the corresponding arrival times of vehicles using the RF prediction unit, as depicted in Figure 14. Please note that the corresponding arrival times to the next intersection and the intersection after that are calculated by the SDN CN using the information on real trajectory points belonging to the current and next road sections, respectively. Therefore, the error for the arrival time at the intersection, caused by traffic congestion, the vehicle’s moving direction, and speed, is confined to the corresponding road section.

Time (T)					Time (T + 1)				
Vehicle number	Date	Time	Road section	Intersection	Vehicle number	Date	Time	Road section	Intersection
1	9/22	10:00	5		1	9/22	10:00	5	
					1	9/22	10:02		6

Time (T + 1)					Time (T + 3)				
Vehicle number	Date	Time	Road segment	Intersection	Vehicle number	Date	Time	Road segment	Intersection
1	9/22	10:00	5		1	9/22	10:00	5	
1	9/22	10:02		6	1	9/22	10:02		6
					1	9/22	10:11		7
					1	9/22	10:25		9

Figure 14. Predicted visited intersections and corresponding arrival times determined using the RF prediction unit (the red frame shows the next intersection, the intersection after the next, and the corresponding arrival times).

3.1.5. Deep Reinforcement Learning Training Unit

In this study, the DQN approach was employed to make decisions about the packet forwarding transmission path to achieve the shortest end-to-end delay from the source vehicle S to the destination vehicle D. This decision-making process takes into account the presence of intermediate relay nodes up to M hops (Figure 15). The V2X vehicular network environment consists of vehicles. Within its communication radius, the source vehicle S has N_1 neighboring candidate vehicles at Hop 1 that are capable of relaying packets. This implies that, in the current state s_1 , there are N_1 possible actions to choose from. Vehicle v_i represents the i -th candidate node within the communication radius of the current node v_{carry} , which is currently carrying the packet. Importantly, v_i is not chosen as the i -th relay node. D_i represents the total delay required for source vehicle S to relay the packet to v_i successfully.

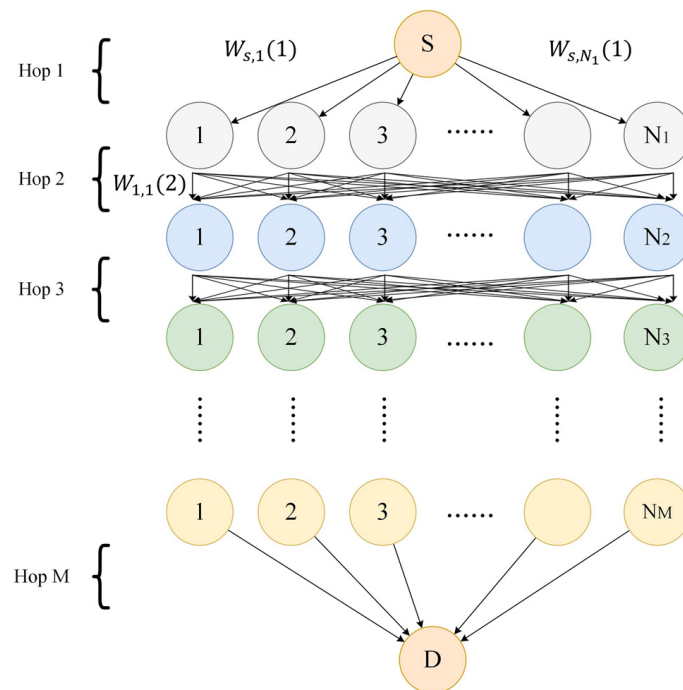


Figure 15. Network model of the decision-making process.

First, we designed the DQN state as follows: $State = \{Vehicle\ ID\ V_i, road\ segment\ ID\ S_i, intersection\ ID\ I_i, vehicle's\ driving\ speed\ speed_i, vehicle's\ driving\ angle\ angle_i, whether\ v_{carry}\ is\ the\ packet-sending\ node\ v_{src}, whether\ v_{carry}\ is\ a\ relay\ node\ v_{relay}, (S_i, I_i), (speed_i, angle_i), and\ (whether\ v_{carry}\ is\ v_{src}, whether\ v_{carry}\ is\ v_{relay})\}$, resulting in a total of 10 dimensions for the DQN neural network. The indication of whether v_{carry} is a relay node v_{relay} signifies that the current node v_{carry} , which is carrying the packet, is neither the sending node v_{src} nor the target node v_{dst} , but rather, is sending other vehicle nodes. The combination of data with similar attributes can increase the sample count, providing more diverse information for the DQN while making decisions and enhancing the decision quality. The *Action* refers to the selection of a neighboring candidate vehicle within the communication radius of the vehicle carrying the packet, which is also the output dimensionality of the DQN network. The *Reward* is designed according to Equation (5), and it encompasses three distinct scenarios: (1) If the DQN selects V_i as a relay node from the candidate vehicles and the packet transmission is complete, the *Reward* is given as $Reward = \frac{1-DR_i}{\sum_{j=1}^{candidate} 1-DR_j}$, where $DR_i = \frac{D_i}{\sum_{j=1}^{candidate} D_j}$ is equal to the proportion of the transmission delay D_i to $\sum_{j=1}^{candidate} D_j$. The objective of the DQN is to determine the packet transmission path with the shortest end-to-end delay. Therefore, assigning higher rewards to candidate vehicles with lower transmission delays D_i (but with higher $1 - DR_i$) is essential. In the end, all candidate vehicle values are normalized within the range [0–1], yielding the final reward $\frac{1-DR_i}{\sum_{j=1}^{candidate} 1-DR_j}$. (2) If V_i is not chosen as a relay node among all candidate vehicles, the reward is set to 0. (3) If V_i is selected as a relay node among all candidate vehicles, the transmission to V_i fails, and the reward is assigned a value of -1 . Based on the rewards provided during the DQN training process, we can obtain weights $[W_{s,1}(1), W_{s,2}(1), \dots, W_{s,N_1}(1)]$ for the selection of actions $[1, 2, \dots, N_1]$. The packet forwarding path must be chosen by using the candidate neighboring vehicle with the highest trained weight at each hop to achieve the shortest end-to-end delay for packet transmission from the source vehicle to the destination vehicle. This selection process is represented by Equation (6). The expected packet transmission path is depicted in Figure 16.

$$Reward = \begin{cases} \frac{1-DR_i}{\sum_{j=1}^{candidate} 1-DR_j}, & \text{packet transmission is complete;} \\ 0, & v_i \text{ is not chosen as a relay node;} \\ -1, & \text{packet transmission fail;} \end{cases} \tag{5}$$

$$a_t = argmax_a Q(w(s_t), a; w) \tag{6}$$

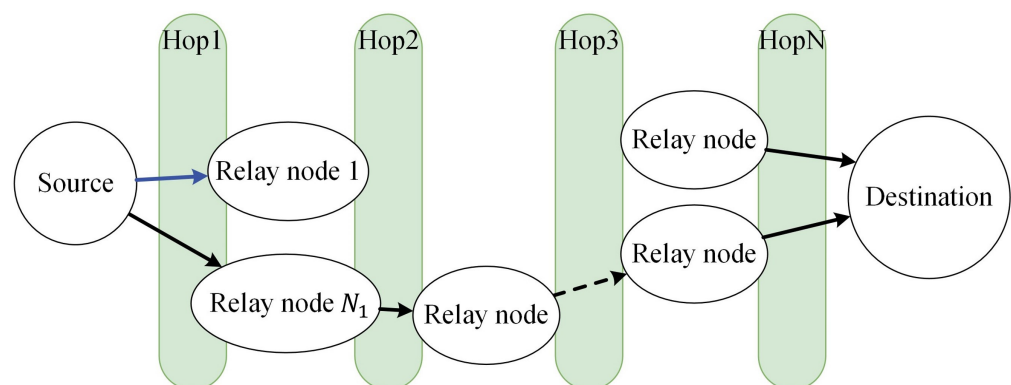


Figure 16. The expected packet transmission path is based on the DQN decision during the offline phase.

3.2. Real-Time Phase

3.2.1. Calculating the Vehicle’s Arrival Time at the Next Intersection Based on Vehicle Information by the Analysis Unit of the RSU

A DRL training unit can determine the optimal end-to-end delay path using the vehicle’s trajectory. Trajectories with insufficient data are excluded from the training process during the offline phase. As depicted in Figure 16, some paths lack intermediary nodes. Hence, untrained vehicle trajectories can be added to determine whether a new route can be formed in the real-time phase. The following sections delve into the detailed calculation of real-time vehicle information.

Vehicles periodically transmit information to RSUs, which are responsible for the calculations. We define vehicle information as $v_{i_T} = \{send_i, speed_i, angle_i, lat_i, lon_i\}$. Assuming that vehicle V_i is on the road, its trajectory V_{i_T} and $angle_i$ indicate that it is moving towards intersection I_x . By using the longitudes and latitudes of vehicle trajectory V_{i_T} and intersection I_x , the distance between them can be calculated. Furthermore, the vehicle’s speed enables the estimation of its arrival time I_{x_T} at the intersection, as shown in Figures 17 and 18. For all vehicles, the analysis unit calculates arrival times at the intersection. It is assumed that the road segment S_1 is composed of intersections 4 and 5. There are five vehicles within road segment S_1 . After transmitting their information to the RSU, the RSU analysis unit calculates their arrival times. The consolidation of vehicle information on the same segment yields a road segment table for each road segment across all intersections. The road segment table includes the Arrival Time to Next Intersection, Vehicle Direction, and Next Intersection fields. This process obtains arrival time data for all segments, as shown in Figure 19. Each RSU records information related to the road segments it covers.

- Input: .

Rectified vehicle trajectory information .

- Output: .

Arrival time at the intersection .

1. for i in the rectified vehicle trajectory information: .

2.
$$V_{i_d} = \sqrt{(V_{i_x} - I_{x_x})^2 + (V_{i_y} - I_{x_y})^2}$$
 // distance between C_{i+1} and the intersection

3.
$$R_{x_T} = V_{i_T} + speed_i \times V_{i_d}$$
 .

4. return I_{x_T} .

Figure 17. Pseudocode is used to calculate the vehicle’s arrival time at the intersection.

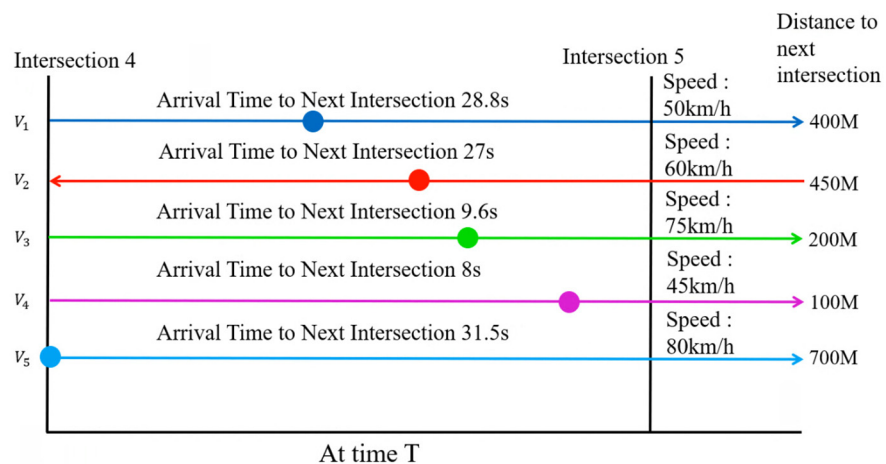


Figure 18. The RSU calculates the arrival time at the next intersection for vehicles in the road section S_1 .

Road Segment Table S_1	Arrival Time to Next Intersection	Vehicle Direction	Next Intersection	Road Segment Table S_{110}	Arrival Time to Next Intersection	Vehicle Direction	Next Intersection
$S_{1_V_1}$	28.8s	Same direction	Intersection 5	$S_{110_V_8}$	5.7s	Same direction	Intersection 70
$S_{1_V_2}$	27s	Opposite direction	Intersection 4	$S_{110_V_{452}}$	3.7s	Opposite direction	Intersection 68
$S_{1_V_3}$	9.6s	Same direction	Intersection 5	$S_{110_V_{584}}$	18.4s	Same direction	Intersection 68
$S_{1_V_4}$	36s	Same direction	Intersection 5	$S_{110_V_{73}}$	28s	Same direction	Intersection 70
$S_{1_V_5}$	31.5s	Same direction	Intersection 5	$S_{110_V_{904}}$	7.4s	Same direction	Intersection 69

Figure 19. Road segment tables record all vehicles’ arrival times at the following intersections.

3.2.2. Estimation of the Minimum Packet Propagation Delay for a Road Segment Using the Vehicle Connection Stability, Road Segment, and Intersection Tables

Due to the high mobility of vehicles, the maintenance of consistent connectivity is often challenging. Therefore, it is necessary to determine whether the connection time between two vehicles is sufficient to transmit a complete packet. According to [35,36], we must first obtain the vehicle speed, moving direction, current location, and transmission range to calculate the available link connection time between two vehicles. Assuming this information is available, the theoretical packet transmission time can be defined as $T = \frac{\text{Packet size}}{\text{Transmission rate}}$. The link connection time between two vehicles must be greater than or equal to the theoretical time T required to transmit one packet (Figure 20). Differences in positions along the X -axis and Y -axis after the packet transmission time T from the initial location are $D_{xij} = (X_j - X_i) + (V_j \cos \theta_j - V_i \cos \theta_i)T$ and $D_{yij} = (Y_j - Y_i) + (V_j \sin \theta_j - V_i \sin \theta_i)T$, respectively. To transmit a complete packet between two vehicles, the distance between two nodes d after the packet transmission time T must be smaller than or equal to the transmission range r , formulated by Equation (7). Let $e = (V_j \cos \theta_j - V_i \cos \theta_i)$, $f = (X_j - X_i)$, $g = (V_j \sin \theta_j - V_i \sin \theta_i)$, $h = (Y_j - Y_i)$. By substituting these values into Equation (5), the packet transmission time T is constrained by Equation (8).

$$d = \sqrt{D_{xij}^2 + D_{yij}^2} \leq r \tag{7}$$

$$\begin{aligned} &\sqrt{(f + eT)^2 + (h + gT)^2} < r, \\ &(e^2 + g^2)T^2 + 2(e f + g h)T + (f^2 + h^2 - r^2) \leq 0, \\ &T \leq \frac{-(e f + g h) + \sqrt{(e f + g h)^2 - (e^2 + g^2)(f^2 + h^2 - r^2)}}{e^2 + g^2} \end{aligned} \tag{8}$$

However, since vehicles do not always travel at a constant speed, the abovementioned equation was modified to incorporate the calculation of the average vehicle acceleration. Assuming vehicle V_i has speed $speed_{i_t}$ at time t and speed $speed_{i_{t-1}}$ at time $t - 1$, the acceleration a_t is obtained by dividing the velocity change ($speed_{i_t} - speed_{i_{t-1}}$) by the time interval ($t - (t - 1)$). Hence, differences in position along the X -axis and Y -axis after the packet transmission time T from the initial location are modified to $D_{xij} = (X_j - X_i) + \sum_{t=1}^T ((V_{j_t} + a_t) \cos \theta_j - (V_{i_t} + a_t) \cos \theta_i)$, and $D_{yij} = (Y_j - Y_i) + \sum_{t=1}^T ((V_{j_t} + a_t) \sin \theta_j - (V_{i_t} + a_t) \sin \theta_i)$, respectively.

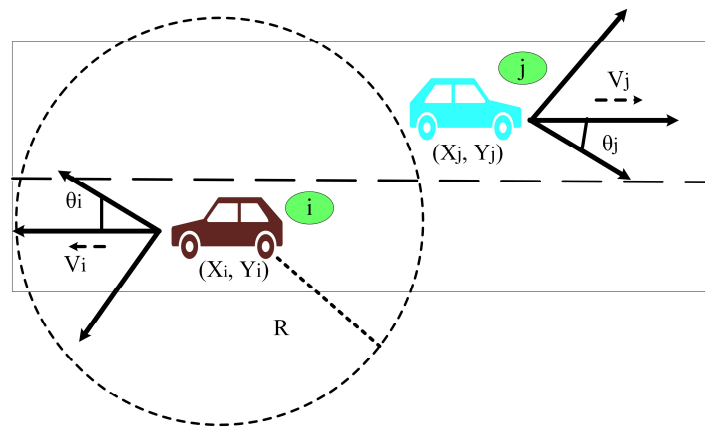


Figure 20. Diagram of the estimated vehicle connection stability.

Next, the analysis unit calculates the vehicle’s connection stability for the road segment where the vehicle v_{carry} is located. If both v_{carry} and the vehicles in that segment cannot transmit packets completely, the vehicles at the intersection where v_{carry} arrives that can be selected for relaying should be determined. We define the time at which vehicles enter each other’s communication radii as the *Encounter Time*. The maximum time taken for vehicles to establish a connection is defined as the *Transfer Time*. The communication radius between two vehicles is set as R . Considering the moving directions of vehicles, we explain this concept in two categories: in the same and opposite directions.

1. Same direction:

- As v_{carry} and the relay vehicle are driving in the same direction and approaching each other, the speed difference between vehicles i and j is defined as $VS = |speed_{i_t} - speed_{j_t}|$. The distance between the two vehicles is described as $VD = \sqrt{D_{xij}^2 + D_{yij}^2}$. When VD is larger than R , indicating that the two vehicles are not within each other’s communication radii, *Encounter Time* = $\frac{VD-R}{VS}$ and *Transfer Time* = 0. When VD is less than or equal to R , indicating that the two vehicles are within each other’s communication radii, *Encounter Time* = 0 and *Transfer Time* = $\frac{VD}{VS}$. In this case, the *Transfer Time* is the time spent by the two vehicles moving from their current positions to the position at which they meet because they are approaching each other.
- As v_{carry} and the relay vehicle drive in the same direction but gradually move apart, the speed difference between vehicles i and j is defined as $VS = |speed_{i_t} - speed_{j_t}|$. The distance between the two vehicles is described as $VD = \sqrt{D_{xij}^2 + D_{yij}^2}$. When VD is greater than R , indicating that the two vehicles are not within each other’s communication radii and will not meet again, *Encounter Time* = ∞ and *Transfer Time* = 0. When VD is less than or equal to R , indicating that the two vehicles are within each other’s communication radii, *Encounter Time* = 0 and *Transfer Time* = $\frac{R-VD}{VS}$. In this case, the *Transfer Time* is the time spent by the two vehicles moving from their current positions until their distance reaches R because they gradually move apart.

2. Opposite direction:

- As v_{carry} and the relay vehicle are driving in opposite directions but approaching each other, the speed difference between vehicles i and j is defined as $VS = speed_{i_t} + speed_{j_t}$. The distance between the two vehicles is described as $VD = \sqrt{D_{xij}^2 + D_{yij}^2}$. When VD is greater than R , indicating that the two vehicles are not within each other’s communication radii, *Encounter Time* = $\frac{VD-R}{VS}$ and *Transfer Time* = 0. When VD is less than or equal to R , indicating that the two

vehicles are within each other’s communication radii, $Encounter\ Time = 0$ and $Transfer\ Time = \frac{VD}{VS}$.

- As v_{carry} and the relay vehicle drive in opposite directions and gradually move apart, the speed difference between vehicles i and j is defined as $VS = speed_i + speed_j$. The distance between the two vehicles is described as $VD = \sqrt{D_{xij}^2 + D_{yij}^2}$. When VD is greater than R , indicating that the two vehicles are not within each other’s communication radii and will not meet again, $Encounter\ Time = \infty$ and $Transfer\ Time = 0$. When VD is less than or equal to R , indicating that the two vehicles are within each other’s communication radii, $Encounter\ Time = 0$ and $Transfer\ Time = \frac{R-VD}{VS}$. In this case, the $Transfer\ Time$ is the time spent by the two vehicles moving from their current positions until their distance reaches R because they gradually move apart.

From Figure 19, it is possible to obtain the time at which each vehicle arrives at the next intersection. As stated above, the $Encounter\ Time$ and $Transfer\ Time$ can be inferred. However, when the positions at which the two vehicles meet or their distance equal to R are not located within the road segment at which vehicle v_{carry} is located, the $Encounter\ Time$ and $Transfer\ Time$ must be updated as follows: We define the arrival times of vehicles i and j at intersection I_x as R_{xiT} and R_{xjT} . If the $Encounter\ Time$ is larger than the minimum value between R_{xiT} and R_{xjT} , i.e., $\min\{R_{xiT}, R_{xjT}\}$, the $Encounter\ Time$ is modified to ∞ . If the $Transfer\ Time$ is larger than $\min\{R_{xiT}, R_{xjT}\} - Encounter\ Time$.

As illustrated in Figure 21, road segment S_1 is composed of intersections 4 and 5, and the communication radius is $R = 300\ M$ meters. There are five vehicles within road segment S_1 . v_{carry} is denoted as V_1 . The VS and VD between V_1 and V_5 are $400\ M$ and $30\ km/h$, respectively. Hence, the $Encounter\ Time$ is $\frac{400\ M - 300\ M}{30\ km/h} = \frac{100\ M}{30\ km/h} \cong 12\ s$. Upon transmitting vehicle information to the RSU, the analysis unit of the RSU is responsible for calculating the estimated vehicle link stability within road segment S_1 , resulting in the estimation of the vehicular connection stability between v_{carry} and all vehicles on road segment S_1 . Figure 22 depicts the addition of vehicle connection stability to the road segment table, as shown in Figure 19. The time spent by V_1 and V_5 moving from their current positions to the position at which they meet and that spent by V_1 and V_5 moving from their current positions until their distance reaches R are $\frac{300\ M}{30\ km/h} = \frac{300\ M}{30\ km/h} \cong 36\ s$ and $\frac{300\ M - 0\ M}{30\ km/h} = \frac{300\ M}{30\ km/h} \cong 36\ s$, respectively. The total $Transfer\ Time$ from when V_1 and V_5 meet until their connection is broken is $36\ s + 36\ s = 72\ s$. The minimum arrival time of V_1 and V_5 at intersection I_5 is $\min\{28.8\ s, 31.5\ s\} = 28.8\ s$. Finally, the modified $Transfer\ Time$ required to finish the packet transmission within road segment S_1 is $28.8\ s - 12\ s = 16.8\ s$.

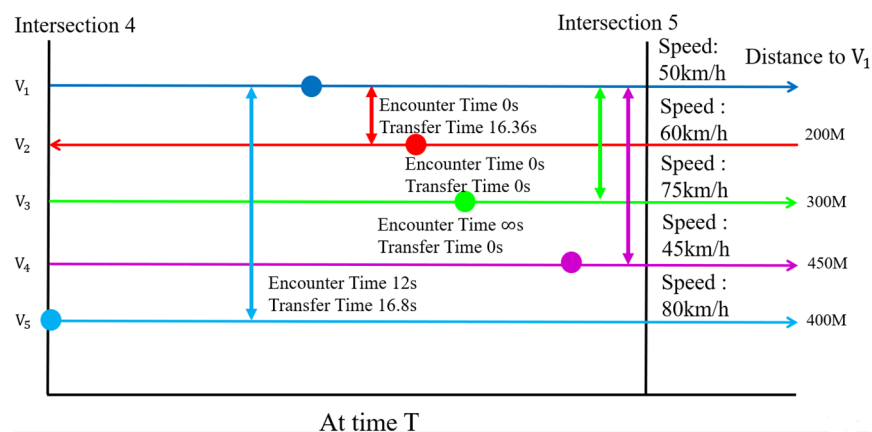


Figure 21. Diagram of the estimated vehicle connection stability for v_1 .

Road Segment Table S_1	Encounter Time to Vehicle	Transfer Time with Vehicle	Arrival Time to Next Intersection	Vehicle Direction	Next Intersection
S_1-V_1	NaN	NaN	28.8s	Same direction	Intersection 5
S_1-V_2	0s	16.36s	27s	Opposite direction	Intersection 4
S_1-V_3	0s	0s	9.6s	Same direction	Intersection 5
S_1-V_4	∞ s	0s	36s	Same direction	Intersection 5
S_1-V_5	12s	16.8s	31.5s	Same direction	Intersection 5

Figure 22. The road segment table for v_1 after adding the vehicle connection stability (the dotted lines represent the vehicles with zero *Transfer Time*).

After creating a road segment table based on the estimated vehicle link stability, the prediction unit will forecast the next intersection that each vehicle moves towards. Based on the associated RSU, intersection tables are compiled with fields recording the arrival time at the next intersection, the next intersection, the predicted intersection after the next, and the predicted arrival time to the intersection after the next. We can calculate the average speed for each road segment by utilizing the average vehicle speeds from different periods. Based on the distance between the next intersection I_i and the predicted intersection after the next I_j , we can estimate the time taken for a vehicle to travel between intersections I_i to I_j and record it as the expected arrival time to the intersection after the next field (Figure 23).

Intersection table I_1	Arrival Time to Next Intersection	Next Intersection	The predicted intersection after next	The predicted arrival time to the intersection after next
S_1-V_1	28.8s	Intersection 5		
S_1-V_5	27s	Intersection 5		
S_n-V_{984}	48s	Intersection 5	Intersection 8	39.7s
S_n-V_{498}	2.4s	Intersection 5		
S_n-V_{48}	18.4s	Intersection 5	Intersection 4	24.7s

■ ■ ■

Intersection table I_{72}	Arrival Time to Next Intersection	Next Intersection	The predicted intersection after next	The predicted arrival time to the intersection after next
$S_{72}-V_{48}$	42.1s	Intersection 70		
$S_{72}-V_{31}$	7s	Intersection 70	Intersection 67	47.1s
S_n-V_{73}	16.7s	Intersection 70	Intersection 59	40.7s
S_n-V_{734}	27.1s	Intersection 70		
S_n-V_{525}	19.7s	Intersection 70	Intersection 65	34.9s

Figure 23. The addition of the predicted intersection after the next and the expected arrival time fields to the intersection after the next into intersection tables.

Finally, we propose two cases to compute the minimum packet propagation delay of a road segment. First, based on the vehicle connection stability on the road segment table, if the link connection time between any two vehicles on a road segment is greater than or equal to the theoretical packet transmission time and any vehicle is within the communication range of its peers, the packet can be propagated from the originating intersection through vehicles within the road segment to the next intersection. In this case, the packet propagation delay on the road segment (D_S) equals the smallest number of vehicles required to forward the packet on the road segment (N) multiplied by the theoretical packet transmission time (P_T), which is formulated as $D_S = N \times P_T$. Second,

if the relay vehicle finds no neighbors within its communication range, the extra packet propagation delay M_T must be added to the *Encounter Time* required for the relay vehicle to meet the next hop vehicle. The total packet propagation delay on the road segment equals the sum of $N \times P_T$ and M_T , which is formulated as $D_S = N \times P_T + M_T$.

We can obtain vehicle transmission paths with sufficient historical data through training units using DRL (Figure 17). However, in scenarios involving real-time vehicles with insufficient data, the information from these vehicles can be incorporated into the decision-making process of the DQN model using the method proposed above for computing the minimum packet propagation delay for a road segment. Consequently, the analysis unit of the RSU can determine whether a new transmission path can be formed, as illustrated in Figure 24.

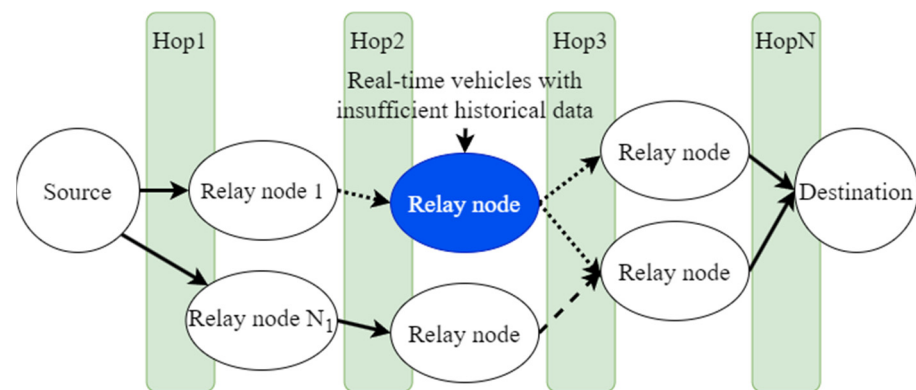


Figure 24. The DQN forwarding path considers real-time vehicles with insufficient data and vehicles with sufficient historical data.

3.2.3. Real-Time DQN Routing Decision for the Packet Transmission Path and the Exception Handling Mechanism by the Integration Unit

After the DQN decides to choose the relay node v_{relay} , the SDN CN transmits the v_{relay} information to the RSU located at the intersection. Then, the RSU notifies v_{carry} that its packet needs to be forwarded to v_{relay} . Because the real-time vehicle characteristics may differ from those in their historical datasets, v_{carry} may have no v_{relay} as its neighbor, or the calculated link connection time between v_{relay} and v_{carry} may be insufficient to transmit the packet at that time, causing packet transmission failure. In this case, the decision made by DQN is meaningless for the real-time vehicle and hence requires the DQN routing decision and exception handling mechanism to be used (Figure 25).

1. If the packet carried by v_{carry} needs to be transmitted to the destination, go to step 2. Otherwise, the DQN routing decision flow ends.
2. The DQN executed in the SDN CN first determines which relay node v_{relay} is chosen to forward the data.
3. This information is initially sent from the SDN CN to the RSU, which then forwards it to v_{carry} , thereby carrying the packet.
4. When the v_{carry} carrying the packet searches its neighbor table and discovers the presence of v_{relay} , it forwards the packet to v_{relay} and the process moves to step 5. If v_{carry} cannot find v_{relay} in its neighbor table, the process moves to step 10.
5. If the packet transmission from v_{carry} to v_{relay} is successful, go to step 6. Otherwise, go to step 7.
6. v_{carry} sends back a success packet to the RSU, notifying them that the transmission has been successful. Go to step 8.
7. If the packet transmission to v_{relay} fails, v_{carry} sends back a failure packet to the RSU, notifying them that the packet transmission has failed.
8. Subsequently, the RSU informs the SDN CN that v_{relay} is unable to complete the packet transmission.

9. The SDN CN reselects another relay node using DQN. Then, the process returns to step 1.
10. The SDN CN sends back a not found packet to the RSU, notifying it that the chosen relay node cannot be found. The RSU then tells the SDN CN to reselect another relay node by going to step 9.

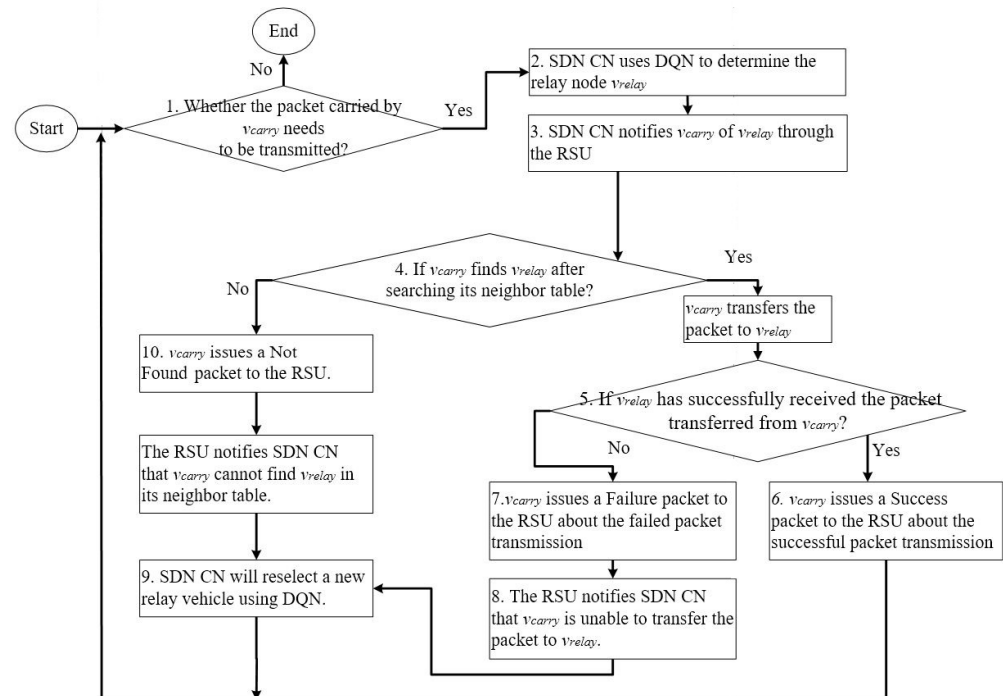


Figure 25. The flow of DQN routing decisions and exception handling processes.

3.2.4. Integration of DQN and RF Using the Real-Time Decision Unit

The dataset employed in this study comprises taxi data. The driving characteristics of taxis can vary due to unpredictable passenger destinations. As mentioned above, RF is used to determine the intersection of vehicles, while DQN determines the shortest end-to-end delay forwarding route. In real-time scenarios, vehicles with insufficient data and new vehicles may appear, and it may be challenging to make instantaneous decisions solely using RF or DQN due to the presence of untrained data. In the IDRF, the real-time vehicle decision unit integrates DQN and RF to address the abovementioned question. As illustrated in Figure 16, for the source vehicle to relay packets to the destination vehicle, relay nodes of M hops are required for packet forwarding. Packet forwarding may occur on either road segments or intersections. During the forwarding of packets on road segments, the vehicles with DQN-predicted results and the real-time new vehicles that DQN has not trained must be considered to determine the best relay vehicle with the shortest delay. When forwarding packets at intersections, the travel directions of vehicles, predicted results from DQN, and real-time vehicles must be considered concurrently. Moreover, the scenario in which the source and destination nodes are real-time vehicles without prior training by DQN must also be considered. The real-time decision unit flow is illustrated in Figure 26.

1. Determine if the destination vehicle is a real-time new vehicle. If the destination vehicle is not new, DQN is used to determine the shortest path from the source vehicle to the destination vehicle for forwarding. Then, go to step 2. Otherwise, go to step 4.
2. If the destination is a new vehicle, determine whether the destination and packet-carrying vehicles are on the same road segment. If they are, go to step 5. Otherwise, go to step 3.
3. If the destination vehicle and the packet-carrying vehicle are not on the same road segment, temporarily set the upcoming intersection of the destination vehicle as the destination intersection.

4. Use DQN to determine the shortest path from the source vehicle to the destination intersection. Then, go to step 6.
5. Forward the packet to the destination vehicle. Then, go to step 15.
6. Determine whether packet forwarding occurs on the road segment. If it does, go to step 7; if not, go to step 11.
7. Determine whether the candidate vehicles forwarding the packet in this hop have real-time new vehicles. If they do, go to step 8; if not, go to step 9.
8. If there are real-time new vehicles, recalculate the DQN delay weights for all candidate vehicles in this hop.
9. Classify the vehicle with the largest DQN delay weight at this road segment as the relay node. If there are no real-time new vehicles, there is no need to recalculate the DQN delay weights for all candidate vehicles in that hop. Instead, use the DQN delay weights to determine the optimal forwarding node.
10. Forward the packet to the selected relay node. Then, go to step 15.
11. If packet forwarding occurs at an intersection, determine whether real-time new vehicles are in this forwarding hop. If there are real-time new vehicles, go to step 12. Otherwise, go to step 13.
12. Calculate the DQN delay weights for all vehicles in that hop.
13. Multiply the DQN delay weight by the RF intersection transfer probability to compute the transfer delay weight. If there are no real-time new vehicles in this forwarding hop, calculate the transfer delay weight using the original DQN delay weight multiplied by the RF intersection transfer probability.
14. Classify the vehicle with the largest transfer delay weight at this road segment as the relay node.
15. Determine whether the packet has been forwarded to the destination vehicle. If it has, the flow of the real-time decision unit has finished. If not, return to step 1.

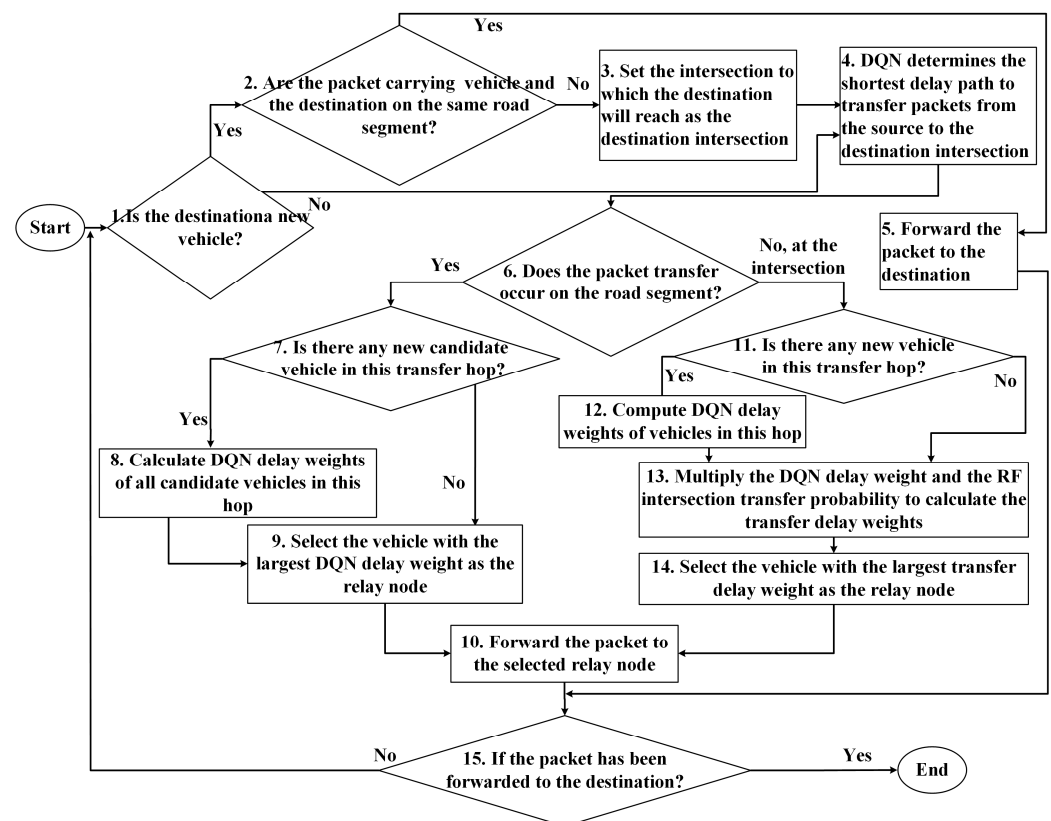


Figure 26. The flow of the real-time decision unit in the IDRF.

As illustrated in Figure 27a, assuming that both the source and destination vehicles have been trained, DQN is used to determine the forwarding path with the shortest end-to-end delay from the source to the destination. In this scenario, v_4 and v_{10} represent real-time new vehicles, while the remaining vehicles are relay nodes trained by DQN. Hence, it is necessary to determine whether a new packet forwarding path can be formed by v_4 and v_{10} . The estimated connection stability of the inter-vehicle connections can be used to determine whether a new link can be established between two vehicles. It is assumed that v_{10} cannot form a new link, but v_4 can connect to S and V_7 , as illustrated in Figure 27b. After adding v_4 to the original DQN path, three new packet forwarding paths can be established: $S \rightarrow V_3 \rightarrow V_6 \rightarrow V_9 \rightarrow D$, $S \rightarrow V_4 \rightarrow V_7 \rightarrow V_{11} \rightarrow D$, and $S \rightarrow V_4 \rightarrow V_7 \rightarrow V_{12} \rightarrow D$. Since v_4 is a newly introduced real-time vehicle, it lacks the DQN-trained weight. The reward between v_4 and its neighbor vehicles is calculated using Equation (5) and is used as the weight of the corresponding hop, as presented in Table 2. First, we calculate the total delay weights for the three paths without considering intersection transfer probabilities. Since path 1 has the highest total delay weight, it is selected as the packet forwarding route, as illustrated in Table 3. Further, packets at Hop 1, Hop 2, and Hop 4 in Figure 27b are transferred to the relay vehicles on road segments. Intersection transfer probabilities calculated by RF are not used at these hops. However, because packets at Hop 3 are transferred to the relay vehicles at intersections, new total delay weights must be calculated for these three packet forwarding paths while considering the intersection transfer probabilities. It is assumed that the intersection transfer probabilities for vehicles at Hop 3 are those shown in Table 4. First, the new delay weight H_w is calculated by multiplying the original DQN delay weight D_w by the RF intersection transfer probability R_w (Equation (9)). The results are presented in Table 5. Next, the IDRf is used to calculate the new total delay weights of these three packet forwarding paths (Table 6). Since Path 3 has the highest total delay weight, the IDRf selects it as the new packet forwarding path. This example demonstrates that without considering the real-time vehicle v_4 and intersection transfer probabilities calculated by RF, Path 1 is chosen as the packet forwarding path instead of Path 3, as it has the highest total delay weight, which means that it has the shortest end-to-end delay.

$$H_w = D_w \times R_w \tag{9}$$

Table 2. The hop delays and DQN delay weights D_w for each hop.

Link	Hop 1 Delay/DQN Delay Weight	Link	Hop 1 Delay/DQN Delay Weight
$S \rightarrow V_1$	1.40 s/0.0702	$V_1 \rightarrow V_5$	0.70 s/0.5388
$S \rightarrow V_2$	0.40 s/0.2463	$V_2 \rightarrow V_5$	1.70 s/0.2218
$S \rightarrow V_3$	0.17 s/0.5789	$V_3 \rightarrow V_6$	2.60 s/0.1450
$S \rightarrow V_4$	0.94 s/0.1046	$V_4 \rightarrow V_7$	4.00 s/0.0944
Link	Hop 3 DQN Delay Weight	Link	Hop 4 DQN Delay Weight
$V_5 \rightarrow V_8$	0.0491	$V_9 \rightarrow D$	0.1420
$V_6 \rightarrow V_9$	0.4270	$V_{11} \rightarrow D$	0.4806
$V_7 \rightarrow V_{11}$	0.1723	$V_{12} \rightarrow D$	0.4615
$V_7 \rightarrow V_{12}$	0.3516		

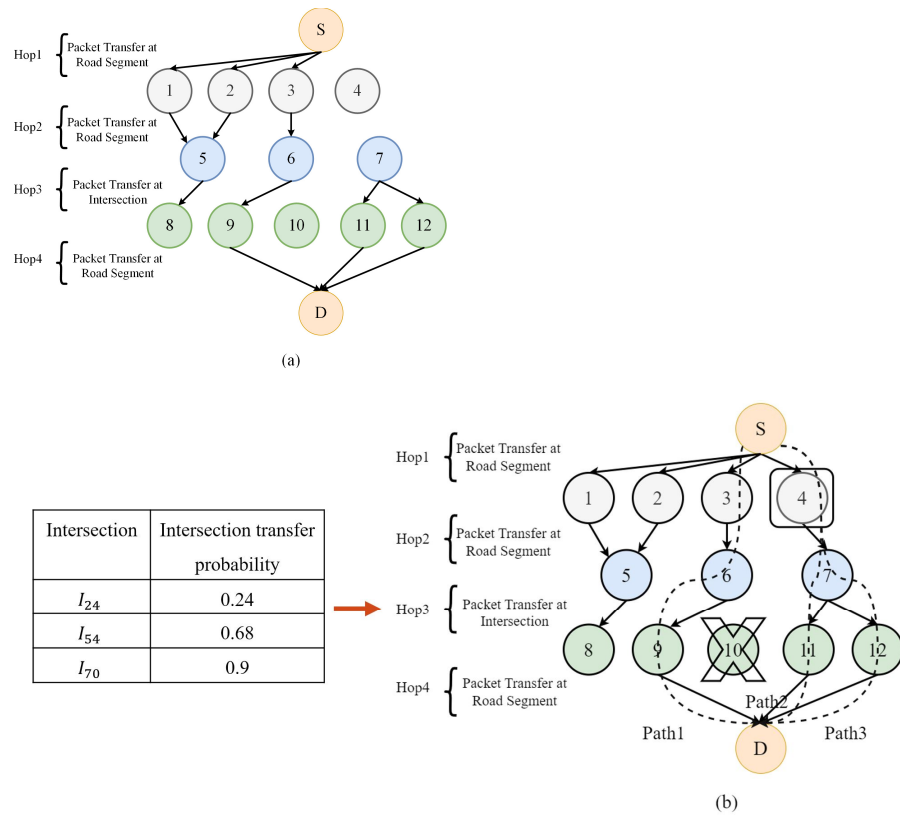


Figure 27. (a) The DQN-decided packet forwarding path from the source to the destination; (b) IDRf real-time vehicle decision that integrates DQN and RF (S and D denote the source and destination vehicles, the arrow indicates the intersection transfer probability of Hop3, and the dotted line are possible paths between the source and the destination).

Table 3. Total delay weights of three packet forwarding paths without considering the intersection transfer probabilities.

	End-to-End Forwarding Path	Total Delay Weights
Path 1	$S \rightarrow V_3 \rightarrow V_6 \rightarrow V_9 \rightarrow D$	$0.5789 + 0.1450 + 0.4270 + 0.1420 = 1.2929$
Path 2	$S \rightarrow V_4 \rightarrow V_7 \rightarrow V_{11} \rightarrow D$	$0.1046 + 0.0944 + 0.1723 + 0.4806 = 0.8519$
Path 3	$S \rightarrow V_4 \rightarrow V_7 \rightarrow V_{12} \rightarrow D$	$0.1046 + 0.0944 + 0.3516 + 0.4615 = 1.0121$

Table 4. The intersection transfer probability R_w for Hop 3.

Intersection	Intersection Transfer Probability
I_{24}	0.24
I_{54}	0.68
I_{70}	0.90

Table 5. The new delay weight H_w for Hop 3.

Link for Hop 3	IDRF Delay Weight of Hop 3
$V_6 \rightarrow V_9$	$0.4270 \times 0.24 = 0.102480$
$V_7 \rightarrow V_{11}$	$0.1723 \times 0.68 = 0.117164$
$V_7 \rightarrow V_{12}$	$0.3516 \times 0.90 = 0.316440$

Table 6. The total delay weights for the three packet forwarding paths calculated by the IDRf.

	End-to-End Forwarding Path	IDRF Total Delay Weights
Path 1	$S \rightarrow V_3 \rightarrow V_6 \rightarrow V_9 \rightarrow D$	$0.5789 + 0.1450 + 0.102480 + 0.1420 = 0.968380$
Path 2	$S \rightarrow V_4 \rightarrow V_7 \rightarrow V_{11} \rightarrow D$	$0.1046 + 0.0944 + 0.117164 + 0.4806 = 0.796764$
Path 3	$S \rightarrow V_4 \rightarrow V_7 \rightarrow V_{12} \rightarrow D$	$0.1046 + 0.0944 + 0.316440 + 0.4615 = 0.976940$

4. System Simulations and Performance Evaluations

4.1. Simulation Environment

The IDRf_DQN, IDRf_DQN_RF, and IDRf algorithms introduced in this paper for decision-making were compared in terms of their performances with TDRL-RP, focusing on vehicle training for decision-making, VRDRT, focusing on road information-based decision-making, and traditional VANET routing algorithms, Epidemic and GPSR, for packet routing. Epidemic was chosen due to its broadcast-based packet forwarding method, which enables the source vehicle to obtain multiple forwarding paths from a single set of pairs. VRDRT, on the other hand, utilizes vehicle positions, distances to neighboring vehicles, and vehicle density statistics information from the dataset. Simulations were conducted using Python and NS-3. Python interacts with NS-3 and was used for DQN training, while NS-3 was used for a network simulation with environmental parameters (Table 7). In this simulation, the MAC protocol used in NS-3 was IEEE 802.11p, and the radio propagation model used was the Log Distance Propagation Loss Model. As mentioned in Section 3.1, NS-3 employed real vehicle trajectory data from Beijing city's taxi dataset, encompassing the GPS trajectories of 10,357 taxis on 2–8 February 2008. The map used in this study includes latitudes 39.89851 to 39.94923 and longitudes 116.36701 to 116.41253, forming a map with dimensions of 3880 by 5636 m. Regarding the dataset, the proposed IDRf_DQN, IDRf_DQN_RF, IDRf, and TDRL-RP algorithms were evaluated against the Epidemic algorithm using the following default simulation parameters: randomly selected pairs of 32, a TTL of 90, a message time of 1.0, and a transmission range of 425. This resulted in 1000 simulations, which were used to generate forwarding paths for the actual vehicle trajectory data from the Beijing city taxi dataset. The x -axis of the result showed the number of source-destination pairs and the transmission range. The y -axis of the simulation results showed the average packet delivery ratio, average end-to-end delay, and average overhead ratio, defined as Equations (10)–(12). The results from 1000 simulation runs were averaged. The figures were drawn with 95% confidence intervals.

- Average end-to-end delay

This paper defines the average end-to-end delay as follows: It is the average delay when all packets are transmitted from the source vehicle to the destination vehicle across all forwarding paths. This average delay accounts for the total delay time of all delivered packets, divided by the total number of delivered packets received by the destination vehicle. The calculation of the average end-to-end delay is performed using Equation (10).

$$\text{Average end-to-end delay} = \frac{\text{Total end-to-end delays of all delivered packets}}{\text{Number of delivered packets}} \quad (10)$$

- Average packet delivery ratio

In this study, the average packet delivery ratio is defined as the number of delivered packets received by the destination divided by the number of packets transmitted by the source, calculated using Equation (11).

$$\text{Average Packet delivery ratio} = \frac{\text{Number of delivered packets}}{\text{Number of transmitted packets by source}} \quad (11)$$

- Average overhead ratio

In this study, the average overhead ratio is the sum of the total number of packets that need to be transmitted through each relay node j in the forwarding path of packet I from the source vehicle to the destination vehicle, including the data packets sent by the source vehicle and the control packets sent by various routing protocols, denoted as np_j^i . The overhead for the forwarding path of packet i is defined as $\sum_{j=1}^h np_j^i$, where h is the total number of relay nodes, i.e., hop counts, between the source and the destination vehicles. Next, the overheads of all packets transmitted by the source vehicle are summed and divided by the total number of transmitted packets by the source vehicle (Equation (12)).

$$\text{Average Overhead ratio} = \frac{\sum_{i=1}^{\text{Number of transmitted packets by source}} \sum_{j=1}^h np_j^i}{\text{Number of transmitted packets by source}} \quad (12)$$

Table 7. NS-3 Simulation Parameters.

Parameter	Parameter Value
Map size	3880 M × 5636 M
Simulation time	12:00–13:00 (3600 s for congestion periods), 13:00–17:00 (7200 s for sparse periods)
MAC protocol	IEEE 802.11p
Radio propagation model	Log Distance Propagation Loss Model
Packet size	1024 Bytes
Buffer size	10 MBytes
Number of trained vehicles	8456 (congestion periods), 3481 (sparse periods)
Number of untrained vehicles	1212 (congestion periods), 584 (sparse periods)
Transmission range (M)	300, 425 (default value), 550, 675
Number of pairs	4, 8, 16, 32 (default value)
Message Time	1
TTL	90
Learning rate α	0.001 (default value), 0.05, 0.01
Discount factor γ	0.9, 0.95, 0.99 (default value)
RF threshold	Gini

4.2. DQN Hyperparameter Configuration and Testing

In the following section, we discuss the hyperparameter configuration used for DQN training in this paper. The vehicle trajectory data of the Beijing city taxi dataset used for training is collected, as described in Section 3.1. The vertical axis primarily focuses on the cumulative reward, defined by Equation (13), while the horizontal axis represents training epochs. This paper defines the cumulative reward as the average of all packet forwarding paths. To begin, the cumulative reward of packet i in the forwarding path of packet i is calculated. Whenever the packet passes through relay node j in the forwarding path of packet i , a reward $Reward_j^i$ is obtained. Thus, $\sum_{j=1}^h Reward_j^i$, where h is the number of relay nodes traversed by packet i along its forwarding path from the source to the destination. Subsequently, the sum of the rewards across all packet forwarding paths

is computed as $\sum_{i=1}^{\text{Number of transmitted packets by source}} \sum_{j=1}^h \text{Reward}_j^i$. Finally, the cumulative reward is determined using Equation (13).

$$\text{Cumulative reward} = \frac{\sum_{i=1}^{\text{Number of transmitted packets by source}} \sum_{j=1}^h \text{Reward}_j^i}{\text{Number of transmitted packets by source}} \quad (13)$$

Figures 28 and 29 illustrate the cumulative rewards obtained from training for the sparse and congested periods with three different learning rates (0.001, 0.005, 0.01). For sparse and congested periods, a learning rate of 0.01 achieves the highest cumulative reward at epochs 3000 and 5000, respectively. However, declines in the cumulative rewards appear as the epochs progress to 12,000 and 15,000, respectively, indicating that an excessively high learning rate of 0.01 is detrimental to the environment. With a learning rate of 0.001, the highest cumulative rewards are achieved stably after epochs 8000 and 11,000, respectively. Comparing the outcomes with learning rates of 0.005 and 0.001, the latter yields higher cumulative rewards, albeit demanding more extensive training durations.

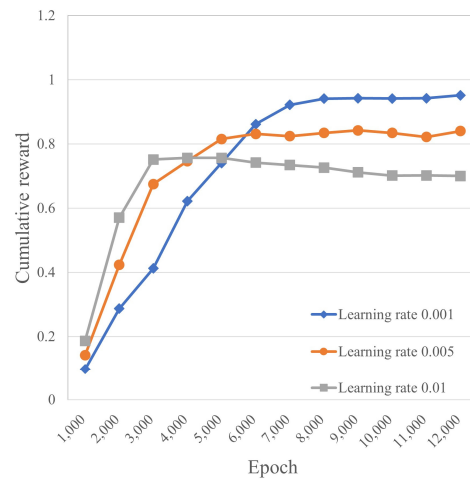


Figure 28. Cumulative rewards of different learning rates for the sparse period.

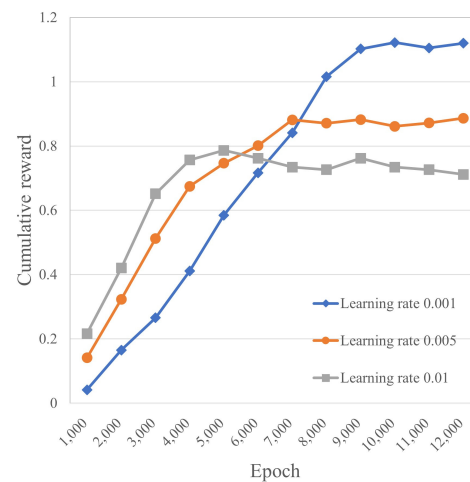


Figure 29. Cumulative rewards of different learning rates for the congested period.

Figures 30 and 31 depict the cumulative rewards from training for the sparse and congested periods with three different discount factors (0.9, 0.95, and 0.99). For the sparse and congested periods, a discount factor of 0.9 achieves the highest cumulative reward at epochs 4000 and 5000 and remains stable as the epochs advance to 12,000 and 15,000, respectively. This indicates that a lower discount factor of 0.9 leads to quicker cumulative

reward accumulation but limits the peak value. A discount factor of 0.95 yields the highest cumulative rewards during epochs 6000–7000 and 7000–8000, maintaining stability as the epochs advance to 12,000 and 15,000, respectively. Meanwhile, a discount factor of 0.99 attains the highest cumulative rewards at epochs 8000 and 9000, and the cumulative rewards remain stable as the epochs progress to 12,000 and 15,000, respectively. Thus, a discount factor of 0.99 is better suited for this environment and dataset. By using the strategy that selects higher long-term rewards, although the cumulative rewards accumulate more slowly, the overall cumulative value increases. This provides more consistent cumulative rewards despite the longer training durations required, lower learning rates, and higher discount factors. Consequently, a learning rate of 0.001 and a discount factor of 0.99 were applied to subsequent simulations.

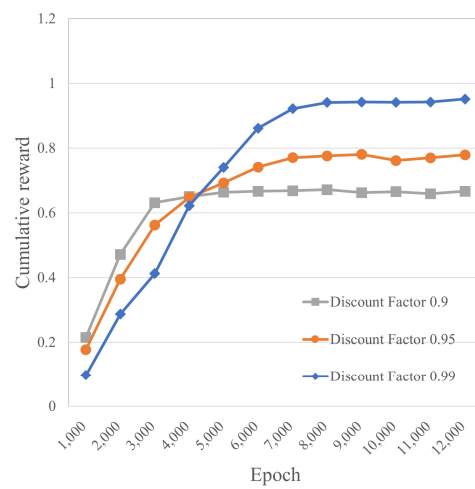


Figure 30. Cumulative rewards with different discount factors for the sparse period.

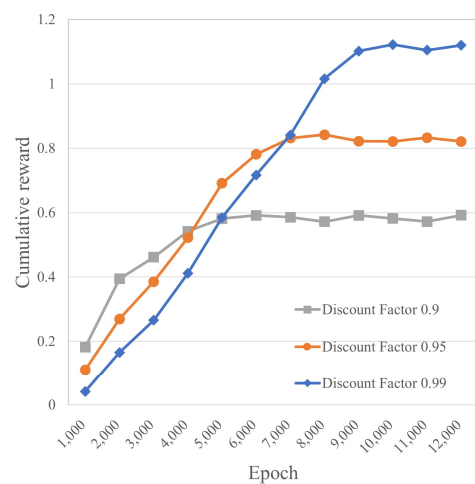


Figure 31. Cumulative rewards with different discount factors for the congested period.

4.3. Different Numbers of Source-Destination Pairs for the Sparse and Congested Periods

Figures 32–37 show the simulation results for the sparse and congested periods. The simulations involve four source-destination pairs (4, 8, 16, 32). They demonstrate the impact on the average packet delivery ratio, average end-to-end delay, and average overhead ratio across all routing methods. The source and destination vehicles were randomly selected for this number of pairs from all of the vehicles. As the number of pairs increases, the network experiences a higher concurrent packet transmission load, leading to a decline in the average packet delivery ratio for all routing algorithms and an increase in both the average end-to-end delay and the average overhead ratio.

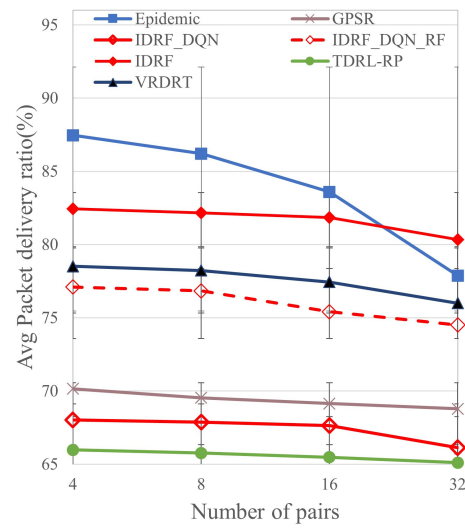


Figure 32. The average packet delivery ratio with different numbers of pairs for sparse periods.

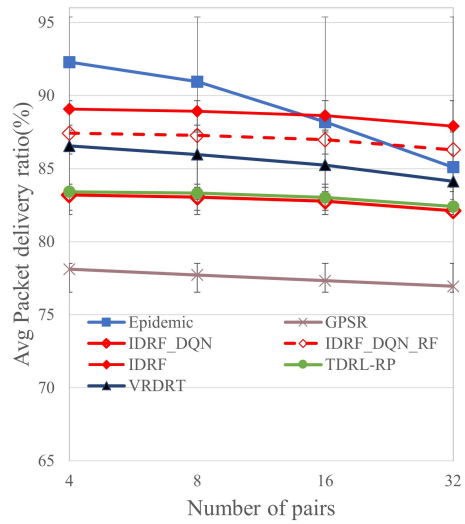


Figure 33. The average packet delivery ratio with different numbers of pairs for congested periods.

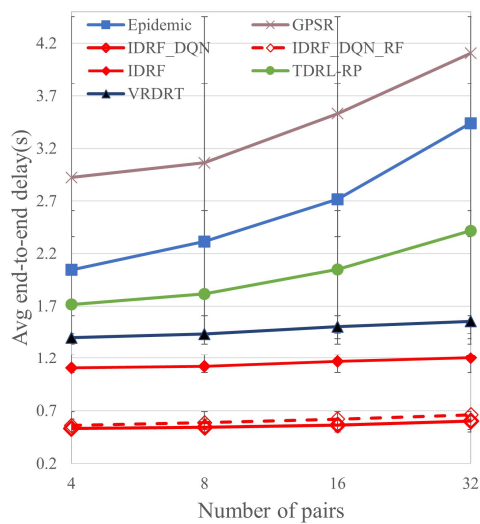


Figure 34. The average end-to-end delay with different numbers of pairs for sparse periods.

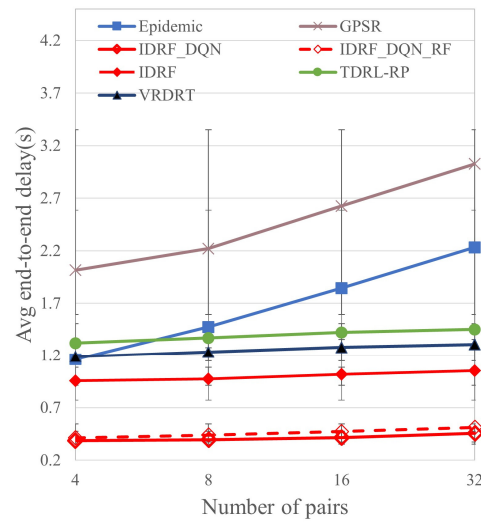


Figure 35. The average end-to-end delay with different numbers of pairs for congested periods.

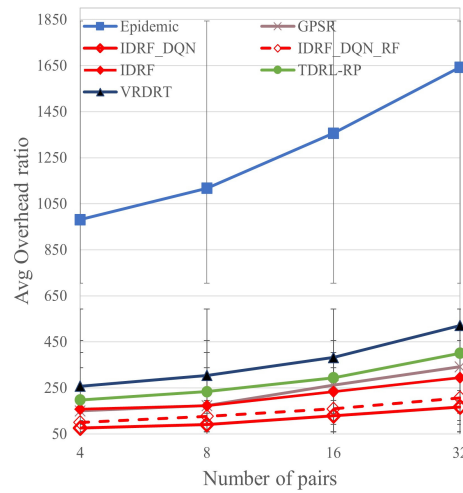


Figure 36. The average overhead ratio with different numbers of pairs for sparse periods.

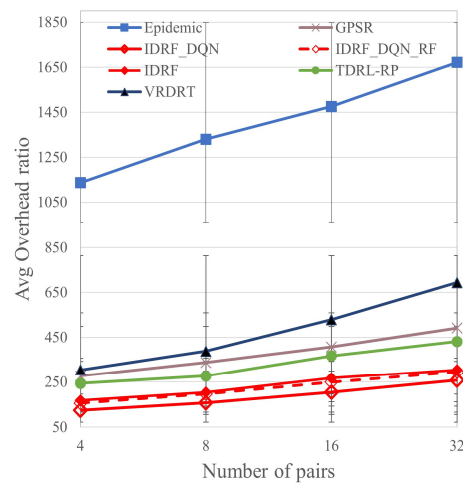


Figure 37. The average overhead ratio with different numbers of pairs for congested periods.

Figures 32 and 33 illustrate the average packet delivery ratio with different numbers of pairs for the sparse and congested periods, respectively. Epidemic disseminates packets through broadcasting, and under the assumption of negligible overhead, it can identify

more transmission paths in an unknown environment. However, as the number of pairs increases, more packets exist concurrently in the network environment, increasing packet collisions and resulting in a lower average packet delivery ratio. GPSR only considers the distance between the current packet and the destination vehicle. It may select relay nodes that cannot reach the destination vehicle during forwarding. With TDRL-RP and VRDRT, no additional processing is performed when untrained real-time vehicles are incorporated, causing both algorithms to have suboptimal average packet delivery ratios. TDRL-RP employs the current vehicle position as input but fails to exhibit good adaptability to dynamic environmental changes. If the current environment differs significantly from the trained environment, such as when the current vehicle position deviates from the trained one, the decisions made by TDRL-RP fall below the performance of GPSR, even in sparse periods. Although VRDRT does not handle untrained real-time vehicles, its training process references the current vehicle positions, the distance between the RSU and a vehicle, and the vehicle density. Consequently, for sparse periods, VRDRT still outperforms GPSR, TDRL-RP, IDRF_DQN, and IDRF_DQN_RF in terms of the average end-to-end delay. VRDRT has a higher average end-to-end delay for congested periods than IDRF_DQN.

The proposed IDRF_DQN algorithm only uses DQN to determine which relay nodes to use for packet forwarding without considering the approaching intersections of the possible relay vehicles and real-time new vehicles. It consistently relays packets to the vehicle determined by the DQN decision to have the highest delay weight, causing its performance to surpass that of TDRL-RP only for sparse periods and that of GPSR exclusively for congestion periods. In IDRF_DQN_RF, the combination of DQN and RF identifies new relay paths. DQN is utilized to train the weights of relay nodes, while RF recalculates the probabilities of approaching intersections. This addresses issues such as consistently forwarding packets to the vehicle with the highest delay weight calculated by DQN, the possibility for the relay to carry packets for a long time, and incomplete packet transmission between vehicles. However, IDRF_DQN_RF does not account for potential improvements in packet relay paths arising from newly introduced real-time vehicles. Therefore, the IDRF framework introduced in this study executes relay vehicle decision-making by considering new vehicles in real time, end-to-end delays, and inter-vehicle connection stabilities in the real-time phase. By combining the integrated RF and DQN models in the offline phase to overcome the abovementioned constraints, the IDRF determines the optimal packet relay paths based on the real-time VANET environment. For sparse periods, with 32 pairs, the average packet delivery ratio of IDRF surpasses that of Epidemic. Notably, for congestion periods with 16 and 32 pairs, the average packet delivery ratios of the IDRF outperform those of Epidemic.

Figures 34 and 35 illustrate the average end-to-end delays for different numbers of pairs for sparse and congested periods, respectively. Epidemic disseminates packets through broadcasting, which could yield the optimal packet forwarding path with the shortest end-to-end delay in an unknown VANET environment. Since there are multiple forwarding paths, Epidemic calculates the average delay for all possible paths through which packets may ultimately be received by the destination vehicle. This aggregation contributes to the relatively higher performance in terms of the average end-to-end delay observed in Epidemic. GPSR, which does not consider how vehicles carry packets, exhibits the poorest average end-to-end delay performance. TDRL-RP and VRDRT, which do not address the handling of untrained real-time vehicles, may result in prolonged packet-carrying delays during transmission to such vehicles.

As mentioned above, the IDRF_DQN approach exclusively prioritizes vehicles with the highest delay weights, namely those associated with the packet forwarding path with the shortest end-to-end delay. Therefore, IDRF_DQN yields the lowest average end-to-end delay for packets received among all methods. IDRF_DQN_RF considers both the DQN for selecting vehicles with the highest delay weights and the RF for determining vehicle movement direction, leading to a slightly higher performance in terms of the average end-to-

end delay compared to IDRf_DQN, yet its performance remains lower than other methods. IDRf is built upon the foundation of IDRf_DQN_RF and further considers new forwarding paths consisting of untrained real-time vehicles. When compared to the optimal propagation paths achieved through trained vehicles, IDRf exhibits a slightly higher average end-to-end delay than IDRf_DQN and IDRf_DQN_RF, but it still outperforms the other methods.

Figures 36 and 37 show the average overhead ratio with different numbers of pairs for sparse and congested periods. Packets are disseminated through broadcasting using the Epidemic protocol. As the number of pairs increases, the number of replicated packets in the VANET environment also increases, resulting in the highest average overhead ratio. Although GPSR only considers the distance relationship between the vehicle carrying the packet and the destination vehicle, it performs better than VRDRT. Given that VRDRT requires the retrieval of vehicle information in each time frame, its increasing average overhead ratio is notable, causing it to exhibit a higher average overhead ratio compared to other protocols, except for Epidemic. Meanwhile, TDRL-RP continually uses trial and error to train different transmission paths during packet forwarding. Consequently, its average overhead ratio is also higher than those obtained by the methods proposed in this paper, namely IDRf_DQN, IDRf_DQN_RF, and IDRf.

IDRf_DQN achieves the lowest average overhead ratio among the compared methods as it focuses solely on employing DQN to make packet relay decisions. However, its performance regarding the average packet delivery ratio is unsatisfactory. IDRf_DQN_RF enhances this by incorporating RF decisions to determine the vehicle movement direction in cooperation with DQN-based packet relay decisions. Although this is better than the strategy of constantly relaying packets to the vehicle with the highest DQN delay weight, the resulting forwarding path might not be the shortest, leading to a slightly higher average overhead ratio than that obtained with IDRf_DQN. Enhancing IDRf_DQN_RF, the IDRf approach considers untrained real-time vehicles. Despite using the analysis unit in the real-time phase to gather real-time vehicle information, the average overhead ratio of the IDRf remains higher than that of IDRf_DQN_RF due to the necessity of traversing more hops in unknown real-time environments. It is noteworthy, however, that even though the IDRf exhibits the highest average overhead ratio among the three methods proposed in this paper, it still performs better regarding this aspect compared to other relevant research approaches.

By comparing the data for different numbers of pairs for sparse and congested periods, it can be observed that when the number of pairs is set to 32, the average packet delivery ratio of the IDRf surpasses that of VRDRT during both sparse and congested periods. This improvement is remarkable, as the IDRf outperforms all other methods. Furthermore, when the number of pairs is set to 32 for the congested period, the average packet delivery ratio of IDRf_DQN_RF also exceeds that of VRDRT. Additionally, the three methods introduced in this paper, IDRf_DQN, IDRf_DQN_RF, and IDRf, consistently demonstrate lower average end-to-end delays and overhead ratios than other methods. When the performance results of Epidemic represent a score of 100%, those of all other routing protocols can be normalized accordingly. The statistical analysis presented in Table 8 indicates that the IDRf achieves significant performance enhancements over TDRL-RP and VRDRT for both sparse and congested periods.

Table 8. Average performance improvements obtained with the IDRf over TDRL-RP and VRDRT with different numbers of pairs for sparse and congested periods.

Sparse Period	IDRf over TDRL-RP	IDRf over VRDRT	Congested Period	IDRf over TDRL-RP	IDRf over VRDRT
Average packet delivery ratio	+19.24%	+4.97%	Average packet delivery ratio	+6.27%	+3.56%
Average end-to-end delay	−32.00%	−12.73%	Average end-to-end delay	−24.57%	−15.86%
Average overhead ratio	−5.14%	−11.68%	Average overhead ratio	−6.59%	−16.68%

4.4. Different Transmission Ranges for Sparse and Congested Periods

Figures 38–43 depict the simulation results for sparse and congested periods with four transmission ranges (300, 425, 550, and 675). The results are used to analyze the impacts on the average packet delivery ratio, average end-to-end delay, and average overhead ratio of all routing algorithms. There are more vehicles during congested periods than in sparse periods, leading to more available transmission nodes. Compared to sparse periods, the routing algorithms’ average packet delivery ratios increase in congested periods, while the average end-to-end delays and overhead ratios decrease. The average packet delivery ratios for all routing algorithms rise in both sparse and congested periods. In contrast, the average end-to-end delays and overhead ratios decrease as the transmission range increases, which increases the ability to select more vehicles for packet transmission.

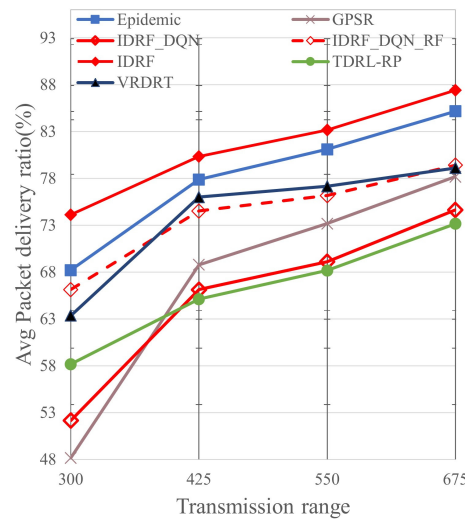


Figure 38. The average packet delivery ratio with different transmission ranges for sparse periods.

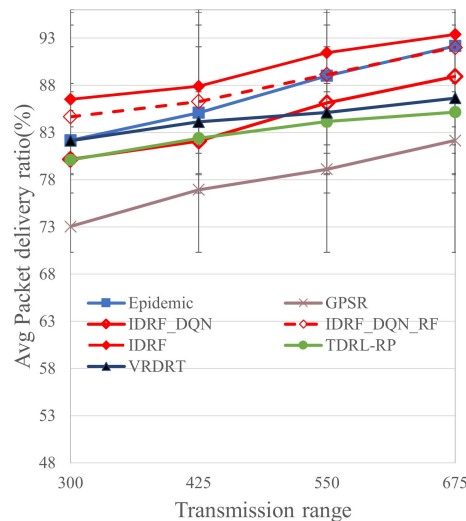


Figure 39. The average packet delivery ratio with different transmission ranges for congested periods.

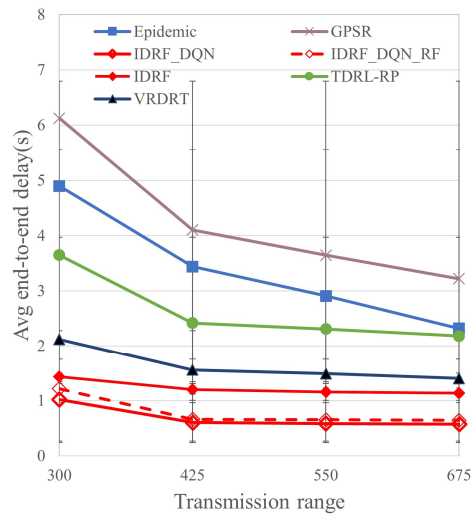


Figure 40. The average end-to-end delay with different transmission ranges for sparse periods.

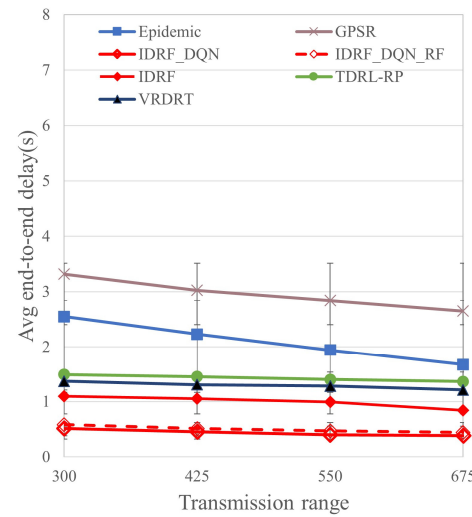


Figure 41. The average end-to-end delay with different transmission ranges for congested periods.

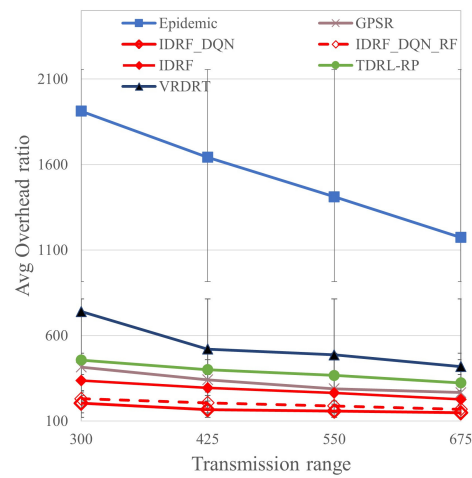


Figure 42. The average overhead ratio with different transmission ranges for sparse periods.

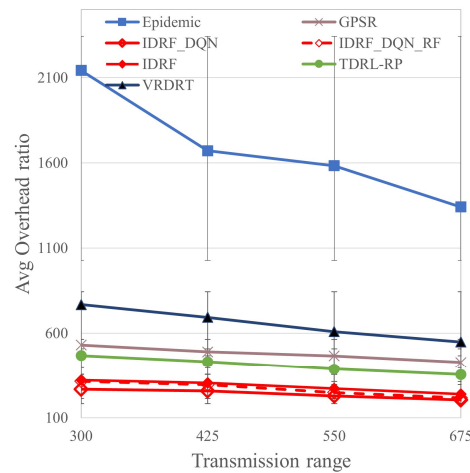


Figure 43. The average overhead ratio with different transmission ranges for congested periods.

The results collected with different transmission ranges for sparse and congested periods show that the IDRf outperforms all other methods regarding the average packet delivery ratio with varying transmission ranges. Moreover, with different transmission ranges for congested periods, IDRf_DQN_RF exhibits a higher average packet delivery ratio than all methods except for the IDRf. The three proposed methods in this paper, IDRf_DQN, IDRf_DQN_RF, and IDRf, all demonstrate lower average end-to-end delays and average overhead ratios than traditional routing methods. According to the statistical analysis presented in Table 9, the IDRf achieves significant performance improvements over TDRL-RP and VRDRT for both sparse and congested periods. Compared to previous observations, a notable aspect here is that with larger transmission ranges, the distance covered by each hop increases, enabling faster delivery to the destination vehicles. Additionally, in congested periods, the higher number of available vehicles for relay vehicle selection contributes to the better performance of the IDRf compared to sparse periods. In conclusion, the IDRf can maintain stability with different environmental parameters and achieve better results for the three performance metrics tested in this study.

Table 9. Average performance improvements with the IDRf over TDRL-RP and VRDRT with different transmission ranges for sparse and congested periods.

Sparse Period	IDRF over TDRL-RP	IDRF over VRDRT	Congested Period	IDRF over TDRL-RP	IDRF over VRDRT
Average packet delivery ratio	+19.54%	+9.62%	Average packet delivery ratio	+7.85%	+6.06%
Average end-to-end delay	−41.36%	−11.84%	Average end-to-end delay	−21.42%	−14.77%
Average overhead ratio	−7.08%	−16.77%	Average overhead ratio	−7.60%	−22.19%

5. Discussion

5.1. Performance Improvement Using the Corrected Trajectory Data

To evaluate performance improvements using the corrected trajectory data over the original data, a simulation was conducted with Epidemic using different numbers of pairs for sparse periods. Epidemic disseminates packets through broadcasting to identify the most transmission paths in VANET. Figures 44–46 illustrate Epidemic’s average packet delivery ratio, end-to-end delay, and overhead ratio with four and eight pairs of source and destination vehicles. These results show that, when using the corrected trajectory data, Epidemic achieves a 20% higher average packet delivery ratio, a 0.9 s lower average end-to-end delay, and a lower average overhead ratio (by 265) than the values achieved using the original trajectory data. Without a loss of generality, other routing schemes also perform better when using the corrected trajectory data.

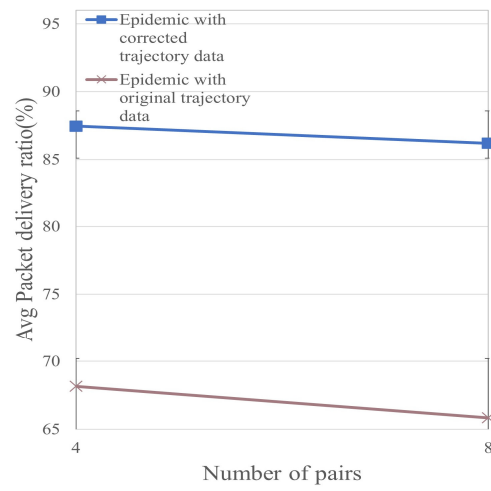


Figure 44. The average packet delivery ratio with the original and corrected trajectory data for sparse periods.

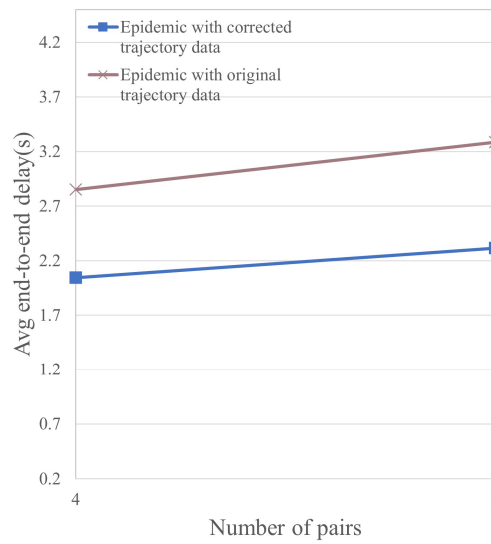


Figure 45. The average end-to-end delay with the original and corrected trajectory data for sparse periods.

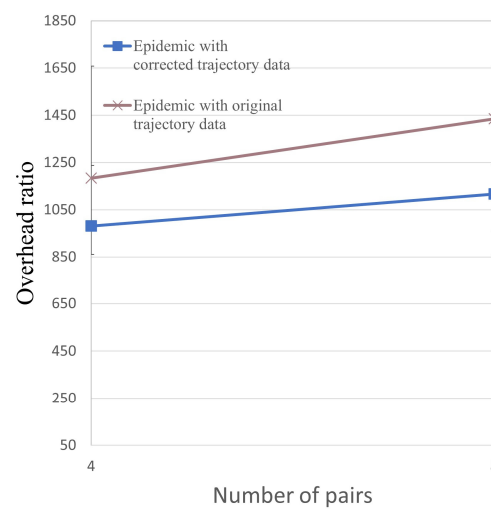


Figure 46. The average overhead ratios of the original and corrected trajectory data for sparse periods.

5.2. Four Approaches to the Correction of Vehicle Trajectory Data

Four approaches to the correction of vehicle trajectory data are presented in the literature. As mentioned above, because the historical GPS trajectory must be rectified before training the DQN model, this study adopted the approach presented in [28], whereby the roads on the map are matched by calibrating known GPS trajectory information with accurate map data. The second approach, the Differential Global Positioning System (DGPS), is an improved GPS method that references ground stations as well as GPS satellites [29]. The DGPS can be used to correct trajectory data by comparing the positions of reference stations and signal sources to correct the difference error in the trajectory data and, therefore, obtain more accurate positional information. The third approach is to smooth GPS trajectory data, which can be another feasible method to obtain more precise data [30]. The errors in the trajectory can be corrected based on historical data and real-time information from other sensors, making the trajectory closer to the actual path. Finally, gradient descent minimizes GPS errors by adjusting parameters to measure the difference between the corrected trajectory and actual geographic positions [31]. Through gradient descent, this loss function is minimized by iteratively adjusting the positions of trajectory points, gradually reducing errors. However, the latter three approaches require relevant real-time information from other reference stations or sensors to correct GPS locations, and cannot be applied to the offline phase of the proposed IDRf.

6. Conclusions and Future Work

Due to the highly dynamic characteristics of VANET networks, the IDRf based on the SDN architecture has been adopted to incorporate historical vehicle trajectories into training and account for the addition of vehicles to the forwarding path in real time. Several algorithms have been proposed for the offline phase, i.e., GPS trajectory correction by the vehicle trajectory continuity algorithm, the determination of the actual vehicle movement trajectory, the addition of arrival information at the intersection, and an estimation of the average vehicle speed on the road segment. These algorithms rectify historical trajectories and employ DQN to determine the optimal relay node and RF to determine the intersection probabilities. Additionally, this study introduces the estimation of vehicle arrival times at intersections, the vehicle link connection stability, and the delay time at the road segment based on real-time information collected by vehicles in the real-time phase. This information is organized into road segment tables and intersection tables for each road segment and intersection. DQN and RF are then recalculated using these tables, considering the addition of new vehicles in real time to estimate the delay of new vehicles and make relay node decisions for new packet forwarding paths. In simulations using the historical GPS trajectories of 10,357 taxis traveling within Beijing city, the IDRf is shown to consistently outperform other algorithms. Compared to TDRL-RP and VRDRT, the IDRf shows performance improvements for the average packet delivery ratio, end-to-end delay, and overhead ratio with different numbers of pairs, and the transmission ranges are at least 3.56%, 12.73%, and 5.14%, and 6.06%, 11.84%, and 7.08%, respectively. This is due to its ability to adapt to real-time changes in VANET networks, which leads to an improved average packet delivery ratio, reduced average end-to-end delay, and a lower average overhead ratio for both sparse and congested periods. Thus, the IDRf addresses the limitations of traditional routing protocols such as TDRL-RP and VRDRT associated with the handling of real-time VANET networks.

In the future, the following research issues will be addressed: First, how can DQN parameters such as reward, learning rate α , and discount factor γ be adapted to real-time VANET environments? Second, with the progress of advanced artificial intelligence models, the IDRf will be improved to transition from the current RF to the Extremely Randomized Trees model [37] and from the current DQN to the Dueling DQN [38,39] or Double DQN [40]. The training times and simulation results of the improved IDRf will be compared to those of the original IDRf to validate this improvement.

Author Contributions: Conceptualization, I.-C.C.; Formal analysis, C.K. and I.-C.C.; Funding acquisition, C.-E.Y. and I.-C.C.; Investigation, C.-E.Y.; Methodology, Y.-S.J., C.K. and I.-C.C.; Project administration, I.-C.C.; Resources, C.-E.Y.; Software, Y.-S.J., Y.-H.H. and Y.-C.C.; Supervision, I.-C.C.; Validation, C.-E.Y.; Visualization, Y.-S.J., Y.-H.H. and Y.-C.C.; Writing—original draft, Y.-S.J.; Writing—review and editing, C.-E.Y., Y.-H.H., Y.-C.C. and I.-C.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Science and Technology Council, Taiwan, under grant number NSTC 112-2221-E-018-009. The APC was funded by NSTC 112-2221-E-018-009.

Data Availability Statement: Data are contained within this article.

Conflicts of Interest: The authors declare no conflicts of interest.

References

- Sharma, S.; Agarwal, P.; Mohan, S. Security challenges and future aspects of fifth generation vehicular ad hoc networking (5G-VANET) in connected vehicles. In Proceedings of the 3rd International Conference on Intelligent Sustainable Systems (ICISS), Thoothukudi, India, 3–5 December 2020; pp. 1376–1380.
- Hamza, I.A.; Razvan, C.V.; John, A.C.; Yusun, C. Towards fast and reliable multihop routing in VANETs. *IEEE Trans. Mobile Comput.* **2020**, *19*, 2461–2474.
- Zhang, C.; Wei, J.; Qu, S.; Huang, C.; Dai, J.; Fu, P.; Wang, Z.; Li, X. Implementation of a V2P-based VRU warning system with C-V2X technology. *IEEE Access* **2023**, *11*, 69903–69915. [[CrossRef](#)]
- 802.11p-2010; IEEE Standard for Information Technology—Local and Metropolitan Area Networks—Specific Requirements—Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 6: Wireless Access in Vehicular Environments. IEEE: New York, NY, USA, 2010.
- Al-Heety, O.S.; Zakaria, Z.; Ismail, M.; Shakir, M.M.; Alani, S.; Alsariera, H. A comprehensive survey: Benefits, services, recent works, challenges, security, and use cases for SDN-VANET. *IEEE Access* **2020**, *8*, 91028–91047. [[CrossRef](#)]
- Sedar, R.; Kalalas, C.; Vázquez-Gallego, F.; Alonso, L.; Alonso-Zarate, J. A comprehensive survey of V2X cybersecurity mechanisms and future research paths. *IEEE Open J. Commun. Soc.* **2023**, *4*, 325–391. [[CrossRef](#)]
- Chatterjee, T.; Karmakar, R.; Kaddoum, G.; Chattopadhyay, S.; Chakraborty, S. A survey of VANET/V2X routing from the perspective of non-learning and learning-based approaches. *IEEE Access* **2022**, *10*, 23022–23050. [[CrossRef](#)]
- Nazib, R.A.; Moh, S. Reinforcement learning-based routing protocols for vehicular ad hoc networks: A comparative survey. *IEEE Access* **2021**, *9*, 27552–27587. [[CrossRef](#)]
- Doddalinganavar, S.S.; Tergundi, P.; Patil, R.S. Survey on deep reinforcement learning protocol in VANET. In Proceedings of the 1st International Conference on Advances in Information Technology (ICAIT), Chikmagalur, India, 25–27 July 2019; pp. 81–86.
- Abdulhae, O.T.; Mandeep, J.S.; Islam, M.T.; Islam, S. Reinforcement-based clustering in flying ad-hoc networks for serving vertical and horizontal routing. *IEEE Access* **2023**, *11*, 143881–143895. [[CrossRef](#)]
- Yang, C.-P.; Yen, C.-E.; Chang, I.-C. A software-defined directional Q-learning grid-based routing platform and its two-hop trajectory-based routing algorithm for vehicular ad hoc networks. *Sensors* **2022**, *22*, 8222. [[CrossRef](#)]
- He, Y.; He, R.; Yu, C. Intersection-based traffic-aware routing with fuzzy Q-learning for urban VANETs. In Proceedings of the International Conference on Security, Pattern Analysis, and Cybernetics (SPAC), Chengdu, China, 18–20 June 2021; pp. 511–515.
- Jiang, S.; Huang, Z.; Ji, Y. Adaptive UAV-assisted geographic routing with Q-learning in VANET. *IEEE Commun. Lett.* **2020**, *25*, 1358–1362. [[CrossRef](#)]
- Wu, C.; Ohzahata, S.; Kato, T. Flexible, portable, and practicable solution for routing in VANETs: A fuzzy constraint Q-learning approach. *IEEE Trans. Veh. Technol.* **2013**, *62*, 4251–4263. [[CrossRef](#)]
- Ladosz, P.; Weng, L.; Kim, M.; Oh, H. Exploration in deep reinforcement learning: A survey. *Inf. Fusion* **2022**, *85*, 1–22. [[CrossRef](#)]
- Li, S.E. Deep reinforcement learning. In *Reinforcement Learning for Sequential Decision and Optimal Control*; Springer Nature: Singapore, 2023; pp. 365–402.
- Zhang, D.; Yu, F.R.; Yang, R. A machine learning approach for software-defined vehicular ad hoc networks with trust management. In Proceedings of the 2018 IEEE Global Communications Conference (GLOBECOM), Abu Dhabi, United Arab Emirates, 9–13 December 2018; pp. 1–6.
- Saravanan, M.; Ganeshkumar, P. Routing using reinforcement learning in vehicular ad hoc networks. *Comput. Intell.* **2020**, *36*, 682–697. [[CrossRef](#)]
- Genuer, R.; Poggi, J.M. *Random Forests with R*; Springer Nature: Cham, Switzerland, 2020.
- Shafin, S.S.; Ahmed, M.M.; Pranto, M.A.; Chowdhury, A. Detection of android malware using tree-based ensemble stacking model. In Proceedings of the 2021 IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE), Brisbane, Australia, 8–10 December 2021; pp. 1–6.
- Sharma, A.; Balasubramanian, V.; Kamruzzaman, J. A temporal deep Q learning for optimal load balancing in software-defined networks. *Sensors* **2024**, *24*, 1216. [[CrossRef](#)]

22. Karmakar, G.; Chowdhury, A.; Das, R.; Kamruzzaman, J.; Islam, S. Assessing trust level of a driverless car using deep learning. *IEEE Trans. Intell. Transp. Syst.* **2021**, *22*, 4457–4466. [CrossRef]
23. Luo, L.; Sheng, L.; Yu, H.; Sun, G. Intersection-based V2X routing via reinforcement learning in vehicular ad hoc networks. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 5446–5459. [CrossRef]
24. Zemouri, S.; Djahel, S.; Murphy, J. An altruistic prediction-based congestion control for strict beaconing requirements in urban VANETs. *IEEE Trans. Syst. Man Cybern. Syst.* **2018**, *49*, 2582–2597. [CrossRef]
25. T-Drive. Available online: <https://www.microsoft.com/en-us/research/publication/t-drive-trajectory-data-sample/?from=http://research.microsoft.com/apps/pubs/default.aspx?id=152883> (accessed on 22 April 2024).
26. Yuan, J.; Zheng, Y.; Xie, X.; Sun, G. Driving with knowledge from the physical world. In Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Diego, CA, USA, 21–24 August 2011; pp. 316–324.
27. Wang, Z.; Zhang, H.; Qian, C.; Li, B.; Cao, Y.; Jiang, M. A hybrid LSTM network for long-range vehicle trajectory prediction based on adaptive chirp mode decomposition. *IEEE Sens. J.* **2024**, *24*, 5359–5369. [CrossRef]
28. Wang, M.; Wang, J.; Song, Y. A map matching method for restoring movement routes with cellular signaling data. In Proceedings of the 2020 8th International Conference on Information Technology: IoT and Smart City (ICIT 2'0), Xi'an, China, 25–27 December 2020; pp. 94–99.
29. Günay, F.B.; Öztürk, E.; Çavdar, T.Y.; Hanay, S.; Khan, A.U.R. Vehicular ad hoc network (VANET) localization techniques: A survey. *Arch. Comput. Methods Eng.* **2021**, *28*, 3001–3033. [CrossRef]
30. Kashinath, S.A.; Mostafa, S.A.; Mustapha, A.; Mahdin, H.; Lim, D.; Mahmoud, M.A.; Mohammed, M.A.; Al-Rimy, B.A.S.; Fudzee, M.F.M.; Yang, T.J. Review of data fusion methods for real-time and multi-sensor traffic flow analysis. *IEEE Access* **2021**, *9*, 51258–51276. [CrossRef]
31. Tian, Y.; Xia, J.; Sun, Y.; Wang, X.; Du, Q.; Bai, W.; Wang, D.; Cai, Y.; Wu, C.; Li, F.; et al. Improved specular point prediction precision using gradient descent algorithm. *Adv. Space Res.* **2020**, *65*, 1568–1579. [CrossRef]
32. Weisstein, E.W. Heron's Formula. Available online: <https://mathworld.wolfram.com/> (accessed on 22 April 2024).
33. Chang, I.-C.; Hung, M.-H.; Chang, C.-R.; Yen, C.-E. NTR: An efficient trajectory-based routing protocol for the vehicular delay tolerant networks. In Proceedings of the 2017 IEEE International Conference on Systems, Man, and Cybernetics, Banff, AB, Canada, 5–8 October 2017; pp. 389–394.
34. Hillebrand, E.; Lukas, M.; Wei, W. Bagging weak predictors. *Int. J. Forecast.* **2021**, *37*, 237–254. [CrossRef]
35. Pal, A.; Dutta, P.; Chakrabarti, A.; Singh, J.P. An efficient load balanced stable multi-path routing for mobile ad-hoc network. *Microsyst. Technol.* **2022**, *28*, 561–575. [CrossRef]
36. Oubbati, O.S.; Chaib, N.; Lakas, A.; Lorenz, P.; Rachedi, A. UAV-assisted supporting services connectivity in urban VANETs. *IEEE Trans. Veh. Technol.* **2019**, *68*, 3944–3951. [CrossRef]
37. Saeed, U.; Jan, S.U.; Lee, Y.-D.; Koo, I. Fault diagnosis based on extremely randomized trees in wireless sensor networks. *Reliab. Eng. Syst. Saf.* **2021**, *205*, 107284. [CrossRef]
38. Zhan, M.; Chen, J.; Du, C.; Xu, Y. Dueling network architecture for multi-agent deep deterministic policy gradient. In Proceedings of the 2021 IEEE 4th International Conference on Computer and Communication Engineering Technology (CCET), Beijing, China, 13–15 August 2021; pp. 163–168.
39. Zhang, D.; Yu, F.R.; Yang, R. Blockchain-based distributed software-defined vehicular networks: A dueling deep Q-learning approach. *IEEE Trans. Cogn. Commun. Netw.* **2019**, *5*, 1086–1100. [CrossRef]
40. Gong, J.; Wan, Y.; Liu, Y.; Li, X.; Zhao, Y.; Wang, C.; Lin, Y.; Fang, X.; Feng, W.; Zhang, J.; et al. Reinforced MOOCs concept recommendation in heterogeneous information networks. *ACM Trans. Web arXiv* **2023**, arXiv:2203.11011. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.