Computers, Materials & Continua

Tech Science Press

Doi:10.32604/cmc.2025.059325

ARTICLE

A Task Offloading Method for Vehicular Edge Computing Based on Reputation Assessment

Jun Li[1,*], **Yawei Dong**[1], **Liang Ni**[1], **Guopeng Feng**[1] and **Fangfang Shan**[1,2]

[1]School of Computer Science, Zhongyuan University of Technology, Zhengzhou, 450007, China
[2]Henan Key Laboratory of Cyberspace Situation Awareness, Zhengzhou, 450007, China

*Corresponding author: Jun Li. Email: lijun@zut.edu.cn

Received: 04 October 2024; Accepted: 12 February 2025; Published: 16 April 2025

ABSTRACT: With the development of vehicle networks and the construction of roadside units, Vehicular Ad Hoc Networks (VANETs) are increasingly promoting cooperative computing patterns among vehicles. Vehicular edge computing (VEC) offers an effective solution to mitigate resource constraints by enabling task offloading to edge cloud infrastructure, thereby reducing the computational burden on connected vehicles. However, this sharing-based and distributed computing paradigm necessitates ensuring the credibility and reliability of various computation nodes. Existing vehicular edge computing platforms have not adequately considered the misbehavior of vehicles. We propose a practical task offloading algorithm based on reputation assessment to address the task offloading problem in vehicular edge computing under an unreliable environment. This approach integrates deep reinforcement learning and reputation management to address task offloading challenges. Simulation experiments conducted using Veins demonstrate the feasibility and effectiveness of the proposed method.

KEYWORDS: Vehicular edge computing; task offloading; reputation assessment

1 Introduction

With the advancement of Vehicular Ad Hoc Networks (VANETs), intelligent connected vehicles can access more computing resources via the Internet. Connected vehicles can access Internet services via Dedicated Short Range Communication (DSRC), such as IEEE 802.11p and C-V2X. Additionally, vehicle sensor devices such as radar and cameras enhance vehicle intelligence, necessitating more robust computing resources to support sensory computing and storage needs. Cloud computing is capable of executing complex computations. However, cloud computing is constrained by network distance and cannot offer low-latency services for vehicular tasks. Edge cloud computing is a paradigm where tasks and data are processed at the edge of the Internet whenever feasible. Vehicular edge computing, comprising vehicles, roadside units, and base stations, aims to enhance the quality of service for connected vehicles by maximizing the utilization rate of spare computing resources. In vehicular edge computing, vehicles can offload their tasks to the edge cloud for collaborative task completion. When vehicular tasks require offloading to the edge cloud, vehicles transmit task-related, location-related, and resource-related information. Decision-making units collect this data from vehicles and edge clouds, including location and resource details. Subsequently, these decision-making units resolve optimization problems related to task offloading, guiding collaborative task completion by vehicles and edge clouds.

Numerous optimization algorithms exist to address the task offloading optimization problem. However, few address the trust issues inherent in vehicular edge computing environments. Each vehicle is an individual in the distributed vehicular network environment, which may lead to undesirable misbehavior. Furthermore, vehicles may fail to provide accurate locations or intentionally withhold them when requesting computational resources for vehicular edge computing. For instance, a vehicle may manipulate its resource demand and task data to acquire additional resources during task offloading requests. Specifically, misbehavior primarily involves misrepresentation of location and task information. Location information misbehavior occurs when task offloading requests use falsified locations. Attackers may exploit location information misbehavior to compromise the vehicular edge cloud. Task information misbehavior occurs when vehicles request task offloading using falsified task information. Such misbehavior can degrade overall vehicle edge computing performance.

To address potential misbehavior during task offloading in vehicular edge computing, we propose a practical task offloading method that utilizes a behavior-based reputation mechanism to mitigate the negative impact of misbehavior. A vehicle's resource access is contingent upon its reputation value in the proposed vehicular edge computing system with a behavior-based reputation mechanism. Generally, a vehicle's reputation value can increase if it behaves appropriately during task offloading, while misbehavior will decrease reputation value. Specifically, the study addresses three problems as outlined below:

- Reputation values can be assessed. The method calculates reputation values to characterize vehicle misbehavior and quantify the honesty of past behaviors using quantifiable reputation values.
- Reputation values are reliable and can be trusted by most edge clouds in distributed environments.
- Reputation values are compatible with the offloading algorithm and can directly influence task offloading decisions and resource allocation. Integrating reputation values into the offloading decision-making process to mitigate undesirable behavior poses a challenging goal.

This paper presents a reputation-based offloading algorithm for vehicular edge computing tasks to address the issues above. Specifically, the contributions of this research are outlined as follows:

(i) This study proposes a method to quantify the reputation value of connected vehicles in vehicular edge computing. The reputation value characterizes the honesty of nodes' past behaviors in vehicular edge computing.

(ii) To ensure the reliability of the global reputation value, this study designs an EigenTrust-based reputation value updating method. After finishing each task offloading, the consistency between the edge cloud's execution and the connected vehicle's actions can be verified. The cloud center can then calculate the vehicle's global reputation value using the EigenTrust-based reputation value global calculation algorithm.

(iii) To enhance decision-making performance by mitigating misbehavior, this study designs a reputation value-based task offloading method that considers the reputation values of vehicles. The technique employs a deep reinforcement learning algorithm to address resource allocation issues and utilizes the EigenTrust-based reputation value to combat misbehavior.

(iv) This study conducts simulation experiments to validate the proposed reputation-based task offloading method. Simulation evaluations demonstrate the deep reinforcement learning-based task offloading algorithm's significant effectiveness for vehicular edge computing. Additionally, the reputation-based reinforcement learning task offloading algorithm effectively mitigates misbehavior's ability to acquire resources.

The rest of this paper is organized as follows: Section 2 briefly introduces the related work. Section 3 presents the system architecture model. Section 4 describes our method in detail. Section 5 explains the experimental setting and compares it with existing methods. Section 6 concludes. Section 7 discusses future work about task offloading that considers vehicle reputation.

2 Related Work

To improve the performance of vehicular edge computing, there are many studies devoted to solving the problems of task offloading [1–3], resource allocation [4], joint path planning [5], and other task offloading problems. Zhang et al. [6] proposed a cloud-based hierarchical vehicular edge computing framework for offloading services. That study modeled resource allocation among vehicular edge computing services as a Stackelberg game to optimize. Bonab et al. [7] proposed a joint radio resource allocation and Mobile Edge Computing (MEC) optimization algorithm in a multi-layer NOMA HetNet, the algorithm aimed to maximize the system's energy efficiency. Huang et al. proposed an offloading scheme based on vehicular communication traffic control [8], which proposed a software-defined network (SDN)-based mobile edge computing architecture. Wang et al. [9] proposed a decentralized computation offloading approach to ensure fairness among edge devices in a fully decentralized environment. Feng et al. [4] proposed an autonomous vehicular edge framework to efficiently manage vehicle resources, which designed a scheduling algorithm based on ant colony optimization to solve the resource allocation problem for inter-vehicle computing workloads. Liu et al. [5] proposed a mobile edge mechanism, which deploys a vehicle-edge (V-edge) and aims to maximize V-edge tasks with sensitive deadlines. Li et al. [10] proposed a dynamic adaptive workload offloading algorithm based on Lyapunov theories and an FC-LSTM based schedule determining algorithm to balance the workload of different cloudlets and minimize the weighted average energy and time consumption of mobile devices. More artificial intelligence-based task offloading algorithms have emerged to improve task offload optimization in complex dynamic environments, especially deep reinforcement learning. Wu et al. [11] proposed a deep reinforcement learning-based online task offloading algorithm for mobile edge computing networks with variable task arrival intensity. Xu et al. [12] proposed a cooperative task offloading scheme for the UAV-enabled MEC systems based on the successive convex approximation method.

Security has always been a key issue in MEC, and much research has been done in academia and industry on vehicular network security and edge computing Security. Security is a decisive factor for public acceptance of MEC and commercial deployment in VANETs [13]. The issue of compromising users' privacy leads to serious consequences. Gyawali et al. [14] conducted a comprehensive survey of state-of-the-art solutions for security and privacy in Vehicle-Assisted Networks (VANETs) and categorized security threats to VANETs. Miao et al. [15] categorized the security requirements of VANETs and proposed solutions to satisfy the security and privacy issues. Standard solutions to privacy issues in VANETs tend to utilize digital signatures such as digital signatures [16,17], group signatures [18,19], and pseudonymous authentication [20] as the basic solution. Wang et al. [21] solved the task offloading optimization problem and established a security protection model by setting different security levels for each task. Based on the proposed model, tasks are offloaded to various locations to improve data security and meet the computing requirements of other tasks. Liu et al. [22] proposed an innovative blockchain-enabled information-sharing solution in a zero-trust context to guarantee anonymity yet entity authentication in edge computing systems.

In summary, current vehicular edge computing is typically based on traditional centralized trust-based security techniques. Task offloading methods are lacking for handling task offload request methods in untrustworthy vehicular edge computing sharing environments.

3 System Architecture Model

3.1 Architecture Model

The paper refers to the vehicular edge computing architecture for the system architecture model. The vehicular edge computing architecture is depicted as Fig. 1. This architecture comprises three layers: the cloud center layer, the edge cloud layer, and the edge device layer. The top layer, the cloud center layer, is the central hub for processing and managing data. Below it lies the edge cloud layer, which consists of roadside units, base stations, and Access Points (APs) equipped with vehicular network capabilities such as IEEE 802.11p, C-V2X, and 5G. The edge cloud layer provides services to edge devices and functions as a decision center for scheduling resources for surrounding task requests. The third layer, the edge device layer, primarily consists of connected vehicles.

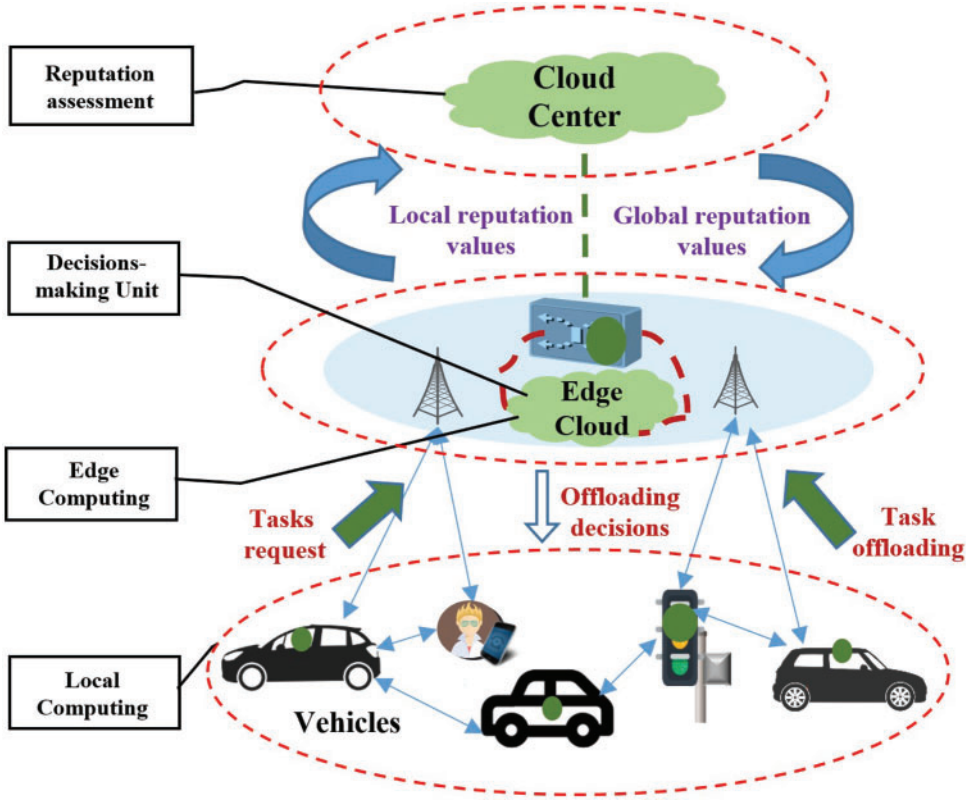

Figure 1: Vehicular edge computing architecture

This paragraph provides an abstract description of the vehicular edge computing architecture. For clarity and ease of presentation, we denote the decision center as **D**, the set of edge clouds as **M**, with individual edge clouds denoted as m_i where $i = 1, 2, 3, \ldots, m$, and the set of connected vehicles as **N**, with individual connected vehicles denoted as n_i where $i = 1, 2, 3, \ldots, n$. The symbol t represents time, where the time slots of neighboring moments are denoted as $\tilde{t}$, and $\tilde{t}$ is set as 1 s in this research. In this section, we utilize the symbol $R_i = \langle f_i, d, d', \psi, \lambda, pos, v, dir \rangle$ to denote the offload request, where f_i denotes the CPU frequency locally used for task R_i. d denotes the input data size of task R_i in bytes. d' denotes the result data size. ψ is used to represent the workload in bytes per CPU cycle. dir denotes the steering direction of the following route, with values ranging from 1 to 4 representing four directions, depending on the passing road and nearby decision

unit. *pos* denotes the position of connected vehicle i at this moment. v denotes the speed of connected vehicle i at this moment. The vehicles need to report task request information to the decision center, and the edge server also needs to report relevant information to the decision center. Specifically, this includes available resources and location, represented as $R_j = \langle f_j, pos \rangle$.

3.2 Problem Formulation

In vehicular edge computing, tasks can be performed in local and vehicle-edge collaborative computing. When a task is performed through local computing, the task delay for task R_i is given by:

$$t_l = \frac{d \cdot \psi}{f_i} \tag{1}$$

If vehicle i offloads its task R_i to edge cloud j, and the allocated computational resources are denoted as $f_{i,j}$, the task R_i delay consists of four parts: the execution time on vehicle i, denoted as $t_i^{e,k}$; the execution time on edge cloud j, denoted as $t_{i,j}^{e,k}$; and the time of data transmission between vehicle i and edge cloud j, denoted as $t_{i,j}^{u,k} + t_{j,i}^{u,k}$. These four delays can be calculated as the following equation:

$$t_{i,j}^{e,k} = \frac{d \cdot (1-\lambda)\psi}{f_i}, t_i^{e,k} = \frac{d \cdot \lambda \cdot \psi}{f_{i,j}} \tag{2}$$

$$t_{i,j}^{u,k} + t_{j,i}^{u,k} = \frac{d}{r} + \frac{d'}{r} \tag{3}$$

where r is the transmission rate. Finally, the computation delay for the vehicle task is the maximum time between the processing time delay of the task on the vehicle and the time spent on edge cloud, expressed as:

$$t_i = max \left\{ t_i^{e,k},\ t_{i,j}^{e,k} + t_{i,j}^{u,k} + t_{j,i}^{u,k} \right\} \tag{4}$$

4 Method

This section presents the proposed task offloading method based on reputation assessment. It describes the proposed method in two parts: reputation management and a deep reinforcement learning task offloading method based on reputation. The deep reinforcement learning task offloading algorithm addresses the resource allocation problem in dynamic vehicular edge computing, and reputation management addresses the trust issues of the sharing computing paradigm.

4.1 Reputation Assessment

To cope with the misbehavior in vehicular edge computing, we design a reputation assessment method for task offloading. The reputation method first assesses the offloading consistency and then calculates the global reputation value used in the task offloading method.

4.1.1 Offloading Consistency Assessment

To assess the offloading consistency, for a given task offloading and resource allocation algorithm, its offloading decision scheduling will be followed by an evaluation of the computational results, including the task's delay, data transmission results, and offloading duration. The paper assumes that the offloading algorithm is accurate enough and the vehicle without misbehavior is consistent between its operation and the expected offloading. We introduce an offloading consistency $s_{i,j}$ corresponding to the reputation value. The consistency assessment compares the execution status after the offloading with the prediction execution

status. We can judge the reputation value by the offloading consistency. As previously analyzed, misbehavior primarily involves misrepresentation of location and task information. The location information misbehavior might affect the task offloading decision specifically regarding task offloading duration. Task information misbehavior might affect task offloading regarding calculation delay, transmission delay, and workload. Therefore, to assess the consistency, the following features are used in Table 1.

Table 1: Features of task offloading

	Features	Description
Decision center	Calculation delay	t^c
	Transmission delay	t^u
	Transmission data size	$d + d'$
	Task offloading duration	$\mathfrak{d}$
	Workload	$d \cdot \psi \cdot \lambda$
Connected vehicle and edge cloud	Calculation delay	t_i^c
	Transmission delay	t_i^u
	Transmission data size	$(d + d')_i$
	Task offloading duration	$\mathfrak{d}_i$
	Workload	ϖ_i

The consistency parameters are assessed as follows:

- Calculation delay offset x_1: It is the relative error between the predicted execution delay and the actual execution delay for each task, i.e., x_1: $\nabla t^e = (|t_i^u - t^u|)/t^u$.
- Transmission delay offset x_2: It is the relative error between the predicted data transmission time and the actual transmission time for each task, i.e., x_2: $\nabla t^u = (|t_i^u - t^u|)/t^u$.
- Data offset x_3: The relative error between the amount of data reported by the connected vehicle and the amount of data actually sent, i.e., x_3: $\nabla(d + d') = (|(d + d')_i - (d + d')|)/(d + d')$.
- Workload offset x_4: The relative error between the workload reported by the networked vehicles and the actual workload, i.e., x_4: $\nabla\varpi = (|\varpi_i - d \cdot \psi \cdot \lambda|)/(d \cdot \psi \cdot \lambda)$.
- Task offloading duration x_5: The relative error between the predicted unloading duration and the actual duration, i.e., x_5: $\nabla\mathfrak{d}_i = (|\mathfrak{d}_i - \mathfrak{d}|)/\mathfrak{d}$.

The final consistency s_{ij} is calculated by averaging all the consistency features.

$$s_{ij} = avg(x_1, x_2, x_3, x_4, x_5) \tag{5}$$

Defining the local reputation value in time t as:

$$c_{ij} = \begin{cases} s_{ij}, & s_{ij} \leq \tau \\ 1, & s'_{ij} > \tau \end{cases} \tag{6}$$

Here, $\tau \in (0,1)$ represents the reputation threshold. A trust threshold is introduced in the offloading consistency calculation to mitigate feature bias resulting from systematic errors, such as suboptimal offloading decisions and inaccurate positioning systems. It is set to 1 when it exceeds τ.

4.1.2 Calculation of Reputation Value

This paper introduces reputation value to deter misbehavior and prevent unintended misbehavior from hindering access to edge cloud resources. Here, $\rho_t \in [0,1]$ represents this paper's global reputation value at time t. A higher value of ρ_t indicates a higher reputation. The study presents the calculation method for the global reputation value, known as the EigenTrust-based method.

In the vehicular edge computing scenario, both the edge cloud and the decision center provide offloading information to a specific cloud center. Based on the offloading consistency, this center calculates a local reputation value c_{ij}, as shown in Eq. (7).

$$\mathbf{c} = \begin{bmatrix} c_{11} & \cdots & \cdots & c_{1n} \\ \cdots & c_{22} & \cdots & \cdots \\ \cdots & \cdots & c_{33} & \cdots \\ c_{m1} & \cdots & \cdots & c_{mn} \end{bmatrix} \tag{7}$$

When the decision center needs to resolve task offloading decisions, the cloud center must provide a global reputation value for vehicle i. In line with the EigenTrust-based peer-to-peer reputation system, a plausible approach involves soliciting opinions from acquaintances of a given node to compute its global reputation. However, the local reputation matrix (7) is an asymmetry in vehicular edge computing. Consequently, calculating global reputation must establish the trust relationship between a specific edge cloud i and the relevant vehicles. Here, similarity is employed to signify the trust relationship between i and j, as follows:

$$cos\left(c_i, c_j\right) = \frac{\boldsymbol{c_i} \cdot \boldsymbol{c_j}}{\|\boldsymbol{c_i}\| \cdot \|\boldsymbol{c_j}\|} \tag{8}$$

Therefore, the global reputation values can be calculated as follows:

$$\rho'_{i,k} = \sum_j cos\left(c_i, c_j\right) \cdot c_{j,k} \tag{9}$$

By introducing the reputation value thresholds, the reputation value is represented as:

$$\rho_{i,k} = \begin{cases} \rho'_{i,k}, & \rho'_{i,k} \leq \tau \\ 1, & \rho'_{i,k} > \tau \end{cases} \tag{10}$$

4.2 Deep Reinforcement Learning Task Offloading Method Based on Reputation

To illustrate the proposed deep reinforcement learning-based task offloading algorithm, we introduce the state space, action space, and reward settings. The task offloading algorithm proposed in this study is based on Double Deep Q-Network (DQN).

4.2.1 State Space Setting

The decision-making unit acquires the environmental states and addresses the optimization objective. The state space includes information about vehicles $R_i = \langle f_i, d, d', \psi, \lambda, pos, v, dir \rangle$ and $R_j = \langle f_j, pos \rangle$. The dimensions of the state space vector depend on the number of vehicles and edge clouds. The state space dimensions are set and the values of the remaining dimensions are filled with zeros when the number of vehicles and edge clouds is low.

4.2.2 Action Space Setting

The joint-optimized task offloading algorithm discretizes the action space to suit deep reinforcement learning. In this paper, we define the offloading policy for vehicle i as $\mathfrak{c} = \langle i, j, \mathfrak{p}, \mathfrak{d} \rangle$, where $\mathfrak{p} \in [0,1]$ represents the allocation of $\mathfrak{p}$ of the available resources of edge cloud j to vehicle i, discretized in this paper into discrete values, e.g., $\mathfrak{p} \in \{0.2, 0.8\}$, and $\mathfrak{d}$ can be discretized into specific time lengths, such as $\mathfrak{d} \in \{4, 10\}$. If $\| \mathbb{M} \| = 2$, then there can be $2 * 2 * 2 + 1 = 9$ ways of offloading tasks. When "$j = 1$", the task is not offloaded and is computed locally by the vehicle.

4.2.3 Reward Setting

Properly setting rewards in reinforcement learning enables deep learning networks to learn policies and converge faster. The paper aims to address the delay problem. Therefore, the reward setting is closely linked to the delay. This paper defines a fixed optimization window $\mathbb{T} = 1, 2, 3, \ldots, t$. Let t_l represent the delay for vehicle local computation and t_{le} denote the delay for vehicle-edge cloud cooperative computation. The ratio $r_{i,t} = \frac{t_l}{t_{le}}$ is a crucial indicator in the proposed Double DQN-based task offloading algorithm. When the connected vehicle opts for local computation, the value is 1. In situations where the connected vehicle i allocates resources from edge cloud j, but the task remains entirely local during the t time period, the indicator is set as $r_\tau \in (0,1)$, with subsequent experiences setting it to 0.8.

$$r_{i,t} = \begin{cases} 1, & \text{local comuputing} \\ \frac{t_l}{t_{le}}, & \text{edge computing} \\ r_\tau, & \text{local completing with assigned resource} \end{cases} \tag{11}$$

The average value of $r_{i,t}$ over optimization windows is the reward value of the proposed Double DQN-based task offloading algorithm. The reward for the offloading decision $\mathfrak{c} = \langle i, j, \mathfrak{p}, \mathfrak{d} \rangle$ is rewarded as:

$$r_i = \sum_{t=1}^{\mathfrak{d}+1} r_{i,t} \tag{12}$$

The following section describes the specific steps of the proposed algorithm, including training and decision process. The algorithm trains two networks: a deep Q-network Q. During training, the deep Q-network Q estimates the Q-value of the action. The total number of training steps for the deep Q-network Q is denoted as T_q.

1) Processing of state inputs. In this paragraph, we label the input states S as $S = s_1, s_2, \ldots, s_s$, as previously defined. Practical training of the Q-network requires the normalization of information across all dimensions. Given that information in different dimensions may not convey the same meaning, each dimension must be processed based on its respective value range. Normalization is employed for this purpose.

$$S' = \frac{S - S_{min}}{S_{max} - S_{min}} \tag{13}$$

2) Definiting Q-network. The Q-network computes the values of state-action pairs to determine the action with the highest Q-value. We define the Q-network using a Multilayer Perceptron (MLP). ReLU activation functions are applied to the neurons in each layer of the Q-network to enhance the nonlinear representation of the network.

3) Selecting actions. The deep Q-network outputs values corresponding to each action, and the optimal action is chosen using softmax. The policy is selected using the $\varepsilon-$greedy algorithm. The DDQN algorithm computes the Q-values of all actions under the state S_t using the estimation Q-network. Then, the policy selects the optimal action with probability ε or randomly selects an action from the action space.

$$A_t = \begin{cases} \arg\max_{a_t} Q\left(S_t, a, w_t\right), & \text{if } p <= \varepsilon \\ a \text{ (chosen randomly)}, & \text{if } p > \varepsilon \end{cases} \tag{14}$$

After getting action A_t, the algorithm can solve the task offloading decision $\mathfrak{c} = \langle i, j, \mathfrak{p}, \mathfrak{d} \rangle$. In addition, the decision-making unit gets a reputation $\rho_{i,t}$ from the cloud center, and the final decision will be $\mathfrak{c} = \langle i, j, \rho_{i,t} \cdot \mathfrak{p}, \mathfrak{d} \rangle$

4) Training Q-network. After the vehicular edge computing conducts the selected action, m samples $\{S_j, A_j, R_j, S'_j, is\_end_j\}$ are sampled from the replay buffer. By the Q network, the trainer calculates y_i is:

$$y_i = R_j + \gamma Q'\left(S'_j, \arg\max_{a'} Q\left(S'_j, a, w\right), w'\right) \tag{15}$$

After calculating the action state values, the mean squared loss function $1/m \sum_{j=1}^{m} \left(y_j - Q\left(S_j, A_j, w\right)\right)^2$ is used to update all the parameters w of deep Q-network by back propagation of loss. Then, the target network is updated by $w' = w$. The training process of the Double DQN-based task offloading algorithm for VEC is described as follows Algorithm 1.

Algorithm 1: Double DQN-based task offloading algorithm for VEC

Input: Edge clouds $M = \{1, 2, \ldots, m\}$, vehicular tasks $N = \{1, 2, \ldots, n\}$
Output: evaluation network Q network w

done = True
Initialize the evaluation network Q and target network Q' as w and w'
for episode = 1 to M **do**
  obs = env.reset()
  done = False
  while done = False **do**
    a_t = argmax $Q(a_i)$
    $a'_t = \langle i, j, \rho_{i,k} \cdot \mathfrak{p}, \mathfrak{d} \rangle$,
    $S, reward, done = env.step(a'_t)$
    $D.push\_back(S_t, a'_t, r_t, S_{t+1})$
    Sample m samples from the replay buffer $D : D_m = \{S_j, a_j, r_j, S_{j+1}\}$
    $y_j = R_j + \gamma Q'\left(S'_j, \arg\max_{a'} Q\left(S'_j, a, w\right), w'\right)$
    Use the mean squared loss function $L = (y_j - Q(S_j, a_j, w))^2$ to update all parameters w of the current evaluation Q network.
    $w \leftarrow w - \mu \frac{\partial L}{\partial w}$
    if $episode\%C == 0$ **then**
      $w' = w$ // update target network according to the frequency C.
    end if
    episode++
  end while
end for

5 Evaluation

Simulation experiments were designed to evaluate the proposed reputation-based task offloading method for vehicular edge computing. The evaluations verify the feasibility of the proposed reputation value-based task offloading. First, the simulation environment and parameters used in this section are described in detail. Subsequently, the experimental results are presented and analyzed.

5.1 Experimental Setting

The impact of misbehavior in vehicular edge computing is verified through experiments conducted using the Veins [23] simulation framework, an open-source platform for telematics network simulation. Veins primarily implements vehicular networks using OMNeT++ [24] and simulates road trajectories using SUMO [25]. The experimental environments were executed on a Debian 11 system with an Intel(R) Core(TM) i7-6700 CPU @ 3.40 GHz.

We utilize the open-source map OpenStreetMap [26] as the scenario source. The area surrounding Minzhuang Road in Haidian District, Beijing, is chosen as the road scenario. The area's topographic map and Veins road network map are depicted in Figs. 2 and 3, respectively. For location misbehavior, this paper utilizes the VeReMi dataset. In this evaluation, a constant attack scenario is set up. As depicted in Fig. 3, the designed scenario includes one decision center and two edge clouds. There are six vehicles and three directions: ′2′, ′3′, and ′4′. The location ′1′ in Fig. 3 represents the initial location of all connected vehicles. The roadblock model and network model adhere to Veins' default parameters. Regarding the wireless transmission model parameters, the operating frequency is 5.890 GHz, and the bandwidth is 40 MHz. The maximum vehicle speed is 12 m/s. The vehicle CPU resource is 1.2 GHz, the data size d is 40,000 Byte, the data size d' is 800 Byte, and the evaluation assumes that the vehicle and edge cloud handle the task at a rate of 500 Byte/Hz. The CPU resource is 7.0 GHz.

Figure 2: Simulated environment

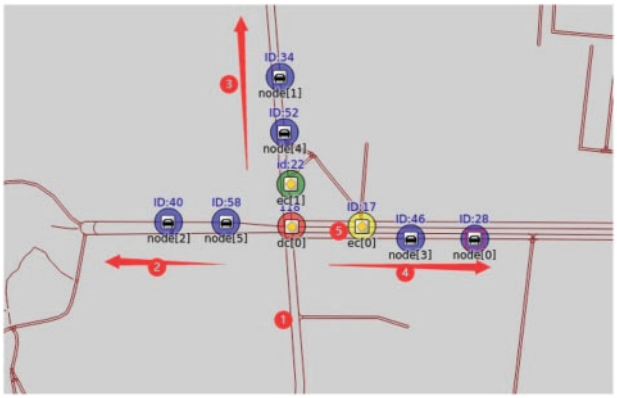

Figure 3: Veins road network diagram

5.2 Evaluation of Deep Reinforcement Learning-Based Task Offloading Algorithms for Vehicular Edge Computing

The first part of the evaluation aims to validate deep reinforcement learning-based task offloading algorithms for vehicular edge computing. The experiment evaluates four offloading methods: the proposed method, random policy selection algorithm, vehicle local computing, and a task offloading decision-making method based on the greedy algorithm. The correlation between reward and delay was depicted in the preceding section. As Fig. 4 shows, local computing by vehicle causes almost minimal rewards, which means there is no optimization for latency. The random policy selection algorithm would not optimize the task delay. The greedy algorithm can effectively optimize the task delay but is still inferior to the proposed algorithm. Therefore, the experimental results depicted in Fig. 4 demonstrate the effectiveness of the proposed method in expediting vehicle-side collaboration. It is evident that the offloading decision progressively reduces task latency with training.

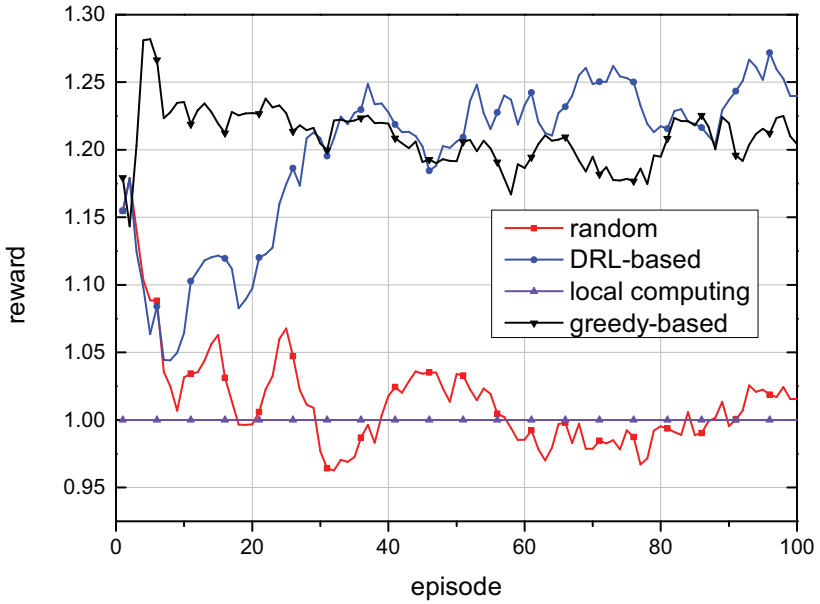

Figure 4: Comparison diagram of task offloading algorithms based on reinforcement learning

The efficacy of different numbers of requesting vehicles is validated by performing task requests with varying total numbers of task-request vehicles. The experiment configures the number of vehicles as 1, 3, and 6. With one vehicle, a networked vehicle heads toward direction ′4′ as Fig. 3 shows. When there are three vehicles, the vehicles head towards the ′2′, ′3′, ′4′ directions, respectively. With six vehicles, each two connected vehicles heads toward the ′2′, ′3′, ′4′ directions, respectively. The Fig. 5 demonstrates that with fewer vehicles, there is reduced competition for resources, allowing vehicles to obtain computational resources efficiently. However, with an increased number of vehicles, the proposed method effectively optimizes average delay.

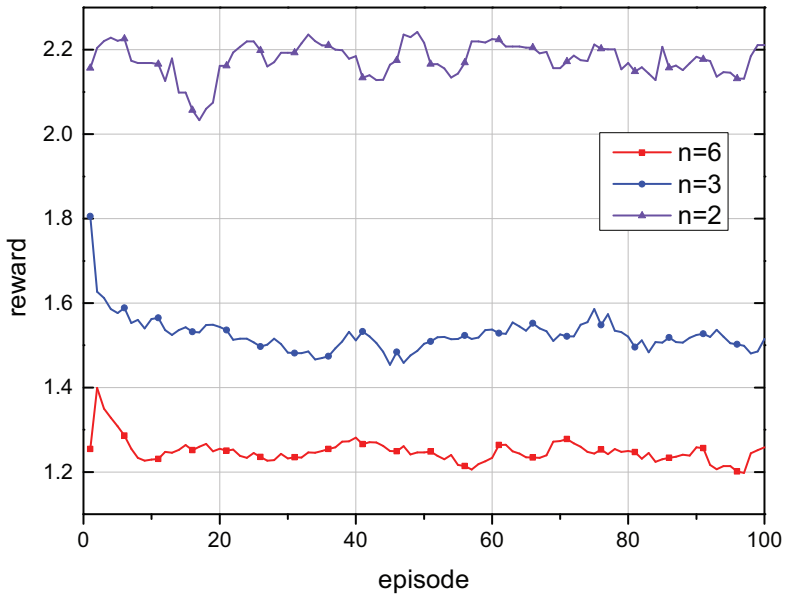

Figure 5: The task offload performance with the differential number of vehicles

5.3 Evaluation of Reputation Assessment-Based Task Offloading Algorithms for Vehicular Edge Computing

The second part of the evaluation aims to verify reputation-based task offloading algorithms for vehicular edge computing. As depicted in Fig. 3, the two vehicles on Minzhuang Road consistently misreport their positions to the decision center as position ′5′ in Fig. 3. This behavior constitutes a standard constant attack, and the attackers request task offloading with the fake positions. The misbehavior by the vehicles, as shown in Figs. 6 and 7, increases overall task latency, thereby reducing overall performance. Setting the reputation value enables the offloading decision to inhibit such misbehavior from adversely affecting the task offloading decision. For instance, setting the reputation value to 0.6 in this experiment results in improved task latency performance. Fig. 8 presents the resource allocation results, demonstrating that the reputation value can limit the capabilities of vehicles engaging in misbehavior.

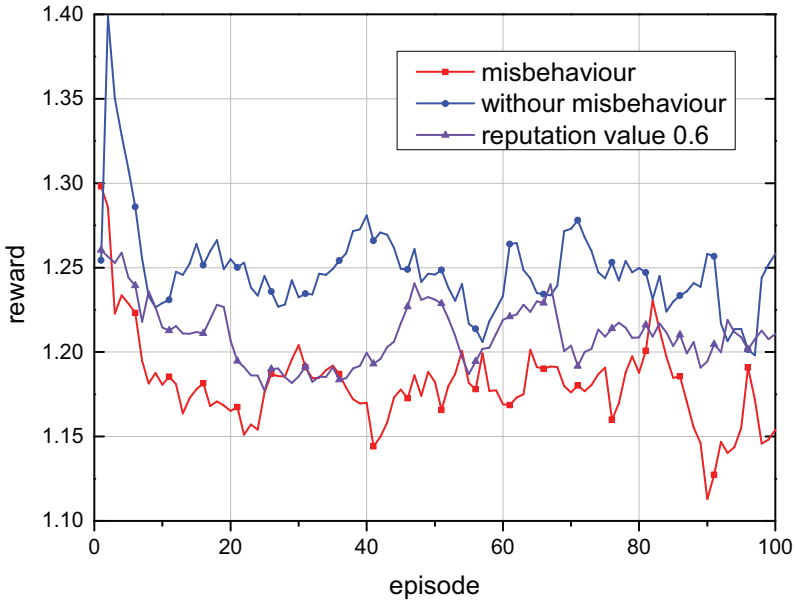

Figure 6: The performance of the proposed DQN-based task offload algorithm

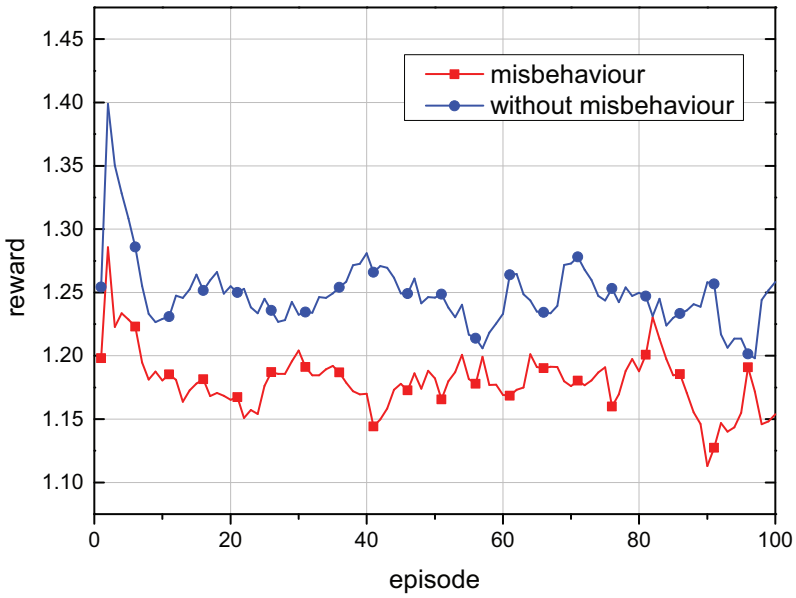

Figure 7: Task offload performance caused by misbehavior

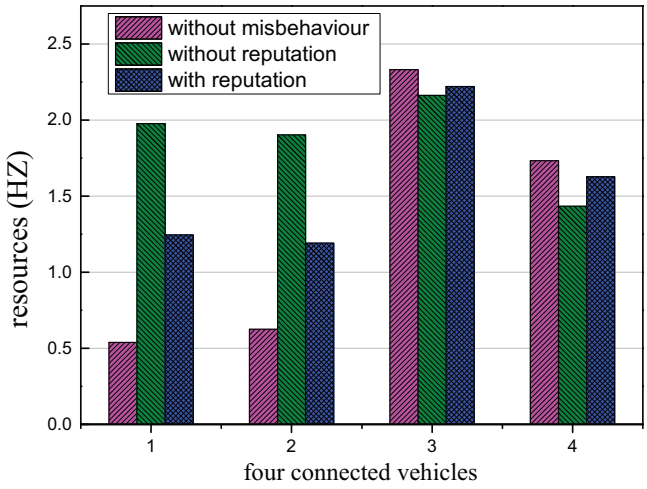

Figure 8: Resource allocation in three modes

6 Conclusion

This paper introduces a task offloading algorithm based on reputation assessment to address the challenge of task offloading in vehicular edge computing systems against potential misbehavior. Firstly, a deep reinforcement learning-based algorithm is proposed for vehicular edge computing task offloading. Secondly, the reputation assessment method for connected vehicles is proposed using task offloading information in vehicular edge computing scenarios. Additionally, a globally reliable reputation value is calculated using the EigenTrust algorithm. Lastly, a reputation-value based task offloading method is introduced to mitigate misbehavior effectively. To validate the proposed approach, simulation experiments based on Veins are conducted. The experimental analyses demonstrate that the proposed reputation-based task offloading algorithm for vehicular edge computing effectively curbs undesirable behaviors and enhances the efficiency of task offloading decisions.

7 Future Work

Investigating task offloading methods that incorporate distributed reputation management in the future will be interesting. In the vehicular edge computing architecture, vehicles form a distributed network. Designing a distributed reputation computation method that allows vehicles to access resources in the vehicular edge computing environment quickly is crucial. Therefore, distributed reputation management is essential to ensure the credibility of reputation. On the other hand, it is also necessary to update the reputation value of vehicles based on their behavior. We will focus on developing task offloading methods incorporating distributed reputation management for vehicular edge computing.

Acknowledgement: None.

Funding Statement: This paper is supported by the Open Foundation of Henan Key Laboratory of Cyberspace Situation Awareness (No. HNTS2022020), the Science and Technology Research Program of Henan Province of China (232102210134, 182102210130), and Key Research Projects of Henan Provincial Universities (25B520005).

Author Contributions: The authors confirmed their contributions to the papers: study conception and design: Jun Li; evaluation: Jun Li; draft manuscript preparation: Yawei Dong, Liang Ni, Guopeng Feng, and Fangfang Shan. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: Not applicable.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest to report regarding the present study.

References

1. Subramaniam EVD, Krishnasamy V. Hybrid optimal ensemble SVM forest classifier for task offloading in mobile cloud computing. Comput J. 2024;67(4):1286–97. doi:10.1093/comjnl/bxad059.
2. Chu W, Jia X, Yu Z, Lui JCS, Lin Y. On incentivizing resource allocation and task offloading for cooperative edge computing. Comput Netw. 2024;246(3):110428. doi:10.1016/j.comnet.2024.110428.
3. Saeik F, Avgeris M, Spatharakis D, Santi N, Papavassiliou S. Task offloading in edge and cloud computing: a survey on mathematical, artificial intelligence and control theory solutions. Comput Netw. 2021;195(3):108177. doi:10.1016/j.comnet.2021.108177.
4. Feng J, Liu Z, Wu C, Ji Y. AVE: autonomous vehicular edge computing framework with ACO-based scheduling. IEEE Trans Veh Technol. 2017;66(12):10660–75. doi:10.1109/TVT.2017.2714704.
5. Liu Y, Li Y, Niu Y, Jin D. Joint optimization of path planning and resource allocation in mobile edge computing. IEEE Trans Mob Comput. 2020;19(9):2129–44.
6. Zhang K, Mao Y, Leng S, Maharjan S, Zhang Y. Optimal delay constrained offloading for vehicular edge computing networks. In: 2017 IEEE International Conference on Communications (ICC); Paris, France: IEEE; 2017. p. 1–6.
7. Bonab MJA, Kandovan RS. Effective resource allocation and load balancing in hierarchical hetnets: toward QoS-Aware multi-access edge computing. Comput J. 2023;66(1):229–44. doi:10.1093/comjnl/bxab157.
8. Huang CM, Chiang MS, Dao DT, Su WL, Xu S, Zhou H. V2V data offloading for cellular network based on the software defined network (SDN) inside mobile edge computing (MEC) architecture. IEEE Access. 2018;6:17741–55. doi:10.1109/ACCESS.2018.2820679.
9. Wang X, Ning Z, Guo S. Multi-agent imitation learning for pervasive edge computing: a decentralized computation offloading algorithm. IEEE Trans Parallel Distrib Syst. 2020;32(2):411–425.
10. Li B, Liu W, Xie W, Li X. Energy-efficient task offloading and trajectory planning in UAV-enabled mobile edge computing networks. Comput Netw. 2023;234(1):109940. doi:10.1016/j.comnet.2023.109940.
11. Wu H, Geng J, Bai X, Jin S. Deep reinforcement learning-based online task offloading in mobile edge computing networks. Inf Sci. 2024;654(3):119849. doi:10.1016/j.ins.2023.119849.
12. Xu D, Xu D. Cooperative task offloading and resource allocation for UAV-enabled mobile edge computing systems. Comput Netw. 2023;223(3):109574. doi:10.1016/j.comnet.2023.109574.
13. Goudarzi S, Abdullah AH, Mandala S, Soleymani SA, Baee MAR, Anisi MH et al. A systematic review of security in vehicular ad hoc network. In: Proceedings of the Second European Symposium WSCN; Jeddah, Saudi Arabia; 2013. p. 1–10.
14. Gyawali S, Qian Y, Hu RQ. Machine Learning and reputation based misbehavior detection in vehicular communication networks. IEEE Trans Vehicular Technol. 2020;69(8):8871–85.
15. Miao D, Liu L, Xu R, Panneerselvam J, Wu Y, Xu W. An efficient indexing model for the fog layer of industrial internet of things. IEEE Trans Indust Inform. 2018;14(10):4487–96. doi:10.1109/TII.2018.2799598.
16. Zhang S, Tao J, Yuan Y. Anonymous authentication-oriented vehicular privacy protection technology research in vanet. In: 2011 International Conference on Electrical and Control Engineering; Yichang, China: IEEE; 2011. p. 4365–8.
17. Wu Q, Domingo-Ferrer J, González-Nicolás U. Balanced trustworthiness, safety, and privacy in vehicle-to-vehicle communications. IEEE Trans Veh Technol. 2009;59(2):559–73.
18. Studer A, Shi E, Bai F, Perrig A. TACKing together efficient authentication, revocation, and privacy in VANETs. In: 2009 6th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks; Rome, Italy: IEEE; 2009. p. 1–9.

19. Sampigethaya K, Huangy L, Li M, Poovendran R, Matsuuray K, Sezaki K. CARAVAN: providing location privacy for VANET. In: Proceedings of Embedded Security in Cars (ESCAR); 2005.
20. Sun Y, Lu R, Lin X, Shen X, Su J. An efficient pseudonymous authentication scheme with strong privacy preservation for vehicular communications. IEEE Trans Veh Technol. 2010;59(7):3589–603. doi:10.1109/TVT.2010.2051468.
21. Wang Z, Ding Y, Jin X, Chen Y, Gao C. Task offloading for edge computing in industrial Internet with joint data compression and security protection. J Supercomput. 2023;79(4):4291–317. doi:10.1007/s11227-022-04821-9.
22. Liu Y, Hao X, Ren W, Xiong R, Zhu T, Choo KKR, et al. A blockchain-based decentralized, fair and authenticated information sharing scheme in zero trust internet-of-things. IEEE Trans Comput. 2023;72(2):501–12. doi:10.1109/TC.2022.3157996.
23. Veins. [cited 2025 Feb 11]. Available from: https://veins.car2x.org/.
24. OMNeT++. [cited 2025 Feb 11]. Available from: https://omnetpp.org/.
25. SUMO. [cited 2025 Feb 11]. Available from: https://eclipse.dev/sumo/.
26. Li J. [cited 2025 Feb 11]. Available from: https://github.com/usuallyoralways/vectaskoffload.