Dynamics-Based Location Prediction and Neural Network Fine-Tuning for Task Offloading in Vehicular Networks

Yuanguang Wu, Lusheng Wang*, Caihong Kai, and Min Peng
The Key Laboratory of Knowledge Engineering with Big Data, Ministry of Education,
The School of Computer Science and Information Engineering,
Hefei University of Technology, Hefei 230601, China
[e-mail: wanglusheng@hfut.edu.cn]
*Corresponding author: Lusheng Wang

Received August 29, 2023; revised October 18, 2023; accepted December 4, 2023; published December 31, 2023

Abstract

Task offloading in vehicular networks is hot topic in the development of autonomous driving. In these scenarios, due to the role of vehicles and pedestrians, task characteristics are changing constantly. The classical deep learning algorithm always uses a pre-trained neural network to optimize task offloading, which leads to system performance degradation. Therefore, this paper proposes a neural network fine-tuning task offloading algorithm, combining with location prediction for pedestrians and vehicles by the Payne model of fluid dynamics and the car-following model, respectively. After the locations are predicted, characteristics of tasks can be obtained and the neural network will be fine-tuned. Finally, the proposed algorithm continuously predicts task characteristics and fine-tunes a neural network to maintain high system performance and meet low delay requirements. From the simulation results, compared with other algorithms, the proposed algorithm still guarantees a lower task offloading delay, especially when congestion occurs.

Keywords: Internet of Vehicles, task offloading, neural network, fine-tuning, parameter optimization.

This research was supported by Anhui Provincial Natural Science Foundation under grant no. 2308085MF194.

http://doi.org/10.3837/tiis.2023.12.011 **ISSN : 1976-7277**

1. Introduction

Autonomous vehicles are increasingly integrated into social life, greatly changing people's production and lifestyle. An intelligent vehicle can make intelligent decisions based on external environmental conditions and then trigger physical actions [1]. These functions are accompanied by a large number of tasks which are data-intensive, computing-intensive, and delay-sensitive. Although the intelligent vehicle itself has computing resources, it cannot perform large-scale calculations [2]. In order to overcome this limitation of intelligent vehicles, the Internet of Vehicles system extends the concept of cloud computing [3]. In this model, the cloud provides computing resources and in order to scalable extensive data processing the cloud utilizes the resources of local vehicles and remote data centers but the connection between the Internet of Vehicles system and the cloud data center usually requires multiple hops, resulting in long communication time and data transmission delay. Consequently, this model becomes less suitable for delay-sensitive operations. In autonomous driving, intelligent vehicles need to make decisions based on the environment at the millisecond level. With cloudlet and fog nodes, edge computing can bring infrastructure, software services, and platforms closer to data sources [4], thus playing an important role in solving this limitation. In edge computing, an intelligent vehicle first offloads tasks to edge servers for processing, and then returns the results to the vehicle itself. Edge computing can provide high mobility, high security, low delay services, and provide computing resources beyond the vehicle itself [5]. In mobile edge computing (MEC), computing offloading is a highly important technology. Computing offloading decisions and resource allocation are the two most important parts [6].

Task offloading is expressed as a NP-hard problem. Previous studies usually use traditional algorithms. Combining the multi-armed bandit theory, Sun *et al.* in [7] designed an algorithm based on adaptive learning which considered task offloading among vehicles. Dass *et al.* in [8] designed a graph optimization problem and a two-fold efficient heuristic approach. In a multi-user MEC scenario, Li *et al.* in [9] combined dynamic niche and proposed an improved algorithm based on self-organizing learning. For the task offloading, Liu *et al.* designed a matching algorithm based on pricing in [10]. Misra *et al.* in [11] proposed a greedy heuristic approach by jointly considering some vital metrics. Using a direct acyclic graph to represent tasks and their dependencies, Hou *et al.* in [12] presented an algorithm to find optimal offloading orderings. Yuan *et al.* considered the offloading of interdependent tasks and proposed a convex-based algorithm in [13]. Considering that a task can be divided, Li *et al.* proposed an interior point method-based algorithm in [14]. Considering the cloud-MEC collaborative scenario, Zhao et al. designed a distributed algorithm based on the game-theoretic and Lagrange multiplier method in [15]. Considering the dependence of tasks, Liu et al. in [16] proposed a heuristic ranking-based algorithm. For the collaborative computation among the vehicles, the unmanned aerial vehicle, and the terrestrial computing servers, Zhao *et al.* in [17] proposed an online distributed algorithm based on Lyapunov optimization.

Although the task offloading decision using the traditional algorithm described above can be close to the optimal solution, it requires enough time to reach convergence and is difficult to apply to the Internet of Vehicles scenario with high real-time requirements. As a result, deep learning methods are widely used. Considering some vital conditions, Liu *et al.* proposed an algorithm based on deep reinforcement learning in [18]. Considering the changes in network conditions, Yang *et al.* in [19] designed an algorithm based on multi-task learning. Due to the service availability and mobility of vehicles and the priority of tasks, Shi *et al.* in [20] developed a soft actor-critic based algorithm. Taking into account the dynamics of vehicles and frequent changes in decision-making, Guo *et al.* in [21] designed an intelligent task

offloading deep Q-learning based algorithm. Considering the heterogeneous environment, Sun et al. in [22] proposed a deep deterministic policy gradient-based multiple continuous variable decision model. To reduce the impact of congestion on the QoS in the Internet of Vehicles scenarios, the load balance in the MEC system is necessary to consider. Taking into account vehicles' mobility and delay requirements, Hsu *et al.* in [23] proposed an algorithm that combined particle swarm optimization and neural network. Considering the intrinsic task dependency, Wang *et al.* in [24] proposed an off-policy reinforcement learning algorithm. In our previous studies in [25], a location prediction-based algorithm was proposed, which saved the time consumption of signaling and optimization. However, that study did not take into account that task characteristics were constantly changing in the Internet of Vehicles scenario. A pre-trained deep neural network (DNN) was always used to optimize task offloading, making that algorithm only adaptable to a relatively stable scenario.

When pedestrians and vehicles move, the difference in characteristics of tasks could be relatively large, so a fixed neural network is difficult to always get the most suitable offloading decision, leading to a degradation of the system performance. Therefore, we propose a neural network fine-tuning task offloading algorithm, combining with location prediction for pedestrians and vehicles by the Payne model of fluid dynamics and the car-following model, respectively. The proposed algorithm can complete the prediction procedure before the vehicles move to a certain location in front. After the locations are predicted, characteristics of tasks can be obtained and the neural network will be fine-tuned. When new task characteristics are generated, the fine-tuned neural network can optimize task offloading in real-time and quickly obtain decisions. Finally, by continuously predicting and fine-tuning, it ensures that the offloading decisions made by the neural network become increasingly better. This paper has the following main contributions:

·A dynamics-based location prediction algorithm is proposed for the pedestrian-vehicle intersection scenario. The location prediction algorithm uses the Payne model of fluid dynamics and the car-following model to predict locations for pedestrians and vehicles, respectively.

·A neural network fine-tuning task offloading algorithm is proposed to ensure a neural network can adapt to the scenario with large differences in task characteristics. Because the algorithm combines location prediction, it can sense the change of task characteristics in the first time and fine-tune a neural network in time. The fine-tuned neural network makes a better task offloading decision to meet the low delay requirements.

The remaining sections are organized as follows. In Section 2, the system model is presented. In Section 3, we introduce dynamics-based location prediction. In Section 4, the proposed algorithm is described. In Section 5, we perform simulations and show the results and performance of this algorithm. We give the conclusion in Section 6.

2. System Model

The scenario we study starts from the normal driving of vehicles, then vehicles gradually generate congestion, and finally the congestion dissipates. Specifically, at the crossroads, pedestrians walk along the road, and vehicles slow down and wait for all pedestrians to pass, resulting in road congestion. Until all pedestrians pass completely, vehicles accelerate and continue to move forward. In the scenario, the number and type of tasks generated by vehicles are related to the number of pedestrians around, the size of each task is related to the number of pedestrians within a certain range of the vehicle, and the location distribution of tasks is related to the locations of vehicles. Because locations of vehicles and pedestrians are

constantly changing, task characteristics are continuously changing over time.

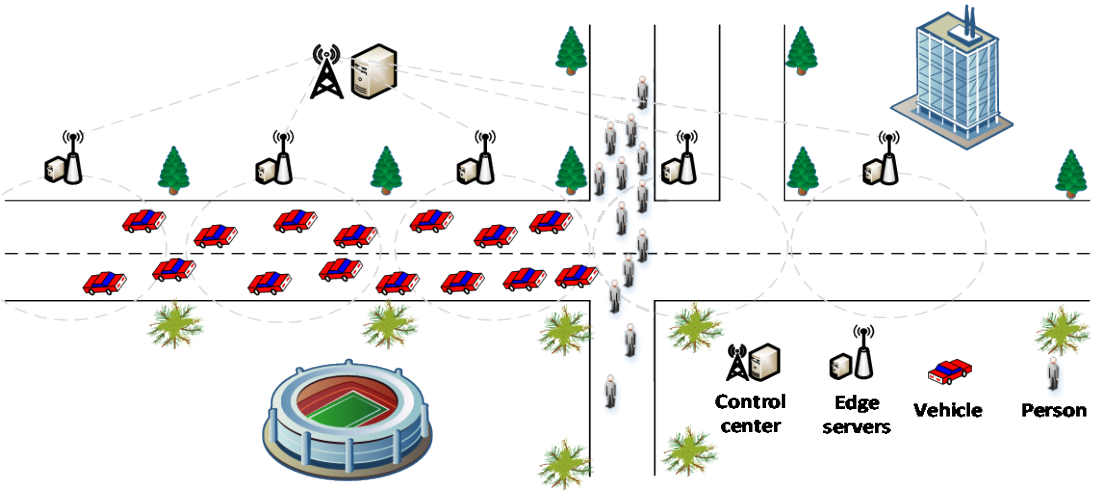


Fig. 1. Pedestrian-vehicle interaction scenario.

As shown in **Fig. 1**, there are K pedestrians, N vehicles and J RSUs with MEC servers in the scenario. These RSUs are marked as $\mathbf{U}=\{U_j \mid j=1,2,...,J\}$ and placed equidistantly on the side of the road. $D_{n,i}=\{C_{n,i},Z_{n,i},T_{n,i}^{\max}\}, i=1,2,...,I_n$ indicates vehicle n generates a total of I_n tasks, where $C_{n,i}$, $Z_{n,i}$, $T_{n,i}^{\max}$ represent the task size, the complexity, and the maximum tolerable delay, respectively.

The sum of the offloading delays of task $D_{n,i}$ to U_j is given by

$$t_{n,i,j}^{total}=t_{n,i,j}^{comp}+t_{n,i,j}^{trans}+t_{n,i,j}^{compwait}+t_{n,i,j}^{transwait}, \tag{1}$$

To the right of (1), the first item is the computation delay, which is related to the task complexity. The second term is the transmission delay, which is related to the task size and the distance from the selected RSU to the task. The last two terms are the computation and transmission waiting delay, which are related to the task arrival rate λ_j. λ_j of the jth RSU is determined by the number of tasks within the RSU coverage radius R.

Aiming at minimizing the total offloading delay of all tasks in the system, task offloading can be modeled as an optimization problem, given by

$$\min\left(\sum_{n=1}^{N}\sum_{i=1}^{I_n}t_{n,i,j}^{total}\right), \tag{2a}$$

$$s1.\sum_{j=1}^{J}x_{n,i,j}=1,\forall n,i, \tag{2b}$$

$$x_{n,i,j}\in\{0,1\},\forall n,i,j, \tag{2c}$$

$$t_{n,i,j}^{total}<T_{n,i,j}^{\max}. \tag{2d}$$

where the task $D_{n,i}$ offloads to U_j, indicated by $x_{n,i,j}=1$. The first and second constraints ensure that only one RSU can be selected for a task to offload. The third constraint indicates that the tolerable delay should be greater than the offloading delay of each task.

3. Location Prediction

Aiming at the shortcoming of using classical deep learning algorithms for task offloading, we propose a neural network fine-tuning task offloading algorithm, combining with location prediction for pedestrians and vehicles by the Payne model of fluid dynamics and the car-following model, respectively. After the locations are predicted, characteristics of tasks can be obtained and the neural network will be fine-tuned. Then, the fine-tuned neural network is used to generate task offloading decisions with better performance. Finally, by continuously predicting task characteristics and fine-tuning, the neural network is ensured to adapt to the scenario with large differences in task characteristics. Task characteristics are closely related to the locations of vehicles and pedestrians. Next, we describe the prediction methods for the locations of pedestrians and vehicles.

Based on the fluid dynamics model, considering the characteristics of pedestrian flow, this flow may be approximated as fluid motion, and the variations of its average density and average velocity with time and space are described by means of fluid dynamics. The pedestrian flow is predicted by the Lighthill-Whitham-Richards (LWR) model, which is a fluid dynamics model of one-dimensional pedestrian [26] [27].

The LWR model applies the continuity equation in fluid dynamics to the pedestrian flow, compares the pedestrian flow to a continuous medium, and establishes a continuous equation (3a) of pedestrian conservation, which ensures the conservation of the number of pedestrians on the road, and assumes that the pedestrian flow on the road is always in a state of motion balance. The model can simulate the nonlinear characteristics of pedestrian shock wave formation and congestion grooming and can be used to describe the situation when there is no ramp in the road to make the road pedestrian flow in and out. However, since LWR assumes a balanced relationship between velocity and density, it cannot accurately describe the unbalanced pedestrian flow process.

Payne in [28] proposed a new dynamic equation (3b) to solve this problem. The Payne model is given by

$$\frac{\partial k(x,t)}{\partial t}+\frac{\partial[k(x,t)\cdot u(x,t)]}{\partial x}=0, \tag{3a}$$

$$\frac{\partial u(x,t)}{\partial t}+u(x,t)\frac{\partial u(x,t)}{\partial x}=-\frac{1}{\tau}(u(x,t)-u_e(k(x,t)))-\frac{v}{k(x,t)\tau}\frac{\partial k(x,t)}{\partial x}, \tag{3b}$$

where $k(x,t)$ represents the pedestrian density, $u(x,t)$ represents the average velocity of pedestrians and $u_e(k(x,t))$ is the velocity-density balanced function. τ is the relaxation time. $v=-0.5\frac{\partial u_e(k(x,t))}{\partial k(x,t)}>0$ is the expectation index. To the right of (3b), the first item represents the acceleration process in which the pedestrian adjusts its velocity to achieve a balanced velocity-density relationship. The second item is the expected item, which is the process of the pedestrian's reaction to the pedestrian state in front of him. For example, if the density in front becomes higher due to congestion, the pedestrian has to reduce the velocity, and vice versa. Because the average velocity of the pedestrian flow cannot instantaneously follow the density change. However, according to the density in front, the pedestrian will adjust the velocity, the Payne model can be used to predict pedestrian flow.

If the above model is used to predict pedestrians, the selection of the balanced function $u_e\left(k(x,t)\right)$ must be considered. In essence, the so-called velocity-density relationship has only statistical significance. The Lee velocity-density balanced function is given by

$$u = u_e\left(k(x,t)\right) = u_f \frac{1 - k(x,t)/k_j}{1 + G(k(x,t)/k_j)^{\theta}}, \tag{4}$$

where u_f represents the unimpeded velocity, indicating that the maximum velocity of pedestrian walking without congestion or obstruction. k_j represents the jamming density, indicating that pedestrians will not be able to flow freely and begin to form congestion when the pedestrian density exceeds the jamming density. G and θ are constants.

This paper chooses the Lee balanced function because this function can describe the pedestrian situation we study. Under different pedestrian densities, the factors affecting pedestrian velocity are also different. The balanced function expresses what factors are related to pedestrian velocity. In the discrete state, pedestrian flow is in the stage where individual behavior characteristics play a major role. Pedestrian velocity fluctuates nonlinearly with the change of environmental state and individual behavior characteristics. With the increase of pedestrian flow density, individual behavior is restricted, and the influence of different age distribution on pedestrian flow velocity is relatively small, which is basically linear. When pedestrians are in a high-density continuous flow state, the influence of individual behavior characteristics and changes in the surrounding environment on the high-density continuous flow state is almost negligible.

The Payne model in fluid dynamics is often solved using difference methods. These methods divide the continuous spatial and temporal domains into discrete grids or cells, and then use numerical approximation methods such as finite difference, finite element, etc., to transform the partial differential equations into difference equations. Difference methods can provide numerically stable solutions, ensuring that the obtained solution is reliable and accurate. This is crucial for complex fluid dynamics problems where exact solutions are difficult to obtain analytically.

We need to discretize the space and time, standardize the space step and the time step, and then discretize the Payne model. The space domain is x meters long and d meters wide. $[0, x]$ is discretized into M cells, where the i th cell is $I_i = [x_{i-1/2}, x_{i+1/2}]$ of length Δx. T^{pre} represents the prediction time. The time domain $[0, T^{pre}]$ is also discretized and divided into time level $\{t^n \mid n = 0,1,...\}$ and the time step Δt^n is given by $\Delta t^n = t^{n+1} - t^n$. Here we choose the equal interval time step Δt.

The space step Δx is generally set according to the working conditions, but the time step Δt is an important parameter in numerical simulation. If the time step is too large or too small, it will have a large deviation from the true value. Therefore, on the premise of ensuring the stability of the calculation, the results of the experimental data can be used as the verification data to select the time step or according to the time scale of the physical phenomenon to be studied. It is necessary to ensure that the time step can capture the flow characteristics of this physical phenomenon. At the same time, meeting the Courant-Friedrichs-Lewy (CFL) condition is a necessary condition to ensure the numerical stability and accuracy of the fluid model. It stipulates that the distance of the wave walking over a period of time is less than that of a specific unit. The equation is as follows

$$\Delta t \le C \frac{\Delta x}{\max(u(x,t))}, \tag{5}$$

where C is the coefficient of stability. In order to satisfy the stability condition, it is necessary to take the minimum limit on all grids as the period, so the maximum flow rate on all grids should be taken.

Algorithm 1 Location Prediction Based on Dynamics

Set T^{pre}, x, Δx, k_j, u_f, τ, d, Δt, etc.

Initialize vehicle locations $\mathbf{P}^{VC}(t_0)=\{P_n^{VC}(t_0)\mid n=1,2,...,N\}$ and pedestrian locations $\mathbf{P}^{PS}(t_0)=\{P_k^{PS}(t_0)\mid k=1,2,...,K\}$

for $t=t_0,t_0+\Delta t,t_0+2*\Delta t,...,T^{pre}-\Delta t$ **do**

  if $t=t_0$ **then**

    According to the initial location of pedestrians, the pedestrian density of all grids at time t can be obtained by the definition of pedestrian density

    for $i=1,2,...,M$ **do**

      Calculate the pedestrian velocity using (4)

    end for

  else

    for $i=2,3,...,M$ **do**

      The use of open boundary conditions at the entrance means that the measured data are completely used

      Calculate current pedestrian density on the pedestrian density at the previous moment using (6a)

    end for

    for $i=1,2,...,M$ **do**

      if $k_i^t\neq 0$ **then**

        Calculate pedestrian velocity using (6b)

      else

        Calculate pedestrian velocity using (4)

      end if

    end for

  end if

  for $k=1,2,...,K$ **do**

    According to the current location of the pedestrian $P_k^{PS}(t)$, the next moment location $P_k^{PS}(t+\Delta t)$ is calculated by the grid velocity and walking time of the pedestrian

  end for

  $\mathbf{P}^{PS}(t+\Delta t)=\{P_k^{PS}(t+\Delta t)\mid k=1,2,...,K\}$ can be obtained

  for $n=1,2,...,N$ **do**

    According to the latest locations of pedestrians, the vehicle locations are obtained by using the car following model

  end for

  $\mathbf{P}^{VC}(t+\Delta t)=\{P_n^{VC}(t+\Delta t)\mid n=1,2,...,N\}$ can be obtained

end for

The final predicted location $\mathbf{P}^{PS}(T^{pre})=\{P_n^{PS}(T^{pre})\mid n=1,2,...,N\}$ and $\mathbf{P}^{VC}(T^{pre})=\{P_n^{VC}(T^{pre})\mid n=1,2,...,N\}$ can be obtained.

For the Payne model, the finite difference about space and time is carried out, and the Euler integral method is used to discretize the model. Then the Payne prediction model can be given by

$$k_i^{n+1} = k_i^n - \frac{\Delta t}{\Delta x}(k_i^n u_i^n - k_{i-1}^n u_{i-1}^n), \tag{6a}$$

$$u_i^{n+1} = u_i^n - \Delta t[u_i^n \frac{u_i^n - u_{i-1}^n}{\Delta x} + \frac{1}{\tau}(u_i^n - u_e(k_i^n) + v\frac{k_{i+1}^n - k_i^n}{k_i^n \Delta x})], \tag{6b}$$

where the pedestrian density k_i^n and the pedestrian velocity u_i^n are used to represent the values of the approximate solution of cell I_i under time t^n.

In order to determine the task characteristics in advance, the location prediction algorithm uses the Payne model of fluid dynamics and the car-following model to predict locations for pedestrians and vehicles, respectively. **Algorithm 1** shows the location prediction process of pedestrians and vehicles.

4. Dynamics-Based Location Prediction and Neural Network Fine-Tuning Task Offloading Optimization Algorithm

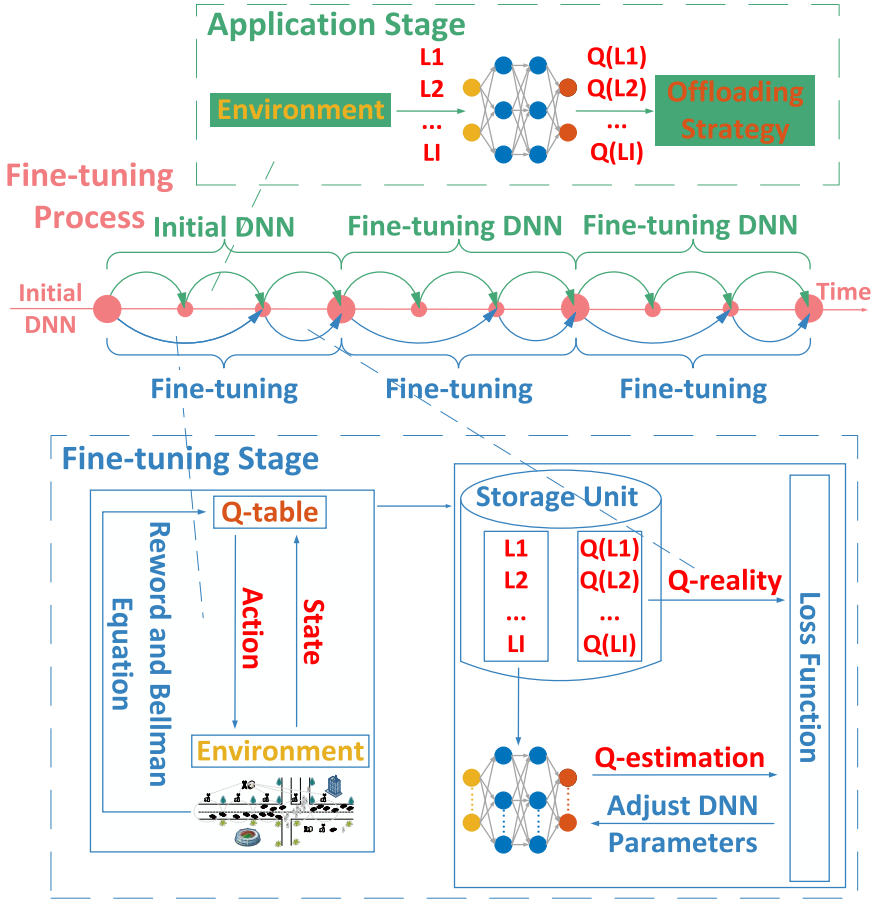


Fig. 2. A neural network fine-tuning task offloading optimization algorithm based on location prediction.

In this section, we introduce a neural network fine-tuning task offloading optimization algorithm based on location prediction.

In our previous research [25], when we use a neural network to optimize task offloading of

the pedestrian-vehicle interaction scenario, the neural network can make suitable task offloading decisions at the beginning. However, as pedestrians and vehicles change, when the difference in task characteristics at different times is relatively large, we continue to use pre-trained neural network to optimize task offloading, which leads to a decline in system performance. Since retraining the neural network takes too long, we consider continuously fine-tuning the neural network based on the similarity of task characteristics in adjacent time.

A neural network is fine-tuned periodically, it can slowly adapt to the scenario over time. However, when a congestion occurs suddenly in the scenario, the fine-tuning does not give a better task allocation in time. The normal driving of the vehicle will be affected and lead to the occurrence of accidents. In order to avoid this situation, it is necessary to predict the situation around the environment in advance, so that the neural network can perceive the changes of the environment in advance and fine-tune it, and then obtain better decisions in the first time. Finally, as shown in **Fig. 2**, we design a neural network fine-tuning task offloading algorithm, combining with location prediction for pedestrians and vehicles by the Payne model of fluid dynamics and the car-following model, respectively.

Specifically, the algorithm first requires a neural network used for decision-making previously, which is obtained by the deep Q-learning based algorithm. On the same timeline, with the change of task characteristics, we use the neural network to make decisions while predict the location of pedestrians and vehicles and fine-tuning another same neural network. Whenever the fine-tuning ends, we replace the decision-making neural network with the fine-tuned neural network. Finally, the fine-tuned neural network can be used to generate task offloading decisions with better performance and repeat the above process periodically. Fine-tuning means that we obtain small batches of experience from the surrounding environment through Q-learning over a period of time, and then train the neural network.

The algorithm contains the two following stages.

1) Fine-tuning stage: Each fine-tuning may improve the neural network's performance, so fine-tuning can be uninterrupted and periodic. The fine-tuning stage includes Q-learning sampling and neural network training. Before entering the fine-tuning stage, we first need to obtain an initial neural network to make decisions previously and get the decision results. In the proposed algorithm, a training sample are represented as

$$\mathbf{S}_{n,i,j} = \left\langle \mathbf{L}_{n,i}, \mathbf{Q}_t\left(\mathbf{L}_{n,i}, U_j\right)\right\rangle, \tag{7}$$

where $\mathbf{L}_{n,i} = \left\{C_{n,i}, Z_{n,i}, l_{n,i,j}, \lambda_j \middle| j = 1,2,...,J\right\}$ represents a state. $\mathbf{Q}_t\left(\mathbf{L}_{n,i}, U_j\right)$ denotes the Q-value vector of all J actions corresponding to state $\mathbf{L}_{n,i}$. $l_{n,i,j}$ represents the distance from the task $D_{n,i}$ to U_j.

After setting relevant parameters, i.e. learning rate α, discount factor γ, maximum number of Q-learning iterations E, maximum number of neural network training iterations F, and prediction time T^{pre}, we need to initialize the Q table, and use the location prediction to predict the location of pedestrians and vehicles at time $t + T^{pre}$ according to the initial location of pedestrians and vehicles at time t, and generate corresponding task characteristics. According to these task characteristics, all states at time $t + T^{pre}$ are constructed and input into the initial neural network. The obtained output is assigned to the Q table, and then the RSU corresponding to the maximum Q-value for each task is obtained through the Q table as the task offloading decision x_τ.

Next, a state $\mathbf{L}_{n,i}$ is selected randomly. Before selecting the RSU, the total offloading delay T^{before} is calculated. Then, an action U_j and the next state $\mathbf{L}_{n',i'}$ are selected randomly. After selecting the RSU, the total offloading delay T^{after} is calculated. The above two delay differences $T^{before} - T^{after}$ are used as the reward value of this state-action pair. The Q-value of this state-action pair according to the Bellman equation and the offloading strategy are updated if $T^{before} - T^{after}$ is positive. The corresponding samples are generated and stored into the storage unit. When the number of Q-learning iterations reaches E, we select all the new training samples to train the neural network. Then the above process is repeated uninterruptedly.

2) Strategy decision stage: This stage specifically means that we input the four parameters of each state into the fine-tuned neural network to make better offloading decision. The time required for the fine-tuning stage does not affect the time of the strategy decision stage. The proposed algorithm can quickly obtain task offloading decisions.

When using the proposed algorithm to optimize task offloading, in the fine-tuning stage, the number of samples obtained, the number of times required for neural network training, and the period of fine-tuning will affect the algorithm performance. Therefore, we need to optimize the parameters in the algorithm. We know that the number of Q-learning iterations determines the number of samples obtained, the number of neural network training iterations determines the learning effect of the neural network, and the fine-tuning period is determined by both, so the algorithm performance is also determined by them.

Because the training results of neural networks and the sampling process of Q-learning have randomness, the performance after fine-tuning is also affected by this randomness, and it is impossible to determine the specific expression between the fine-tuning performance and the two parameters, so the combination of parameters that achieves the optimal performance is searched by statistical methods.

Firstly, we need a metric to evaluate fine-tuning performance. The deep Q-learning based algorithm and the Q-learning based algorithm are considered as benchmark algorithms to evaluate the performance of the proposed algorithm. Let T be the duration of the scenario and $w(t)$ be the time delay obtained by task offloading of the scenario using one of the algorithms at moment t. Then the performance evaluation of the proposed algorithm is given by

$$PE = \frac{\sum_T w^{Proposed}(t) - \sum_T w^{QL}(t)}{\sum_T w^{DQL}(t) - \sum_T w^{QL}(t)}. \tag{8}$$

Secondly, the values of the two parameters are fixed, and then the proposed algorithm is used to optimize decisions of task offloading. After the optimization is completed, the performance evaluation under this combination of parameters is calculated using (8). Then we repeat the above process dozens of times and take the average value of PEs as the performance of the algorithm under this combination of parameters.

Finally, according to the statistical true values, we fit the surface diagram of parameters and PEs. The relationship between the two parameters and PE is not simply linear, so we use a non-parametric approach to analyze directly from the data. Local linear regression [29] is a non-parametric method used to fit nonlinear data. From the fitting results, the optimal parameter combination can be directly obtained.

5. Simulation Results and Analysis

In this section, we prove the prediction accuracy of the Payne model by comparing the prediction results of pedestrian flow. To verify the performance of the proposed algorithm, we conduct simulations. The simulation settings and result analysis are as follows.

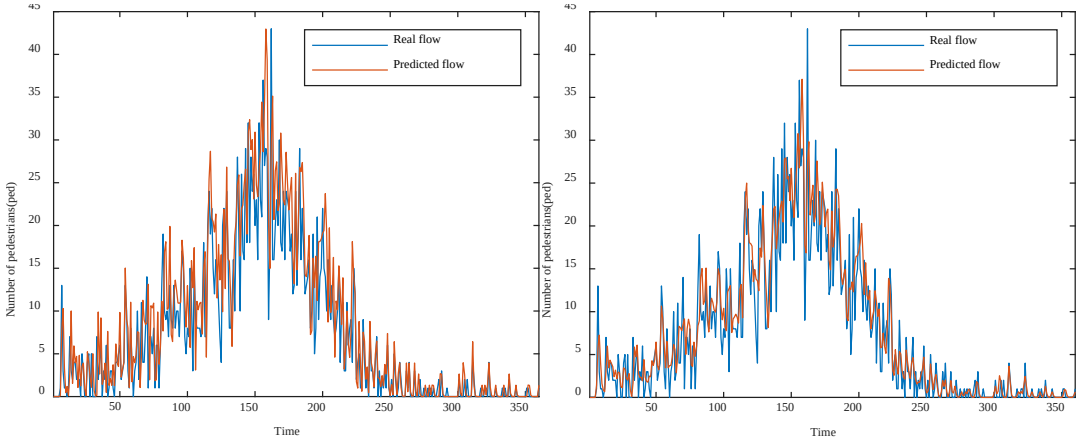
5.1 Prediction Accuracy

Table 1. Simulation parameters for prediction accuracy

Parameters	Value	Parameters	Value
τ	2 s	u_f	1.12 m/s
Δx	0.5 m	C	0.5
x	30 m	T^{pre}	1800 s
Δt	0.1 s	G	100
k_j	5.68 ped/m^2	θ	4

The pedestrian data of collected in this paper comes from a university road. The road is 30 meters long and 25 meters wide. We record pedestrian flow within 5 seconds each time for a total of 30 minutes. Based on historical pedestrian data, prediction simulation parameters can be determined. For the LWR model and the Payne model, the initial pedestrian density distribution is determined. The open boundary condition is used at the entrance, that is, the measured data is completely used, and then the model is used for prediction. The predicted location is the pedestrian flow at the section of 25 meters. **Table 1** shows simulation parameters for prediction accuracy.

1) Comparison of the predicted values of the true values of the LWR model and the Payne model.



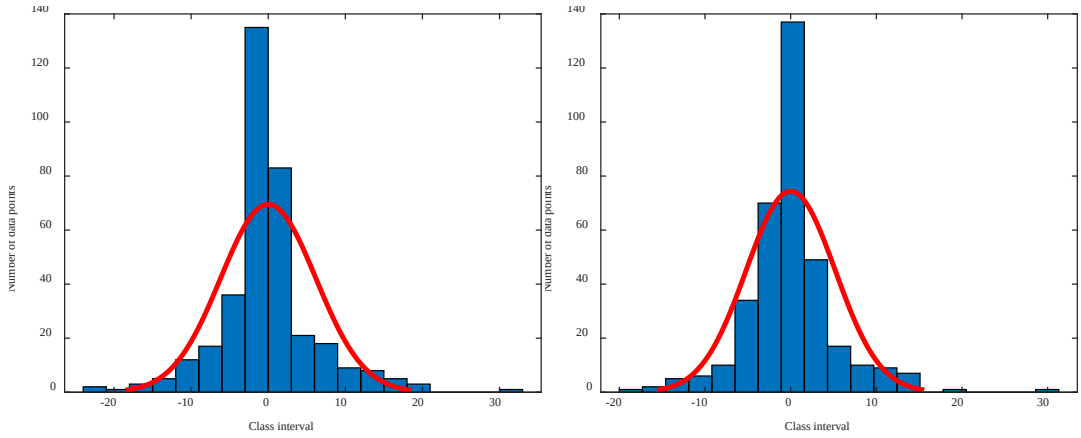
(a) LWR model (b) Payne model

Fig. 3. Comparison of true value and predicted value under different model prediction.

As shown in **Fig. 3**, at the peak time of commuting and classes on the university campus, the pedestrian density increases first and then decreases, and the whole process of pedestrian density is more concentrated. Considering the influence of the number of pedestrians on the size, type and number of tasks generated by the vehicles, we try to predict the precise values of the number of pedestrians. In order to evaluate the error between the model predicted values and the true values, the commonly used indicator is the Root Mean Square Error (RMSE).

According to the data calculation, the RMSE of the LWR model is 4.2917, while the RMSE of the Payne model is 3.3714, which indicates that the Payne model has better prediction performance.

2) The error histograms of the predicted and real values of the two models after difference respectively.



(a) LWR model error histogram (b) Payne model error histogram

Fig. 4. Comparison of error histograms predicted by different models.

In addition, the prediction performance of the model can be further evaluated by the error histogram. The error histogram shows the number of data points in different error ranges, which helps to understand the distribution of prediction bias of the model. As shown in Fig. 4, the abscissa represents a specific range of error, the ordinate represents the number of data points in a sample that fall within a specific range of error, and the red line is the normal distribution curve. When the error histogram is similar to the shape of normal distribution, it shows that the prediction error of most data points is small and concentrated near the average value, while the larger error is rare, which indicates that the model has high-precision prediction ability, with small and stable prediction biases. By comparing the error histograms of the two models, it can be observed that the sample data of the Payne model is more concentrated near the center and less distributed on both sides. In contrast, the sample data of the LWR model is less in the center and more distributed on both sides, which indicates that the Payne model has better prediction performance.

In summary, by comparing the RMSEs and error histograms of the two models, it is verified that the Payne model can predict the number of pedestrians more accurately.

3) The running time of location prediction algorithm.

The time complexity of Algorithm 1 is $O(n^2)$. The simulation was performed on a laptop with an i5-9300H processor and GTX1650 graphics card. Under this computer configuration, the running time of Algorithm 1 is 0.18 seconds.

5.2 Parameter Optimization

We use Matlab tools to simulate the scenario shown in **Fig. 2**. The laptop configuration used for the simulation was an i5-9300H processor and GTX1650 graphics card. The selected scenario is the part from the formation to the dissipation of the congestion of pedestrians and vehicles. We record 500 cycles and a cycle is 25 ms. Some parameters of **Table 1** are changed

due to the difference of pedestrian speed and road. **Table 2** shows simulation parameters for parameter optimization and performance evaluation.

Table 2. Simulation parameters for parameter optimization and performance evaluation

Parameters	Value	Parameters	Value
J	5	E	[100,9000]
K	13	F	[100,2000]
N	16	α	0.4
C	0.01~0.6 MB	γ	0.9
Z	0.01~0.6 GHz	T^{pre}	[0.14,57.24] s
$T^{\max}$	[20,200] ms	x	80 m
R	30 m	d	10 m
λ_j	[0,100] task/s	u_f	0.82 m/s

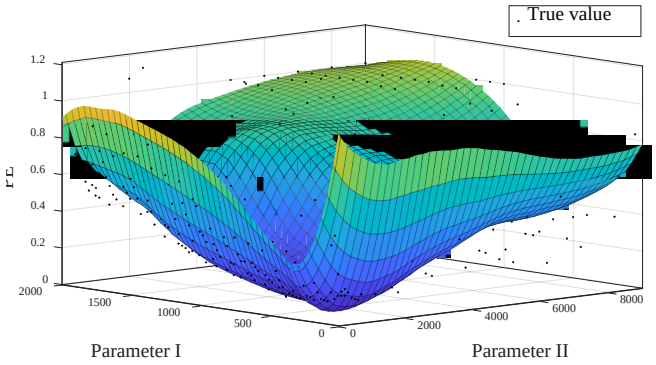


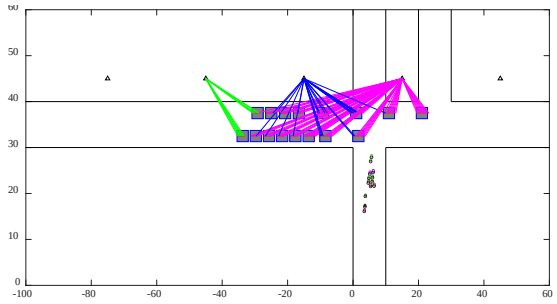
Fig. 5. Two parameters vs. PE.

The statistical true values and the three-dimensional surface fitting result are shown in Fig. 5. The R-square value after fitting is 0.8447, which is a statistic used to measure the goodness of the fitting model, and the closer the R-square is to 1, the more accurate the fitting is. Parameter I and parameter II represent the number of neural network training iterations and the number of Q learning iterations, respectively. Equation (8) shows that the smaller the PE is, the closer the performance of the proposed algorithm is to the optimum, and then the fine-tuning performance will be better. **Fig. 5** shows that there exists a combination of two parameters that makes the performance of fine-tuning optimal. According to the fitting result, the optimal parameters combination of the algorithm is the number of neural network training iterations 300 and the number of Q-learning iterations 1000. Later we use this optimal combination of parameters for algorithm performance evaluation.

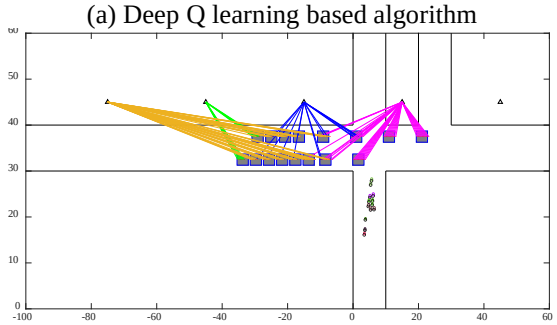
5.3 Performance Evaluation

The performance of the proposed a neural network fine-tuning task offloading algorithm combined with location prediction is evaluated by comparing the benchmark algorithms.

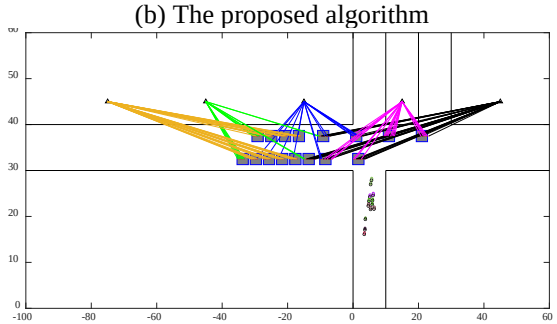
1) Comparison of the Results of Tasks Offloading to RSUs: In the simulation scenario, triangles, squares and circles represent RSUs, vehicles and pedestrians, respectively. The red dot on the square represents the task generated by the vehicle, and the connection of the red dot and the triangle represents the task chooses the RSU for offloading. When congestion occurs, results using different algorithms are shown in **Fig. 6**.



(a) Deep Q learning based algorithm



(b) The proposed algorithm



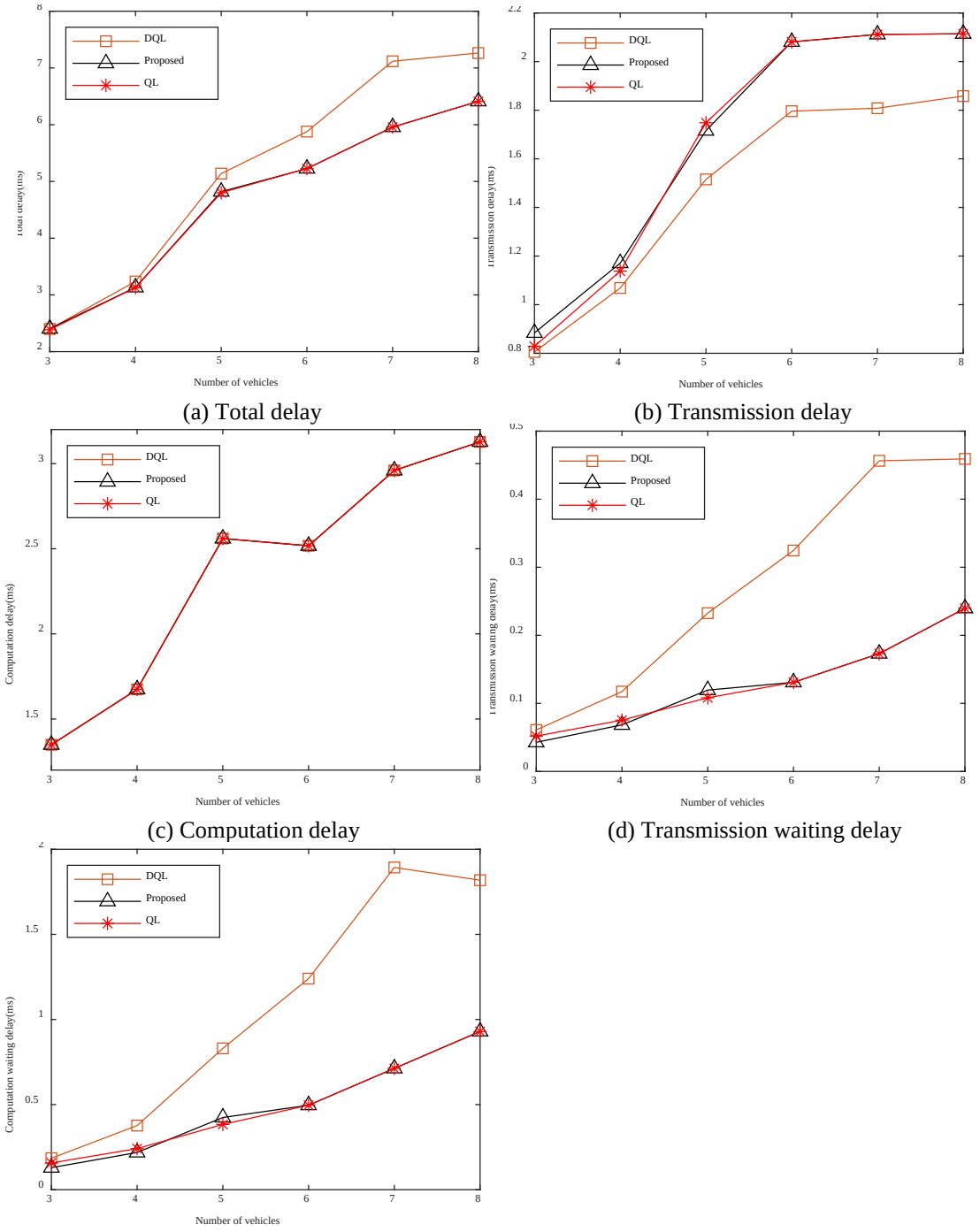
(c) Q learning based algorithm

Fig. 6. Comparison of the results of tasks offloading to RSUs.

As shown in **Fig. 6(a)**, the pre-trained neural network in deep Q-learning only learns the samples of the past time. When the current task characteristics are quite different from those of the past tasks, the offloading decisions made by this neural network are no longer applicable to the current task characteristics, which may lead to some of the RSUs not being fully utilized.

Fig. 6(b) shows that the proposed algorithm can achieve better task allocation than the deep Q-learning-based algorithm. The proposed algorithm continuously fine-tunes the neural network through new samples, and the neural network after each fine-tuning can make better decisions than before. Due to the insufficient number of fine-tuning, the neural network is unable to make the optimal task allocation decisions at present. However, as the number of fine-tuning increases over time, the decisions obtained by using the neural network are getting better and better, and even can achieve optimal task allocation.

With sufficient optimization time, the near-optimal task allocation can be achieved by the Q-learning based algorithm as shown in **Fig. 6(c)**, but each optimization time takes too long, which makes it impossible to complete the optimization before the new tasks are generated, so the task offloading cannot be optimized in real time.



(a) Total delay

(b) Transmission delay

(c) Computation delay

(d) Transmission waiting delay

(e) Computation waiting delay

Fig. 7. Comparison of offloading delay.

2) Comparison of Offloading Delay: In this study, the congestion stage of the 4th second is selected. As shown in **Fig. 7**, different algorithms are used to obtain the change of various delays as the number of vehicles grows.

Fig. 7(a) shows that as the number of vehicles grows, the total delays obtained by the three algorithms increase. **Fig. 7(b)** shows that the deep Q learning based algorithm cannot adapt to the change of task characteristics, so the tasks are offloaded to the nearest base stations, which has the best effect in terms of transmission delay. **Fig. 7(c)** shows that, due to the same task complexity, the computation delays are equal. **Fig. 7(d), 7(e)** show that the more the number of vehicles, the greater the waiting delay.

In terms of the total delay, the Q learning based algorithm is always optimal. As the number of vehicles grows, the total delay obtained by the deep Q learning based algorithm is very different from the optimal one due to the change of task characteristics, but the proposed algorithm can be highly close to the optimal effect.

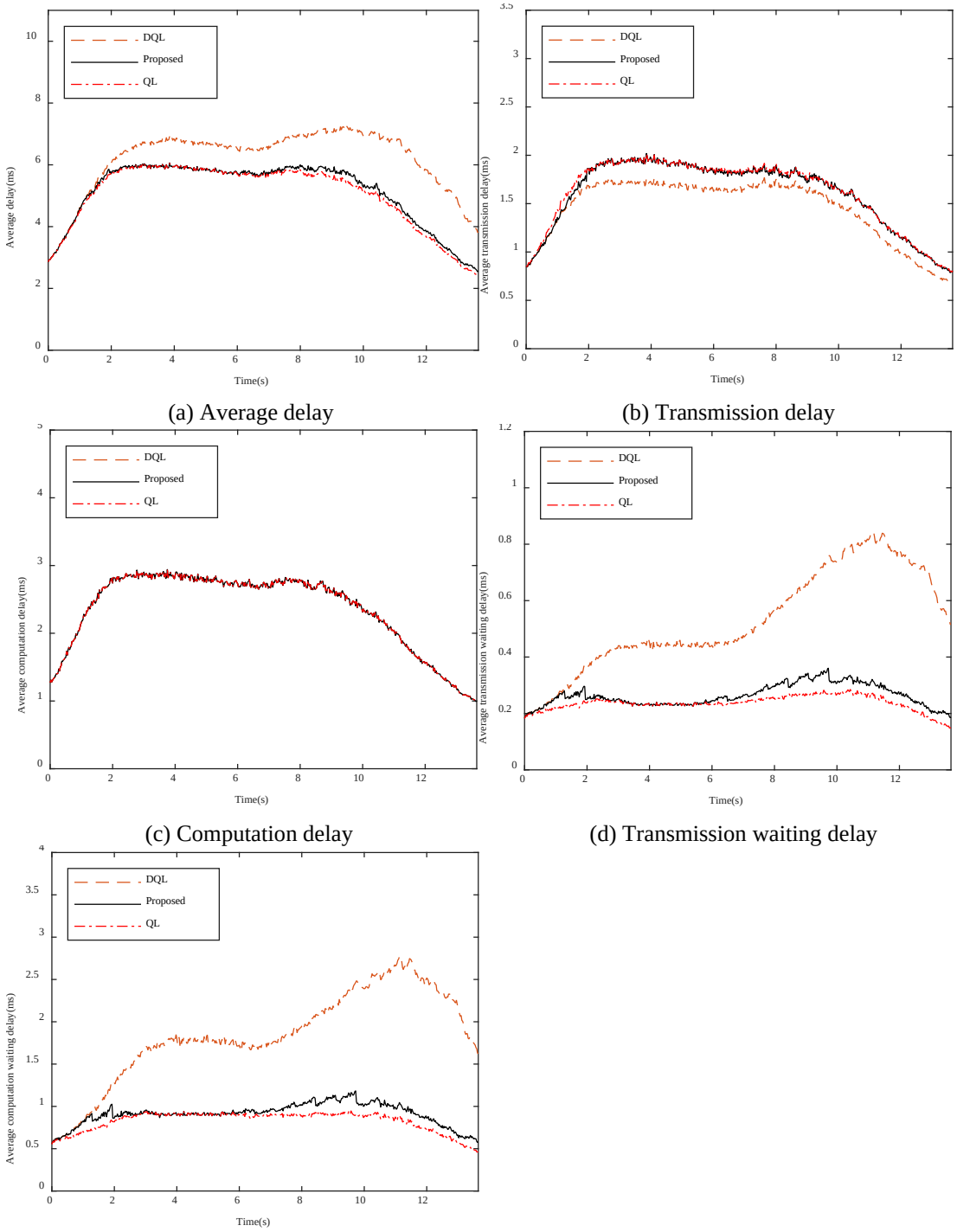
3) Comparison of Various Average Offloading Delay Throughout the Period: As shown in **Fig. 8**, different algorithms are used to obtain the change of various delays at 25 ms intervals.

Fig. 8(a) shows that, in terms of the average task offloading delay, the Q-learning based algorithm has the best performance. However, the algorithm takes a long time to run, which makes it impossible to optimize task offloading in real time. In terms of average delay, the deep Q-learning based algorithm can still close to the optimum at the beginning but the gap with the optimum keeps getting larger as characteristics of tasks change, indicating that the neural network is no longer applicable. The initial network used by the proposed algorithm is the same as the deep Q-learning based algorithm, so initially, both have the same performance. The proposed algorithm constantly predicts task characteristics and fine-tunes the neural network, so the average delay obtained is initially larger than the optimal one, and then it gradually approaches the optimal one, indicating that the neural network becomes applicable again. The prediction of task characteristics ensures that the neural network can give high performance task offloading decisions in time.

Fig. 8(b), 8(c) show that the deep Q-learning based algorithm cannot adapt to the change of task characteristics, so the tasks are offloaded to the nearest base stations and the transmission delay is the smallest. The data sizes of tasks generated at the same time during the whole period are the same. Because we assume that all RSUs have the same computing power, the computation delays of the three algorithms are equal.

Fig. 8(d), 8(e) show that, when congestion occurs, the number of tasks is the largest, resulting in the largest waiting delay for all algorithms. The deep Q-learning based algorithm offloads the tasks to the nearest base stations, which also lead to a larger waiting delay, especially when congestion occurs. During the whole period, the proposed algorithm continuously predicts locations and fine-tunes a neural network, and the obtained waiting delays are basically close to the optimal.

4) Comparison of Decision Time: The average simulation time from task generation to the acquisition of task offloading decision is 1769.2 ms, 5.3 ms, and 4.8 ms for the Q-learning based algorithm, the deep Q-learning based algorithm, and the proposed algorithm, respectively. In the pedestrian-vehicle interaction scenario, the cycle for the vehicle to generate new tasks is very short. It can be seen from the decision time of each algorithm, the Q-learning based algorithm runs for a long time, which makes it impossible to complete the optimization before the new tasks are generated, so it is not suitable for the vehicular networks with high real-time requirements. The deep Q-learning based algorithm and the proposed algorithm combine a neural network that can learn the complex relationship between task characteristics and decisions, exhibiting a certain degree of generalization ability. When new tasks are generated, the neural network can also obtain offloading decisions in time.



(a) Average delay

(b) Transmission delay

(c) Computation delay

(d) Transmission waiting delay

(e) Computation waiting delay

Fig. 8. Comparison of various average offloading delay throughout the period.

6. Conclusion

This paper proposed a neural network fine-tuning task offloading algorithm, combining with location prediction for pedestrians and vehicles by the Payne model of fluid dynamics and the car-following model, respectively, so the task characteristics can be determined in advance. The algorithm periodically predicted locations of pedestrians and vehicles and fine-tuned a neural network for task offloading optimization. After fine-tuning the neural network using these task characteristics, the most suitable task offloading decision could be obtained. With the change of task characteristics in the scenario, the algorithm proposed in this paper could periodically predict locations and fine-tune the neural network to maintain high system performance and meet the low delay requirement. From the simulation results, compared with other algorithms, the proposed algorithm still guaranteed a lower task offloading delay, especially when congestion occurs.

Acknowledgement

This research was supported by Anhui Provincial Natural Science Foundation under grant no. 2308085MF194.

References

[1] A. Waheed et al., "A Comprehensive Review of Computing Paradigms, Enabling Computation Offloading and Task Execution in Vehicular Networks," *IEEE Access*, vol. 10, pp. 3580-3600, 2022. Article (CrossRef Link)

[2] H. Zhao, Q. Zhu, Y. Chen, and Y. Zhu, "A Research of Task-Offloading Algorithm for Distributed Vehicles," in *Proc. of 2020 IEEE International Conference on Communications Workshops (ICC Workshops)*, pp. 1-5, 2020. Article (CrossRef Link)

[3] Q. Wu, H. Ge, H. Liu, Q. Fan, Z. Li, and Z. Wang, "A Task Offloading Scheme in Vehicular Fog and Cloud Computing System," *IEEE Access*, vol. 8, pp. 1173-1184, 2020. Article (CrossRef Link)

[4] C. Mouradian, D. Naboulsi, S. Yangui, R. H. Glitho, M. J. Morrow, and P. A. Polakos, "A Comprehensive Survey on Fog Computing: State-of-the-Art and Research Challenges," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 1, pp. 416-464, Firstquarter 2018. Article (CrossRef Link)

[5] M. Asim, Y. Wang, K. Wang, and P. -Q. Huang, "A Review on Computational Intelligence Techniques in Cloud and Edge Computing," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 4, no. 6, pp. 742-763, Dec. 2020. Article (CrossRef Link)

[6] P. Mach and Z. Becvar, "Mobile Edge Computing: A Survey on Architecture and Computation Offloading," *IEEE Communications Surveys & Tutorials*, vol.19, no.3, pp. 1628-1656, thirdquarter 2017. Article (CrossRef Link)

[7] Y. Sun et al., "Adaptive Learning-Based Task Offloading for Vehicular Edge Computing Systems," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 4, pp. 3061-3074, April 2019. Article (CrossRef Link)

[8] P. Dass and S. Misra, "DeTTO: Dependency-Aware Trustworthy Task Offloading in Vehicular IoT," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 12, pp. 24369-24378, Dec. 2022. Article (CrossRef Link)

[9] R. Li, C. S. Lim, M. E. Rana, and X. Zhou, "A Trade-Off Task-Offloading Scheme in Multi-User Multi-Task Mobile Edge Computing," *IEEE Access*, vol. 10, pp. 129884-129898, 2022. Article (CrossRef Link)

[10] P. Liu, J. Li, and Z. Sun, "Matching-Based Task Offloading for Vehicular Edge Computing," *IEEE Access*, vol. 7, pp. 27628-27640, 2019. Article (CrossRef Link)

[11] S. Misra and S. Bera, "Soft-VAN: Mobility-Aware Task Offloading in Software-Defined Vehicular Network," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 2, pp. 2071-2078, Feb. 2020. Article (CrossRef Link)
[12] C. Hou and Q. Zhao, "Optimal Task-Offloading Control for Edge Computing System with Tasks Offloaded and Computed in Sequence," *IEEE Transactions on Automation Science and Engineering*, vol. 20, no. 2, pp. 1378-1392, April 2023. Article (CrossRef Link)
[13] Y. Yuan, X. Xu, M. Sun, and P. Zhang, "Terminal Cooperative Interdependent Computing Task Offloading for 6G," *IEEE Transactions on Network Science and Engineering*, vol. 9, no. 4, pp. 2846-2856, July-Aug. 2022. Article (CrossRef Link)
[14] H. Li, X. Li, M. Zhang, and B. Ulziinyam, "Multicast-Oriented Task Offloading for Vehicle Edge Computing," *IEEE Access*, vol. 8, pp. 187373-187383, 2020. Article (CrossRef Link)
[15] J. Zhao, Q. Li, Y. Gong, and K. Zhang, "Computation Offloading and Resource Allocation for Cloud Assisted Mobile Edge Computing in Vehicular Networks," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 8, pp. 7944-7956, Aug. 2019. Article (CrossRef Link)
[16] J. Liu, J. Ren, Y. Zhang, X. Peng, Y. Zhang, and Y. Yang, "Efficient Dependent Task Offloading for Multiple Applications in MEC-Cloud System," *IEEE Transactions on Mobile Computing*, vol. 22, no. 4, pp. 2147-2162, April 2023. Article (CrossRef Link)
[17] J. Zhao, X. Sun, X. Ma, H. Zhang, F. R. Yu, and Y. Hu, "Online Distributed Optimization for Energy-Efficient Computation Offloading in Air-Ground Integrated Networks," *IEEE Transactions on Vehicular Technology*, vol. 72, no. 4, pp. 5110-5124, April 2023. Article (CrossRef Link)
[18] Y. Liu, H. Yu, S. Xie, and Y. Zhang, "Deep Reinforcement Learning for Offloading and Resource Allocation in Vehicle Edge Computing and Networks," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 11, pp. 11158-11168, Nov. 2019. Article (CrossRef Link)
[19] B. Yang, X. Cao, J. Bassey, X. Li, T. Kroecker, and L. Qian, "Computation Offloading in Multi-Access Edge Computing Networks: A Multi-Task Learning Approach," in *Proc. of 2019 IEEE International Conference on Communications (ICC)*, Shanghai, China, pp. 1-6, 2019. Article (CrossRef Link)
[20] J. Shi, J. Du, J. Wang, J. Wang, and J. Yuan, "Priority-Aware Task Offloading in Vehicular Fog Computing Based on Deep Reinforcement Learning," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 12, pp. 16067-16081, Dec. 2020. Article (CrossRef Link)
[21] H. Guo, J. Liu, J. Ren, and Y. Zhang, "Intelligent Task Offloading in Vehicular Edge Computing Networks," *IEEE Wireless Communications*, vol. 27, no. 4, pp. 126-132, Aug. 2020. Article (CrossRef Link)
[22] F. Sun, Z. Zhang, X. Chang, and K. Zhu, "Toward Heterogeneous Environment: Lyapunov-Orientated ImpHetero Reinforcement Learning for Task Offloading," *IEEE Transactions on Network and Service Management*, vol. 20, no. 2, pp. 1572-1586, June 2023. Article (CrossRef Link)
[23] C. -H. Hsu, Y. Chiang, Y. Zhang, and H. -Y. Wei, "Mobility-Aware QoS Promotion and Load Balancing in MEC-Based Vehicular Networks: A Deep Learning Approach," in *Proc. of 2021 IEEE 93rd Vehicular Technology Conference (VTC2021-Spring)*, Helsinki, Finland, pp. 1-6, 2021. Article (CrossRef Link)
[24] J. Wang, J. Hu, G. Min, W. Zhan, A. Y. Zomaya, and N. Georgalas, "Dependent Task Offloading for Edge Computing based on Deep Reinforcement Learning," *IEEE Transactions on Computers*, vol. 71, no. 10, pp. 2449-2461, 1 Oct. 2022. Article (CrossRef Link)
[25] D. Zheng, L. Wang, C. Kai, and M. Peng, "Resource Optimization for Task Offloading with Real-Time Location Prediction in Pedestrian-Vehicle Interaction Scenarios," *IEEE Transactions on Wireless Communications*, vol. 22, no. 11, pp. 7331-7344, Nov. 2023. Article (CrossRef Link)
[26] Lighthill, Michael James and Gerald Beresford Whitham. "On kinematic waves II. A theory of traffic flow on long crowded roads," *Proceedings of the Royal Society of London Series A*, vol. 229, no. 1178, pp. 317-345, May 1955. Article (CrossRef Link)
[27] Paul I. Richards, "Shock Waves on the Highway," *Operations Research*, vol. 4, no. 1, pp. 42-51, Feb. 1956. Article (CrossRef Link)

[28] Payne, H.J, "Models of Freeway Traffic and Control," *Mathematical Models of Public Systems*, vol. 28, pp. 51-61, January 1971.
[29] William S. Cleveland and Susan J. Devlin, "Locally Weighted Regression: An Approach to Regression Analysis by Local Fitting," *Journal of the American Statistical Association*, vol. 83, no. 403, pp. 596-610, 1988. Article (CrossRef Link)

Yuanguang Wu received the B.E. degree in Marine Technology from Zhejiang Ocean University, Zhoushan, China, in 2021. He is currently a master student in Communications Engineering Department at Hefei University of Technology, Hefei, China. His research interests are wireless communications, task offloading in vehicular networks, and edge computing.

Lusheng Wang received his B.Sc. in Communications Engineering in 2004 from Beijing University of Posts and Telecommunications (BUPT), China and his Ph.D. in 2010 in Computer Science and Networks from Telecom ParisTech (ENST), France. Currently, he is a research professor of Communications Engineering Department at Hefei University of Technology (HFUT), China. His research interests are resource allocation in wireless communication systems, task offloading in vehicular networks, pedestrian flow prediction, and chaotic time series prediction in transportation and finance.

Caihong Kai received the B.S. degree from Hefei University of Technology, Hefei, China, in 2003, the M.S. degree in Electronic Engineering and Computer Science from University of Science and Technology of China, Hefei, China, in 2006, and the Ph.D. degree in Information Engineering from the Chinese University of Hong Kong, Hong Kong, China, in 2010, respectively. She is now a Professor of the School of Computer Science and Information Engineering, Hefei University of Technology. Her research interests are in wireless communication and networking, network protocols, and performance evaluation.

Min Peng received the B.S. and Ph.D. degrees from the Department of Electronic Engineering and Information Science, University of Science and Technology of China (USTC), Hefei, China, in 2005 and 2010, respectively. From 2011 to 2013, he was a Research Engineer with the Huawei Nanjing Research Institute, Nanjing. In 2013, he joined the Department of Communication Engineering, Hefei University of Technology, Hefei, China. His current research interests include delay-tolerant networks, wireless communication, IoT, indoor locationing, and sensor networks.