Complex & Intelligent Systems (2025) 11:47
https://doi.org/10.1007/s40747-024-01651-5

ORIGINAL ARTICLE

# Pri-DDQN: learning adaptive traffic signal control strategy through a hybrid agent

Yanliu Zheng[1] · Juan Luo[1] · Han Gao[1] · Yi Zhou[2] · Keqin Li[3]

Received: 25 July 2022 / Accepted: 22 October 2024

**Abstract**
Adaptive traffic signal control is the core of the intelligent transportation system (ITS), which can effectively reduce the pressure on traffic congestion and improve travel efficiency. Methods based on deep Q-leaning network (DQN) have become the mainstream to solve single-intersection traffic signal control. However, most of them neglect the important difference of samples and the dependence of traffic states, and cannot quickly respond to randomly changing traffic flows. In this paper, we propose a new single-intersection traffic signal control method (Pri-DDQN) based on reinforcement learning and model the traffic environment as a reinforcement learning environment, and the agent chooses the best action to schedule the traffic flow at the intersection based on the real-time traffic states. With the goal of minimizing the waiting time and queue length at intersections, we use double DQN to train the agent, incorporate traffic state and reward into the loss function, and update the target network parameters asynchronously, to improve the agent's learning ability. We try to use the power function to dynamically change the exploration rate to accelerate convergence. In addition, we introduce a priority-based dynamic experience replay mechanism to increase the sampling rate of important samples. The results show that Pri-DDQN achieves better performance, compared to the best baseline, it reduces the average queue length is reduced by 13.41%, and the average waiting time by 32.33% at the intersection.

**Keywords** Adaptive traffic signal control (ATSC) · Double DQN · Decay $\varepsilon$-greedy · Priority-based experience replay · Reinforcement learning (RL)

## Introduction

### Background

With the rapid development of the economy and the acceleration of urbanization, the urban transportation system has become one of the main public infrastructures of the city. However, the infrastructure is unable to cope with the rapidly growing number of vehicles, which causes many traffic problems in the city, especially traffic congestion during peak hours. Traffic congestion has become a worldwide "urban disease", which causes heavy losses in terms of economy, energy and time every year. For example, due to traffic congestion, American loses $68 billion a year, Beijing emits an extra 16,700 tons of carbon dioxide and every person average delay 66 min everyday. Traffic congestion has become a global issue that must be addressed.

In recent years, countries around the world are taking a series of measures to solve traffic congestion. On the one hand, these countries have improved infrastructure construc-

✉ Juan Luo
juanluo@hnu.edu.cn

Yanliu Zheng
yanliu_zheng@hnu.edu.cn

Han Gao
774144163@qq.com

Yi Zhou
zhouyi@henu.edu.cn

Keqin Li
lik@newpaltz.edu

[1] College of Computer Science and Electronic Engineering, Hunan University, Lushan South Road, Changsha 410082, Hunan, China

[2] The School of Artificial Intelligence, Henan University, Mingli Road, Zhengzhou 450046, Henan, China

[3] The Department of Computer Science, State University of New York at New Paltz, New York, NY 12561, USA

Published online: 18 November 2024

tion, planned urban roads and increased the carrying capacity of traffic. However, the implementation of the infrastructure wastes a lot of time and money. On the other hand, many countries are working to develop intelligent transportation by using existing transportation infrastructure to improve traffic control and management. The Vehicle Infrastructure Cooperative System(VICS) technology can dynamically collects and fuses traffic information, which helps realize information interaction between vehicles and road side infrastructure.

Figure 1 shows a VICS traffic signal control scenario. $p_1$ is the current phase of the traffic light, which means the vehicle in south-north has the right to pass through. VICS controls traffic lights at intersections in real-time to reduce traffic congestion, pollution, and number of traffic accidents. In conclusion, the traffic signal adaptive control is an excellent approach to alleviate traffic congestion. Because of the complexity of the ITS and the heterogeneity and diversity of the infrastructure, the data in VICS presents variety and volume, which increases the difficulty of processing data. However, the real-time and high-efficiency requirements of ATSC mean it urgently needs efficient data fusion processing and a real-time intelligent decision-making traffic signal control algorithm.

Intersections are the basic nodes in the urban traffic system, which are the location for vehicles and pedestrians to converge, turn and evacuate. It is also the main body of traffic signal control. Thus, the control and optimization of traffic signal for a single intersection can effectively alleviate urban traffic congestion [1]. Scholars studied traffic signal control and implemented a large number of projects known as real-time offline control. These studies predicted the traffic flow online and selected the control strategy in accordance with the prediction results. Many traffic signal control systems, such as SCATS, TRANSYT, and SCOOT, use this approach extensively.

Traditional traffic signal control methods are dependent on models and cannot dynamically change traffic signal management strategies. It's hard to build an appropriate model for traffic signal light, because the traffic flows are complex, dynamic and changeable. Reinforcement learning can achieve good learning performance in large spaces and complex non-linear systems without mathematical models and prior knowledge of the environment.Therefore, reinforcement learning [2] has become an important filed in intelligent transportation research.

## Contributions

We propose a multi-objective optimization adaptive traffic signal control algorithm, named Pri-DDQN, to improve the flexibility and intelligence of traffic signal control. The contributions of this paper are as follows:

(1) We build a hybrid agent for the traffic light based on the cognitive agent and reactive agent structure. The hybrid agent learns traffic information from the environment and refines the traffic state from lane level to vehicle level. Meanwhile, it makes real-time control strategy to optimize the randomly changing traffic flow.

(2) Pri-DDQN achieves the adaptive traffic signal control by optimizing the waiting times and queue length at intersections. Pri-DDQN adds two convolutional layers for the agent to finely extract traffic environment characteristics, which enhances the expressive ability of the agent. Pri-DDQN asynchronously updates the target network parameters to improve the learning ability of the agent to quickly make ATSC strategy.

(3) We use power function to dynamically change the exploration rate to accelerate the convergence of Pri-DDQN. Meanwhile, we establish a priority-based dynamic experience replay mechanism to increase the sampling rate of important samples. The sampling rate and priority of samples are changing with the network and training time.

## Related works

Traffic signal control has started in the 20th century. At first, the single-point control mode with fixed time is adopted to calculate the optimal phase segmentation or cycle length of the intersection by analyzing offline traffic data. The theoretical basis of the fixed-time scheme was the steady-state stochastic delay model of unsaturated traffic flow proposed by Webster [3] in 1958. In this theory, the traffic state took the delay time of vehicles as the evaluation index. Zhang et al. [4] improved the Webster algorithm by applying the traffic flow fluctuation theory. The authors proposed the calculation formula of the shortest signal cycle at the intersection by considering the effect of shock waves on the queuing length of vehicles. The above works have studied the traffic model based on fixed-time methods. However, fixed-time methods are scheduled and optimized based on past traffic data, which cannot satisfy the real-time traffic demand.

The artificial intelligence also affects the field of intelligent transportation. New methods are introduced into traffic signal control, such as genetic algorithm, particle swarm optimization algorithm, neural network [5] and fuzzy control [6]. Thus, the mode of traffic signal control has changed from fixed duration to semi-adaptive. Liu et al. [7] designed a fuzzy control system based on a four-phase phasing sequence to control traffic lights at a single intersection. They adopted genetic algorithm with an elite reservation strategy to optimize the fuzzy rules and membership function in the fuzzy control system. Bi et al. [8] evaluated turning vehicles and lane length, established the main road traffic flow and evaluation model, and proposed a coordinated arterial traffic type-2

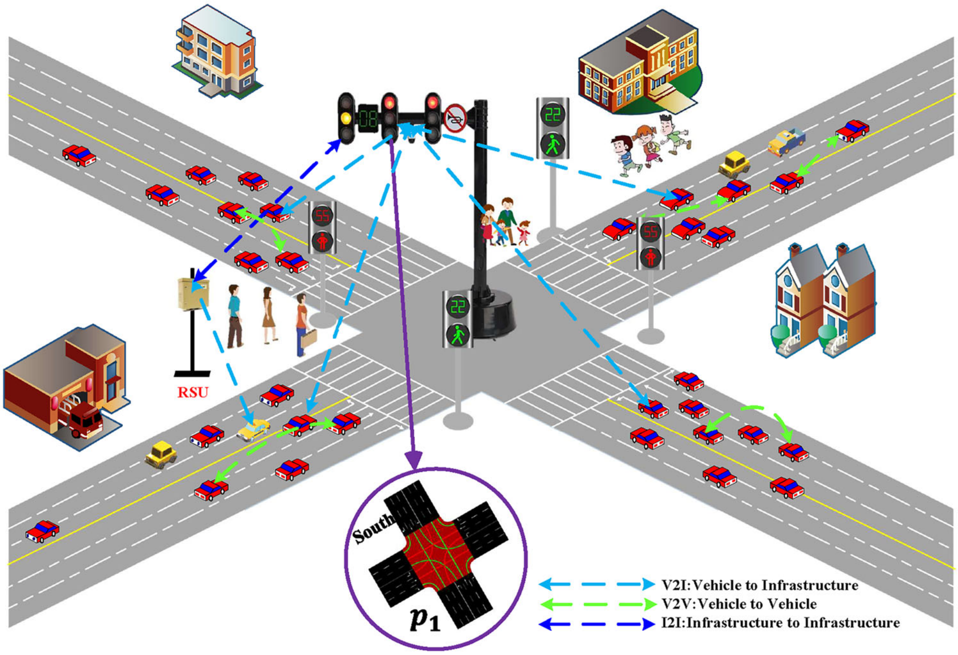

RSU
South
$p_1$
V2I:Vehicle to Infrastructure
V2V:Vehicle to Vehicle
I2I:Infrastructure to Infrastructure

**Fig. 1** The scenario of adaptive traffic signal control

fuzzy logic control method to alleviate the pressure of the main road. Li et al. [9] mined the topology information of the road network by using the graph convolutional neural network, and integrated human knowledge and experience into the model through the deep imitation learning method to realize the real-time adjustment of traffic control strategy in accordance with traffic conditions. However, the methods of above works are presented based on historical data. Thus, their methods cannot effectively control the traffic signal light when the traffic flows are complex, dynamic and changeable.

In order to simulate the complex and changeable traffic flow environment, the traffic signal control models based on cellular automata are developed. Sanchez-Medina [10] developed a traffic micro-simulator based on cellular automata, which can simulate many situations, such as overtaking and multilane traffic. At the same time, a traffic signal regulation model optimized by genetic algorithm was designed and implemented on the Beowulf cluster. Most of them cannot adapt to dynamic and sudden traffic demands because they have used the centralized control methods that is poor in scalability.

To ensure the real-time performance of the control strategy, real-time traffic data must be taken as input, and the duration of traffic lights must be dynamically adjusted accordingly. Reinforcement learning provides an effective solution for real-time traffic signal regulation. In 1989, Watkins [11] proposed the Q-learning algorithm, which did not require pre-modeling and was highly adaptable to the external environment. The Q-learning algorithm was suitable for traffic control problems and had attracted the attention of a large number of scholars. Yu et al. [12] presented the signal control issue as a Markov model choice problem and proposed a signalized cross-section system adaptive control model based on Markov's discrete decision-making process. In [13], the topic of traffic signal control was represented as a reinforcement learning issue. The authors automatically extracted all important information from real-time traffic data, learned and outputted the optimal traffic signal control strategy, and introduced experiential replay and target Q-network to solve the instability problem of reinforcement learning. However, with the rapid growth of the state space size, these methods cannot be applied to large networks.

As the number of traffic lights growing, model becomes complex. Traditional reinforcement learning relies on the assumption of simplified state and manual feature extraction [14], and the value and strategy functions are simple and cannot deal with high-dimensional real-time traffic information. Deep learning addresses these problems by combining it with reinforcement learning with decision making abilities. Liang et al. [15] discretized complex traffic scenes by dividing intersections into small grids. Then,the authors modeled the traffic light time that changed between two adjacent cycles as a high-dimensional Markov decision process, which increased or decreased the selected phase time from the next cycle stage.

To overcome traffic signal control challenges, researchers are currently combining a range of reinforcement learning algorithms with priority experience replay approaches in the control model, which can learn better strategies in normal traffic flow rates. Zhong et al. [16] proposed the Nature DQN algorithm to optimize the signal control strategy of a single intersection. The Nature DQN algorithm controled the intersection traffic signals, and the optimal signal scheme was sought through implicit modeling to control the changes in actions and environmental states. Li et al. [17] designed a traffic signal timing plan. The authors used the deep-stacked autoencoder neural network to estimate the Q function and found an appropriate signal timing strategy by implicitly modelling control behavior and system state changes. However, in the experiment, the traffic environment modeling was simple and only considered the situation of the straight lane. Lee et al. [18] improved the extensibility of reinforcement learning algorithms to achieve global optimization. It shared CNN parameters with all intersections, and the parameters of the last hidden layer and the output layer were fixed to 1 to overcome the disaster of bits in state and action space.

Combining with the theory of maximum pressure in the field of transportation, Huawei et al. [19] improved the overall network throughput and minimized travel time by defining intersection pressure, and improved the value of intersection pressure as a reward in reinforcement learning. Liang et al. [20] proposed a DRL model to control the cycle of traffic lights by combining various optimization elements for enhancing performance, i.e., dueling network, target Q-network, double Q-network, and priority experience replay. The model was verified to be effective on the SUMO simulation platform. Zheng et al. [21] proposed the FRAP design scheme based on the intuitive notion of phase competition in traffic signal control, that is, the signal with high traffic volume (i.e., high demand) would be given priority when two traffic signals conflict. In addition, the invariance of the symmetry, such as turning over and rotation in the traffic flow, was realized. Zang et al. [22] proposed a new framework MetaLight, which was based on meta-learning algorithm, including periodically alternating individual-level adaptation and global level adaptation.

In addition, when optimizing the traffic signal cycle duration at a single intersection, most existing methods only optimize one objective, such as the average delay of traffic at the intersection, the number of stops, queue length, delay, and through-flow volume, etc., [23, 24]. Reference [25]provided a detailed review of the objectives that can be optimized, including liquidity objectives and sustainability objectives. Most research into the traffic signal control problem has primarily used mobility objectives.Without considering the different requirements of different traffic states for control indicators, and have their limitations. Therefore, it is necessary to weigh multiple optimization objectives in a comprehensive manner.

In summary, existing traffic signal control methods, such as fuzzy control, Q-learning, DQN, and max-pressure, can achieve adaptive traffic signal control and alleviate traffic congestion. However, most of them neglect the important difference between samples and the dependence among traffic states. Their algorithms are inefficient and cannot quickly adapt to the dynamic changes in the traffic environment. Reducing the waiting time and queue length when vehicles pass through the intersection, and flexibly adjusting the traffic signal control strategy at the intersection, which are of great significance for alleviating traffic congestion. Therefore, there is an urgent need for flexible reinforcement learning methods to adaptively control traffic signals.

## Traffic model

### Hybrid traffic signal control agent model

The cognitive agent and the reactive agent have different characteristics. Cognitive agents are knowledge-based, and their environment models are known in advance, which are not suitable for dynamically changing traffic scenario. The reactive agent associates the perception with the action through the condition-action rule, the degree of intelligence is low. Therefore, it is not the best way to construct a TSC agent by only using a cognitive agent or a reactive one.

Considering the characteristics of the cognitive agent and the reactive agent, we construct a hybrid TSC agent based on "Perception-Cognitive-Behavior" model, as shown in Fig. 2. The hybrid agent has higher intelligent and adapt to dynamically changing traffic flow to efficiently control the traffic signal at intersections. Simultaneously, we use normalization to improve the discrete traffic state coding to enhance the intelligence of the agent.

The process of the hybrid TSC agent is as follows. Firstly, the hybrid TSC agent observes the traffic environment such as vehicle position, speed, and signal phase at the intersection. They are used as inputs of the Pri-DDQN, and the important features of the traffic environment are extracted through convolution operations. Secondly, the agent learns the decision-making model of the intersection through the priority-based experience replay mechanism, and updates the model online to control the traffic signal. Finally, the traffic environment enters a new state and gives an immediate reward to the agent. Subsequently, the agent relearns from the environment and makes new control strategies continually, to optimize traffic at intersections.

### Traffic environment

The traffic environment includes intersections, roads, and traffic lights. The intersection consists of four roads,and a traffic light. Each road is 300 ms and is divided into three lanes, i.e., left turn, straight and right turn, as shown in Fig. 3a. We only pay attention to the entrance lane information, because the vehicle which passed the intersection has no effect on the signal control. So there are twelve control signals in the intersection, which is expressed as $p_i = \{l_1, l_2, l_3, \cdots, l_{12}\}$. The control signal in the south

**Fig. 2** The hybrid TSC agent model

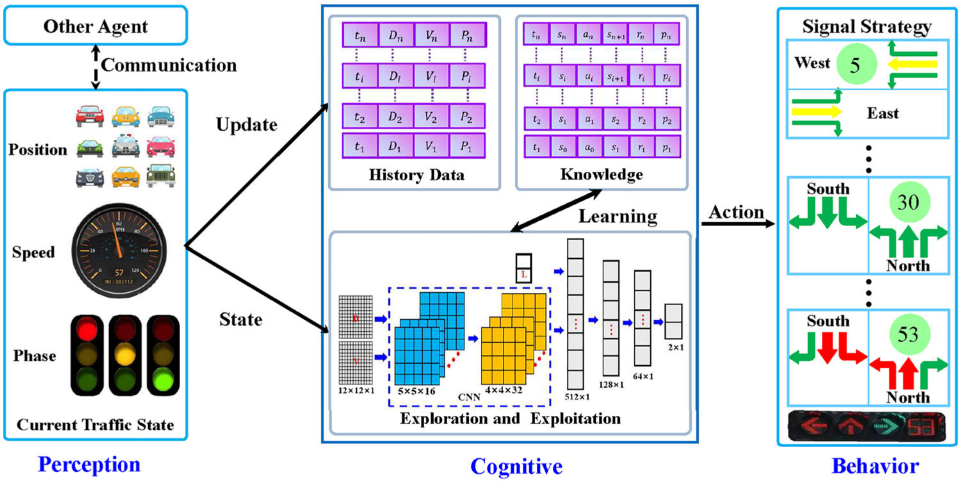
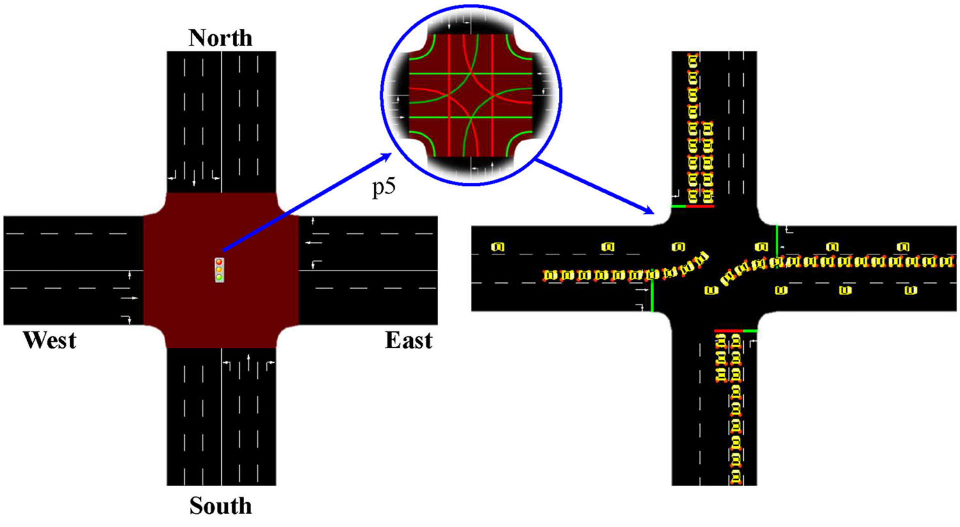


**Fig. 3** Intersection model



(a) Intersection

(b) Traffic control signal priority

entrance is expressed as $\{l_1, l_2, l_3\}$. $\{l_4, l_5, l_6\}$ is the east entrance control signal. $\{l_7, l_8, l_9\}$ is the control signal of the north entrance, while $\{l_{10}, l_{11}, l_{12}\}$ is the control signal of the west entrance. The control signal for the straight lane has a high priority than that of the left turn lane. We can see this in Fig. 3b, in phase $p_5$, there both the left turn lane and the straight lane are green, the vehicles on left turn lane give way to the vehicles on the straight.

In our environment, the basic traffic signal control strategy has eight phases, as shown in Table 1. $G$ represents the green signal with a high priority, and the vehicle can pass through the intersection without halting. $\{GGgGrrGGgGrr\}$ means that the north–south straight lane and all right-turn lane have the right of way, and the other lanes have no right of way. $g$ represents the regular green signal of the lane. $r$ represents the red signal of the lane, which is a stop signal. $y$ represents the yellow signal of the lane, reminding the vehicle to slow down and give way, or stop and wait when through the intersection. Figure 4 shows the phases of the traffic light, for example, $p_1$ is green phase for all lanes of the north–south direction.

**Table 1** Traffic signal phase

| Phase | Describtion | Last time (s) |
|---|---|---|
| $p_1$ | $GGgGrrGGgGrr$ | 30 |
| $p_2$ | $GygGrrGygGrr$ | 5 |
| $p_3$ | $GrGGrrGrGGrr$ | 10 |
| $p_4$ | $GryGrrGryGrr$ | 5 |
| $p_5$ | $GrrGGgGrrGGg$ | 30 |
| $p_6$ | $GrrGygGrrGyg$ | 5 |
| $p_7$ | $GrrGrGGrrGrG$ | 10 |
| $p_8$ | $GrrGryGrrGry$ | 5 |

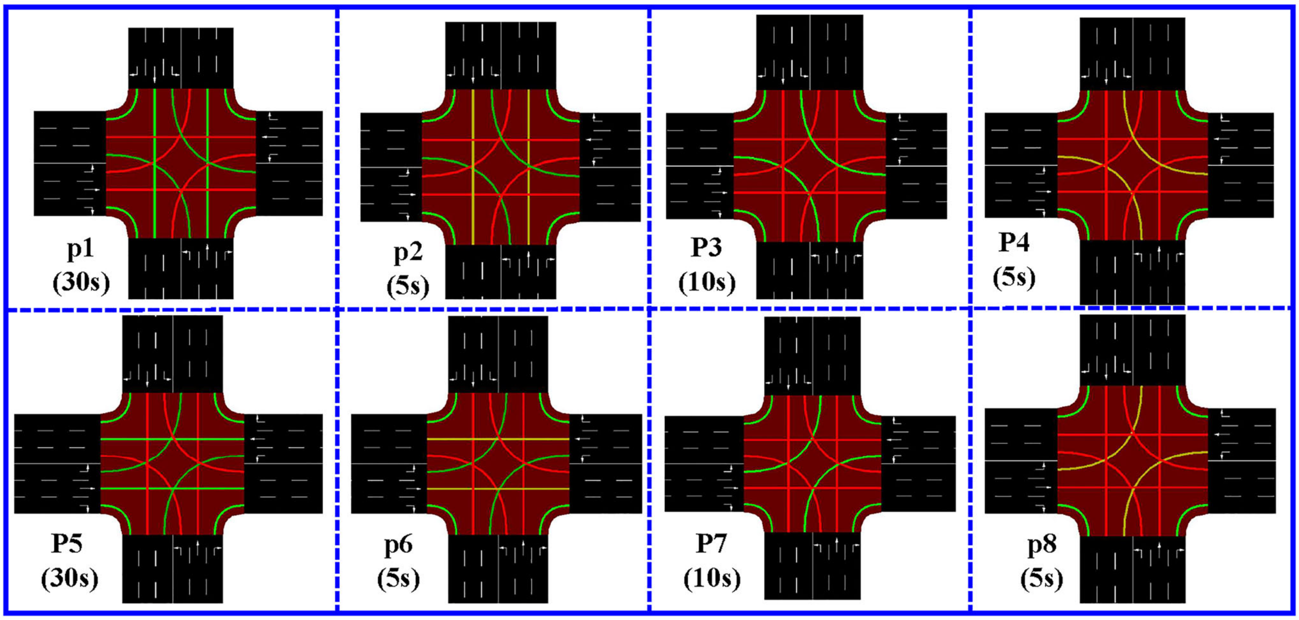


**Fig. 4** Traffic signal phase

## Reinforcement learning for urban traffic control

In this section, we define the state space $S$, action space $A$ and reward function $R$ of the agent in Pri-DDQN algorithm.

### State

The state space consists of vehicle position, speed, and current signal phase, which is denoted as $S$, $S = [D, V, p_i]$. Each lane is divided into the same cells. $D$ is the vehicle position matrix and represents whether a vehicle is located in the cell on each lane. $V$ is the vehicle speed matrix, which represents the standardized vehicle speed in the cells on each lane. $p_i$ is the phase of the signal light at intersection. The discrete traffic state encoding (DTSE) is shown in Fig. 5.

We define the state space that refine the traffic state from lane level to vehicle level. The position matrix value is Boolean type, and if there is a vehicle at the cell, it is 1, otherwise 0. We calculate $D_{i,j}$ by using Eq. 1,

$$D_{i,j} = \begin{cases} 1, & \text{there is a car;} \\ 0, & \text{otherwise,} \end{cases} \tag{1}$$

where $i$ is the lanes number, and $j$ is the number of the cell in the lane $i$, $1 \leqslant i \leqslant N$, $1 \leqslant j \leqslant N$.

We set the vehicle length plus the safe distance as a cell length, then each lane can be divided into $N$ cells. $N$ is calculated using Eq. 2,

$$N = \frac{l}{l_c + d_0}, \tag{2}$$

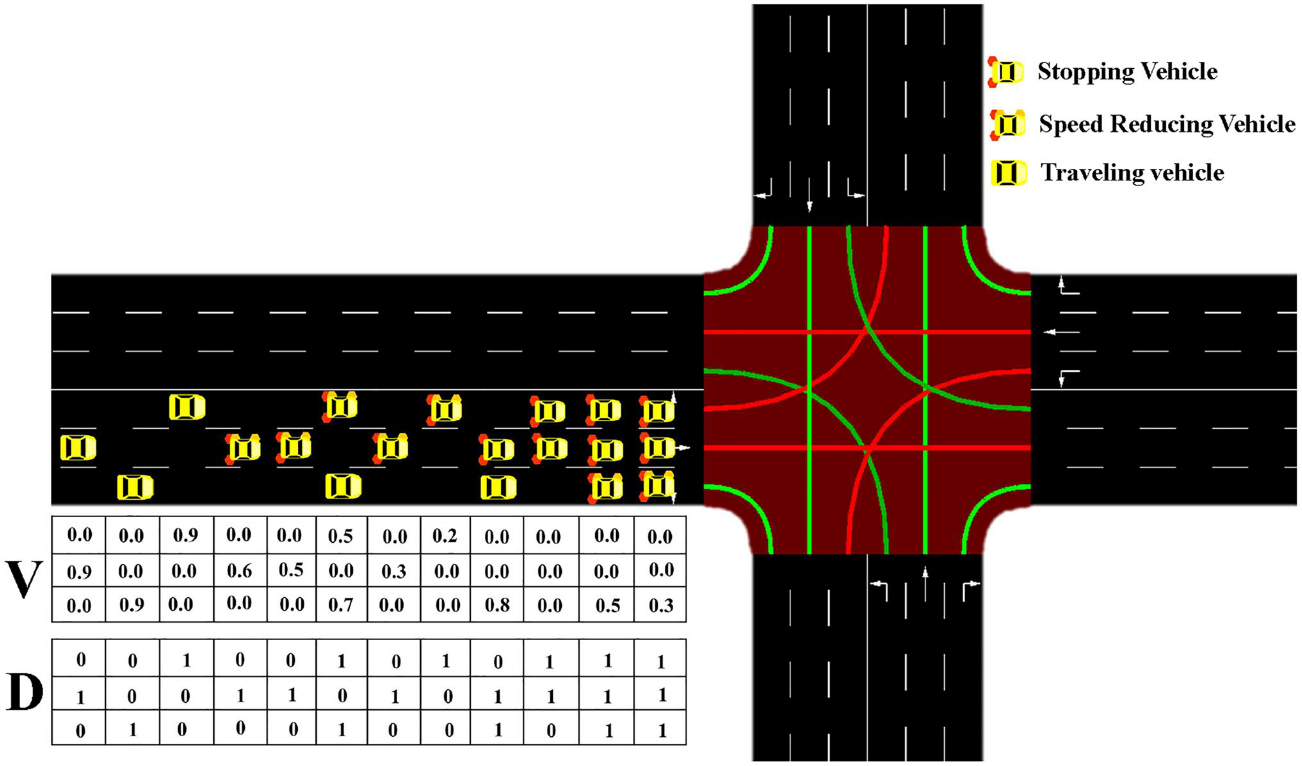
where $l$ is the road length, $l_c$ is the length of the vehicle, and $d_0$ is the safe distance between vehicles.

The matrix $V$ records the vehicle speed. For convenience of recording, the value is the ratio of the vehicle speed to the max speed of lane, that is, when the vehicle is traveling at maximum speed, $V_{i,j}$ is recorded as 1. When the vehicle is in the deceleration state and the speed is half of the maximum speed, $V_{i,j}$ is recorded as 0.5. We set $V_{i,j}$ to be the value calculated using Eq. 3,

$$V_{i,j} = \frac{v_k}{V_{max_i}}, \tag{3}$$

where $v_k$ is the speed of the $k$-th car, $k = 1, 2, 3, \cdots$, and $V_{max_i}$ is the maximum speed of lane $i$, $1 \leqslant i \leqslant 12$. Through this matrix, we can judge which vehicles are waiting for the red light.

Besides position and speed of vehicle, the state includes the phase $P$ of the traffic signal light, which is also recorded as a matrix. Each traffic signal control cycle contains eight phases, which are expressed as $P = \{p_1, p_2, p_3, \cdots, p_8\}$ and correspond to $\{0, 1, 2, 3, 4, 5, 6, 7\}$. $p_1$ and $p_5$ are the two core phases, which last for 31 s. In these two states, the signal is green for all lanes, and the priority of the straight lane is higher than that of the left turn lane.



| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| V | 0.0 | 0.0 | 0.9 | 0.0 | 0.0 | 0.5 | 0.0 | 0.2 | 0.0 | 0.0 | 0.0 | 0.0 |
| | 0.9 | 0.0 | 0.0 | 0.6 | 0.5 | 0.0 | 0.3 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| | 0.0 | 0.9 | 0.0 | 0.0 | 0.0 | 0.7 | 0.0 | 0.0 | 0.8 | 0.0 | 0.5 | 0.3 |

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| D | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 |
| | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 |
| | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 1 |

**Fig. 5** The discrete traffic state encoding (DTSE)

## Action

$A$ is the action space of the agent that has two elements, $A = \{a_1, a_2\}$. $a_1$ represents north–south pass through, and $a_2$ means east–west pass through. For example, $A = [1, 0]$ represents vehicles on north–south roads gain right of way. $a_1$ and $a_2$ correspond to core phases $p_1$ and phase $p_5$, respectively. We add the transition phase between $p_1$ and $p_5$ to prevent traffic accidents caused by directly changing phase. The changing process is shown in Table 2. If the decision of the agent is consistent with the traffic light phase, the signal light phase does not need to be changed. It will be make decision again after 15 s, which increases the green time of the current phase. If the agent's decision is inconsistent with the traffic light phase, it needs to complete the phase switch through the transition phase.

When the left turn vehicle meets a vehicle going straight, the former should politely give way to the latter first. $p_2$, $p_3$, and $p_4$ are the transition phases between the core phases $p_1$ and $p_5$, which meet the "green-yellow-red-green" changing sequence in real life. In $p_2$, the straight lane turns yellow, and the left turn lane is still green. In $p_3$, the straight lane turns red, and the left turn lane changes to priority green. These two states are designed to prevent vehicles in the straight lane from crossing the intersection while allowing left-turning vehicles that do not cross the intersection to pass through the intersection. In $p_4$, the left turn lane turns yellow, prompting the coming vehicles to slow down and warning that the traffic light will turn red. Finally, traffic light turns to phase $p_5$, vehicles in the east and west directions can begin to move. $p_6$, $p_7$, and $p_8$ are the transition phases between the core phases $p_5$ and $p_1$, and the changing process is the same as above.

## Reward

In order to encourage the agent to make reasonable decisions as much as possible and relieve the pressure of traffic congestion at the intersection, the environment will give the agent an immediate reward. We design a dynamic reward. After the agent makes decision, traffic lights start to switch phase in accordance with the decision traffic signal control strategy. After the agent performs action, the reward for this action is calculated.

We aim to reduce the waiting time and queue length time of vehicles to pass through the intersection, so we make waiting time and queue length as the reward of the agent. Waiting time is given by the sum of the times that vehicles are stopped. Queue length is calculated for each lane in an intersection. Thus, we calculate $R$ by using Eq. 4,

$$R = -(\omega \times T + Q), \tag{4}$$

**Table 2** Traffic signal phase changing

| $a_{t+1}$ | $a_t$ | $p_t$ | $p_{t+1}$ | Phase change |
|---|---|---|---|---|
| North–South | $a_1$ | $p_1$ | $p_1$ | Keep $p_1$ (15 s) |
| pass through ($a_1$) | $a_2$ | $p_5$ | $p_1$ | Change $p_6$ (5 s) → $p_7$ (10 s) → $p_8$ (5 s) → $p_1$ (30 s) |
| East–West | $a_2$ | $p_5$ | $p_5$ | Keep $p_5$ (15 s) |
| pass through ($a_2$) | $a_1$ | $p_1$ | $p_5$ | Change $p_2$ (5 s) → $p_3$ (10 s) → $p_4$ (5 s) → $p_5$ (30 s) |

where $T$ is the total waiting time of all vehicles to pass through the intersection, and $Q$ is the total queue length of vehicles at the intersection, $\omega$ is the weight of total waiting time to make a balance between $T$ and $Q$.

We set $T$ to be the value calculated by Eq. 5,

$$T = \sum_{k=1}^{K} t_k, \tag{5}$$

where $t_k$ is the delay of vehicle $k$ passing through the intersection, $K$ is the total number of vehicles.

We set $Q$ to be the value calculated by Eq. 6,

$$Q = \sum_{l=1}^{N} q_l, \tag{6}$$

where $q_l$ is the length of the queue waiting in $l$-th lane at the intersection, $N$ is the total number of lanes.

The aim of the agent is to reduce the waiting time and queue length time at the intersection. The smaller sum of the waiting time and queue length is, the less the traffic congestion will be, and the control effect of the agent is better. Therefore, we encourage the agent to explore action to maximize the reward in the next action decisions, and the action is positive regulation. On the contrary, the action is reverse regulation, and the road becomes congested. Therefore, the reward is set to the opposite number of the sum of the waiting time $T$ and queue length $Q$.

## Pri-DDQN for traffic signal control

### Signal control model based on Pri-DDQN

The state of traffic environment has a high spatial dimension and time variability. To overcome the dimensionality curse caused by state space explosion and the overestimation problem of the model, we design a traffic signal control model based on the improved DQN, as shown in Fig. 6.

We use CNN to extract the characteristics of the traffic environment to improve the expressive ability of the model. While a dynamic experience replay mechanism is established to enhance learning efficiency.

In Pri-DDQN model, we build the value network and the target network, which have the same network structure. The value network is used to calculate the action corresponding to the maximum Q value, and the target network is used to calculate the target Q value corresponding to the maximum action. The deep Q network of them is CNN, which is to extract the fine-grained features of the traffic environment, to enhance the expressive ability of the model. More importantly, the overestimation problem is eliminated by decoupling the action selection of the target Q-value and the calculation of the target Q-value. The Q network includes two convolutional layers and two fully connected layers, respectively. The output is the Q value of two actions.

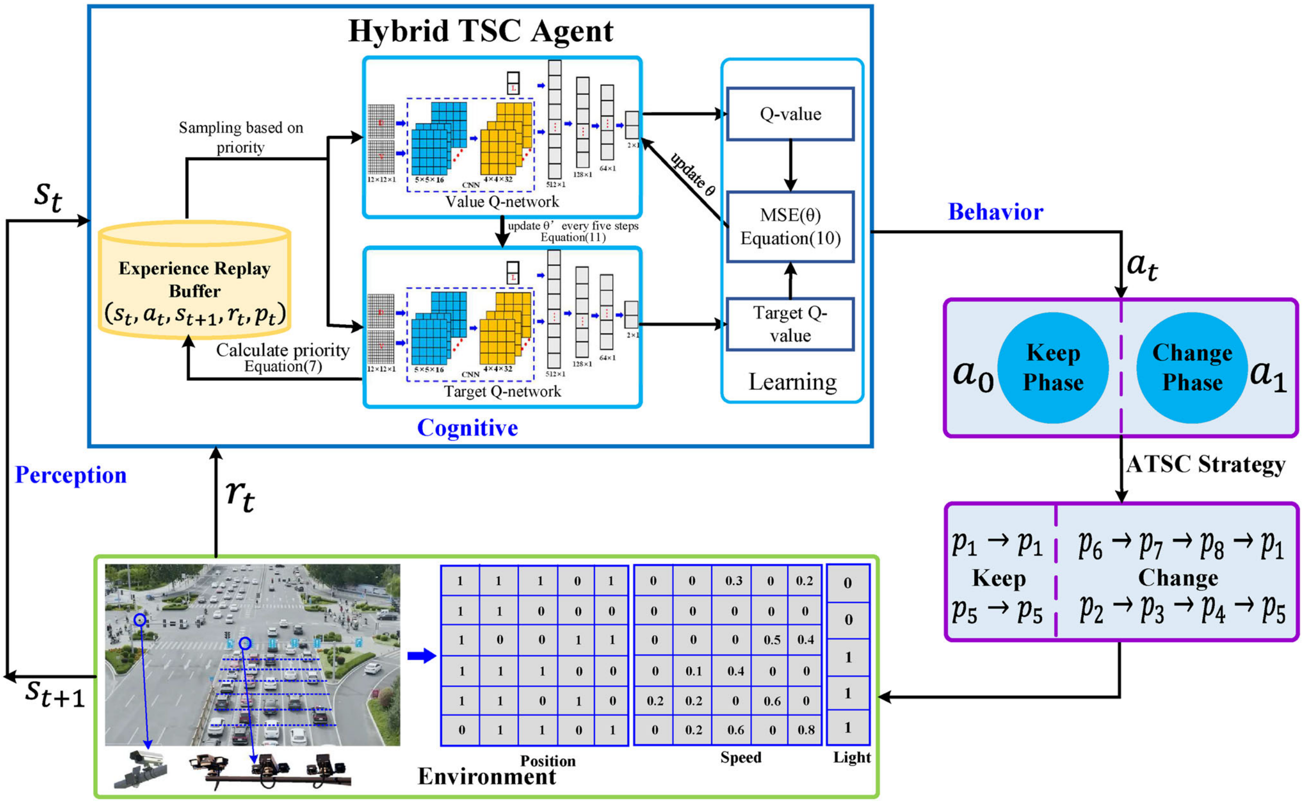
We use the greedy strategy $\epsilon\text{-}greedy$ [26] to select the agent's action. This strategy takes action $a_t$ according to Q-value with probability $(1-\epsilon)$ or random action with probability $\epsilon$. Depending on the decision action, traffic lights perform different signal control strategies.

For example, if the current phase is $a_1$ north–south pass through, and the agent decision is $a_2$ east–west pass through, the traffic signal light phases will be switch from $p_1$ to the target phase $p_5$. Considering the safety in practical application, three transitional phases are present between the two phases. $p_2$, $p_3$, and $p_4$ are the transition phases between the core phases $p_1$ and $p_5$. The traffic light phase conversion process can be expressed as $p_2(5\,s) \rightarrow p_3(10\,s) \rightarrow p_4(5\,s) \rightarrow p_5(30\,s)$. When the traffic light phase switches to $p_5$, vehicles traveling in east–west begin to pass. If the current phase is $a_1$ north–south pass through and the decision result is $a_1$ north–south pass through, we keep the $p_1$ phase that lasts 15 s to enhance the utilization rate of the effective green time, which is expressed as $p_1(30s) \rightarrow p_1(15s)$, and then make a decision again.

At the end of the execution cycle, the agent observes and records the state of the traffic environment, thereby learn to decide the action at the next cycle.

### Improved target network

To solve the problem of correlation and non-static distribution of samples, we introduce the priority-based experience playback mechanism. We set up an experience pool $E = \{e_1, e_2, \cdots, e_t, \cdots, e_T\}$ to store the transfer samples $e_t = (s_t, a_t, r_t, s_{t+1}, p_t)$, which is obtained from the interaction



|  |  |  |  |  |
|---|---|---|---|---|
| 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 |
| 1 | 1 | 1 | 0 | 0 |
| 1 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 |

Position

|  |  |  |  |  |
|---|---|---|---|---|
| 0 | 0 | 0.3 | 0 | 0.2 |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0.5 | 0.4 |
| 0 | 0.1 | 0.4 | 0 | 0 |
| 0.2 | 0.2 | 0 | 0.6 | 0 |
| 0 | 0.2 | 0.6 | 0 | 0.8 |

Speed

|  |
|---|
| 0 |
| 0 |
| 1 |
| 1 |
| 1 |

Light

**Fig. 6** Pri-DDQN model

between agent and environment in each time step. Agent selects some samples to train based on priority. $T$ is the total number of samples, $t$ is the number of the sample.

In order to eliminate the problem of overestimation in the DQN algorithm, we improve the estimation method of the Q-value of the target network. First, we find the action $a'$ with the maximum Q-value in the value network and calculate $a'$ by using Eq. 7,

$$a' = argmax_a Q(s_{t+1}, a; \theta), \tag{7}$$

where $s_{t+1}$ is the new state of environment after executing the action $a$, and $argmax_a Q(s_{t+1}, a; \theta)$ is the action with the maximum Q-value in the Q-network.

Then, we calculate the Q value of the target network according to the choose action $a'$, and set $TargetQ$ to be the value calculated by Eq. 8,

$$TargetQ = r_t + \gamma \times Q(s_{t+1}, a'; \theta'), \tag{8}$$

where $r_t$ is the real reward at $t$, $\gamma$ is the discount factor that indicates the effect of future actions on the current state. We decouple action selection and evaluation to eliminate overestimation.

Finally, we take the Q values of the value network and target network as true and predicted values, respectively. We use the gradient descent method to update network parameters, and the loss function is the mean square error (MSE). We calculate the loss function by using Eq. 9,

$$L(\theta) = E[(TargetQ - Q(s, a; \theta)^2], \tag{9}$$

where $TargetQ$ is the Q-value of the target network, and $Q(s, a; \theta)$ is the Q-value of the value network. We calculate MSE by using Eq. 10,

$$\begin{aligned} MSE(\theta) = \frac{l}{m}\sum_{i=1}^{m}((r_t + \gamma \times Q(s_{t+1}, a'; \theta')) \\ - Q(s, a; \theta))^2, \end{aligned} \tag{10}$$

where $\theta$ is the value network parameters, and $\theta'$ is target network parameters. These values are not the same.

We update the value network parameters in real-time to guarantee the stability of the Q-function, whereas the parameters of the target network are updated every five actions. We update the target network parameters by using Eq. 11:

$$\theta' = \omega \times \theta' + (1 - \omega) \times \theta, \tag{11}$$

where $\theta'$ is the old parameters of the target network, and $\theta$ is the real updated parameters of the value network.

### Improved decay $\varepsilon$-*greedy*

We improve the state space exploration based on the idea of decay $\varepsilon\text{-}greedy$ by first trying to use the power function dynamically changing the exploration rate. It speeds up the algorithm convergence, ensures a better convergence effect, and makes the decision more accuracy. We calculate $\varepsilon$ by using Eq. 12,

$$\varepsilon = m^n, \tag{12}$$

where $m$ dynamic change and is one of our optimization goals, which will be discussed in detail in the experiment. $n$ is the number of iterations.

### Priority-based experience replay mechanism

If the experience pool is full, some samples with the lowest priority is deleted, and the new sample with high priority is added to ensure that samples are diversity. We tend to select samples with a large contribution to speed up agent learning. It is known that the temporal difference(TD) error of the sample to sample will make the algorithm easier to converge. The sample with a large TD error has a greater effect on backpropagation. In the Q network, the TD error is the difference of the Q value between the target Q network and value Q network. In DDQN, the TD error is calculated by Eq. 13,

$$\begin{aligned}\delta_t &= r_{t+1} + \gamma \times Q(s_{t+1}, argmax_a Q(s_{t+1}, a; \theta_t); \theta_t^{'}) \\ &\quad - Q(s_t, a_t; \theta_t),\end{aligned} \tag{13}$$

where $\theta_t$ is the parameters of value network at step $t$, $\theta_t^{'}$ is the parameters of target network at step $t$. The larger the absolute value of the TD error, the greater the loss of the Q network during training. It indicates that the sample brings more information to the Q network, and the sample should have a high priority. Therefore we set the priority $p_t$ to be the value calculated by Eq. 14,

$$p_t = |\delta_t|, \tag{14}$$

If the TD error stored in the experience pool is not updated in time, it cannot accurately reflect the priority of the samples, because the Q network is updated in each iteration. So, we will update its priority after the sample is used. The update is calculated as Eq. 15:

$$\begin{aligned}\delta_{t+\tau} &= r_{t+1} + \gamma \\ &\quad \times Q(s_{t+1}, argmax_a Q(s_{t+1}, a; \theta_{t+\tau}); \theta_{t+\tau}^{'}) \\ &\quad - Q(s_t, a_t; \theta_{t+\tau}),\end{aligned} \tag{15}$$

where $t + \tau$ is the step of sample $t$ when it is sampled.

**Table 3** Traffic signal phase

| Parameter | Value |
|---|---|
| Road length | 250 m |
| Available route | Straight, turn right, turn left |
| Max apeed | 13.89m/s |
| Length of vehicles | 3 m |
| Min gap between vehicles | 1.5m |
| Episodes | 100 |
| MaxStep | 5400 |
| $\gamma$ | 0.9 |
| $\alpha$ | 0.001 |
| Memory size | 600 |
| Batch size | 32 |
| $\epsilon$ | $1 - \frac{episode}{episodes}$ |

### Algorithm pseudo code

The agent initializes the neural network, observes the initialized state $s$ as the input, and selects initialized action $a$ randomly. The output is the Q value of the two actions. We use the decay $\epsilon\text{-}greedy$ policy to choose the action. The agent controls the switching of the traffic light according to the action $a$, calculates the reward $r$, and observes the new state $s'$ after executing action $a$. The algorithm pseudo code is presented in Algorithm 1.

## Experiment analysis

### Experiment set

The experimental simulation environment is built using the Netedit 1.7.0 in SUMO, that is shown in Fig. 3, and the phases of traffic light are shown in Fig. 4.

We use a public dataset in our experiment released by the University of Pennsylvania and Shanghai Jiao Tong University [1, 21, 27] and record vehicle driving information at an intersection in Hangzhou. The main information is vehicles, road, the time of cars entry, and exit road, and speed, which are all processed in accordance with the required simulation format. To evaluate the performance of the algorithm accurately, we select traffic congestion scenarios, which are 1-hour data, because of the large vehicle and change in traffic flow with time. The main experimental parameters are shown in Table 3, and the experiment environment is shown in Fig. 3.

**Algorithm 1** Pri-DDQN Algorithm

```
Input: Replay memory size, batch size, number of episodes, and number of time step, maxSteps
Initialize: Q-network, Target Q-network, Memory, Agent
Output: $Q$ value
for $e = 1, 2, ..., episodes$ do
    Initialize state $s_0$
    for $step = 1, 2, ..., maxSteps$ do
        Take action $a'$ with probability $(1 - \varepsilon)$ or random action with probability $\varepsilon$, $\varepsilon \leftarrow 0.8^{step}$;
        Get reward $r_t$ and observe next state $s_{t+1}$;
        Calculate priority $p_t$ of $(s_t, a_t, s_{t+1}, done)$ by using Eq( 13) and Eq( 14);
        Save $(s_t, a_t, r_t, s_{t+1}, p_t)$ to Memory;
        Agent Replay:
         if $numberofmemory < batchsize$ then
             return
         end if
         mini-batch ← Sample in the experience pool based on priority;
         for $(s_t, a_t, r_t, s_{t+1}, p_t)$ in mini-batch do
             $y_t \leftarrow \begin{cases} r_t, \text{if done} = \text{True} \\ r_t + \\ \gamma \times Q(s_{t+1}, argmax_a Q(s_{t+1}, a; \theta_t); \theta_t^{'}), \\ \text{otherwise} \end{cases}$
         end for
         Perform policy gradient using $y_t$ for updating $\theta$;
         Update target Q-network perform every five actions:
             $\theta' \leftarrow (1 - \sigma) \times \theta' + \sigma \times \theta$
         Save waitingtime, queuelength, reward of every episodes;
         Delete last memory;
    end for
end for
```

## Evaluation of metrics

- The cumulative waiting time $T$ is the total amount of time, it takes for all vehicles to pass through the intersection. The larger the $T$, the more congested the intersection.
- The queue length $Q$ is the sum of the queue lengths in each entrance lane at the intersection when the traffic phase is red. The larger the $Q$, the more congested the intersection.

## Experiment and result analysis

### State space exploring improved

To verify the effectiveness of Pri-DDQN, we conduct experiments on real data sets. In the experiment, the agent is trained 100 episodes in the simulation environment, and each episode lasts 5400 steps. Each step is 1 s in real life, that is, each training lasts 5400 s. We explore the action selection strategy, improve the state space detection method, and explore the different values of $\varepsilon$. The first strategy is the fixed value method, and we set $\varepsilon$ to be 0.1, as shown in Eq. 16,

$$\varepsilon = 0.1. \tag{16}$$

The second strategy is the ratio method. We set $\varepsilon$ to be the value calculated by Eq. 17,

$$\varepsilon = 1 - \frac{n}{maxStep}, \tag{17}$$

where $n$ is the number of simulations, and $maxStep$ is the total number of simulations. The third strategy is the power function, and we set $\varepsilon$ to be the value calculated by Eq. 18,

$$\varepsilon = 0.8^n. \tag{18}$$

In this section, these three methods are explored, and experimental results are shown in Fig. 7. The x-axis is the training time, and the y-axis is the cumulative queue length (CQL) of each training, that is, the sum of the queue length values of each training.

Figure 7 shows the following findings. On the one hand, the CQL of each episode decreases with training time because in the early stage of simulation. The agent has no prior knowledge, and is not familiar with the environment, and has strong randomness in action selection, whose control effect is not ideal. As the number of simulations increases, the agent gradually accumulates experience. Through experience replay and self-learning, high-reward actions are selected on the

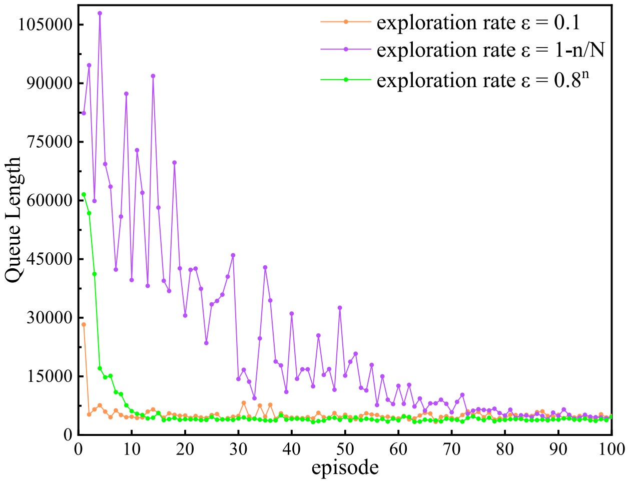


**Fig. 7** Queue length with different exploration rate $\epsilon$

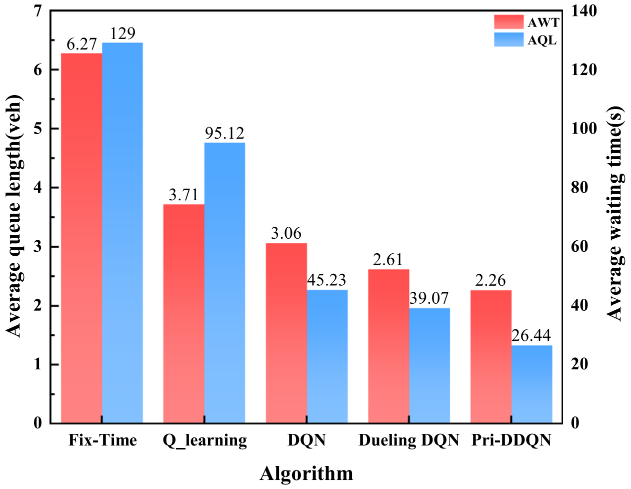


**Fig. 8** AQL and AWT with different algorithm

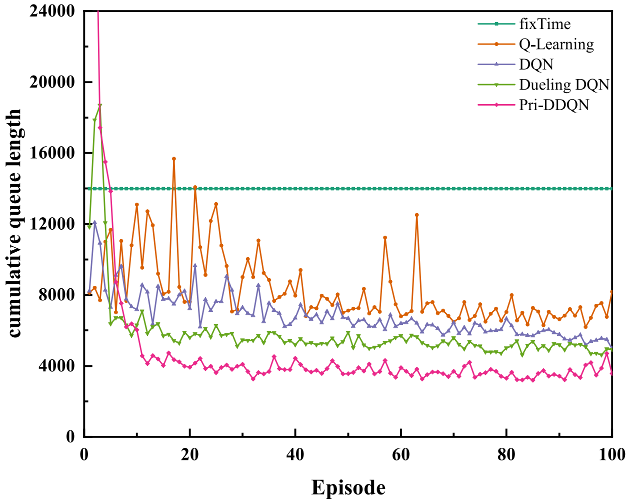


**Fig. 9** The CQL of different algorithm

basis of the environment state to reduce the length of vehicle queues and waiting times and finally converge to stable values.

On the other hand, different $\epsilon$ values have different control effects. When the exploration rate is dynamically valued by power function, the algorithm converges the fastest, and the vehicle queue length is shortest. The CQL is reduced to 3951 s, which is optimized by 21.46% and 78.26% over the fixed value and ratio methods, respectively.

### Comparison of methods

Several state-of-the-art approaches are chosen as baseline methods to validate the performance of the Pri-DDQN algorithm. There are mainly two categories: transportation approaches and RL methods.

- Transportation methods: Fixed-time control is a transportation method that uses a predefined plan for traffic light control.
- RL methods: Q-learning, DQN and Dueling DQN are reinforcement learning methods. The agent decides actions based on the state of the environment and executes appropriate signal control policies.

We compare the Pri-DDQN with the baseline method on the real dataset and take the average queue length (AQL) and average waiting time (AWT) as evaluating indicators. The AQL and AWT of each algorithm are shown in Fig. 8. The x-axis is the algorithm, the left y-axis is the average queue length of the vehicles for each algorithm, and the right y-axis is the average waiting time. Figure 8 gives the AQL and AWT calculated from 100 simulations of 2230 vehicles that enter the intersection and leave the intersection. We can see that, Pri-DDQN outperforms the other four methods with the shortest AQL and shortest AWT. Compared with the fixed-time, Q-Learning, DQN, and Dueling DQN, the AQL was reduced by 63.95%, 39.08%, 26.14%and 13.41%, and the AWT was reduced by 79.5%, 72.2%, 41.54%, and 32.33% respectively. It also shows that the shorter the queue length, the shorter the waiting time of the vehicle.

In terms of convergence and stability, we take the cumulative queue length (CQL) and cumulative waiting time (CWT) of vehicles passing through the intersection as an example. The CQL and CWT for each episode vary with the number of training, as shown in Figs. 9 and 10. The x-axis is the training time, and the y-axis is the CQL and CWT of each training with different algorithms, that is, the sum of the queue lengths and the sum of the waiting time of each training, respectively. The CWT and CWQ of the fixed-time is always a fixed and highest value, while the CQL and CWT of Q-Learning, DQN, Dueling DQN and Pri-DDQN in each

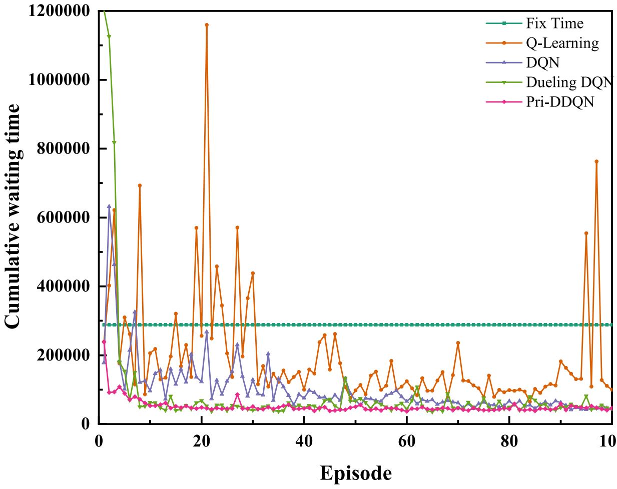


**Fig. 10** The CWT of different algorithm

episode are all decreasing with training time, and Pri-DDQN has the best result.

In addition, whether it is the CQL or CWT, the convergence speed of Pri-DDQN is significantly faster than that of the baseline method, and the stability is the best. The Q-learning determines the agent's action by looking up the Q-table. Because the number of samples increases causes feature curse of dimensionality, the capacity of the Q-table also increases. Thus, the efficiency and stability of the Q-learning algorithm are poor. The DQN algorithm combines neural networks and experience replay. Although it solves the problem of feature dimensionality curse, it ignores the correlation of samples, and its state space exploration method is fixed, which makes it easy to fall into local optimal solutions or overestimation. Dueling DQN is an improved method based on DQN. It decomposes the Q value function into a state value function and an advantage function to better estimate the contribution of different actions to the state and improve learning efficiency. However, it only uses uniform sampling and batch updates, resulting in some low volume but high-value experiences not being efficiently utilized. Pri-DDQN combines the advantages of DQN and Dueling DQN, and uses the power function to dynamically change the agent's exploration rate. It also sorts the experience samples and prioritizes samples with high learning value. Therefore, it has the best convergence speed and experimental effect, effectively alleviating the traffic congestion during peak hours and is more adaptable to real-time changing traffic flows.

## Conclusion

In order to respond to the randomly changing traffic flow and realize the adaptive traffic signal control, we propose a new single-intersection traffic signal control method (Pri-DDQN) based on reinforcement learning. First, we model the traffic environment as a reinforcement learning environment and adopt an improved DTSE method to characterize the randomly changing traffic state. Second, we improved the Double DQN network structure, added a convolutional neural network to extract traffic state features to enhance the expressive ability of the model, and updated the target network parameters asynchronously. In order to accelerate the convergence of the algorithm, we utilized a power function to dynamically change the exploration rate. Finally, We prioritize samples based on reward and established priority-based dynamic experience replay mechanism to increase the sampling rate of important samples and the learning efficiency of the Agent. We validate the effectiveness of the algorithm based on real-world traffic data. The results show that Pri-DDQN achieves better performance, compared to the best baseline, it reduces the average queue length is reduced by 13.41%, and the average waiting time by 32.33% at the intersection.

In future, we will continue to improve the algorithm's performance, such as further optimizing the priority of experience playback in Pri-DDQN to ensure that the diversity and completeness of samples. So that the agent can make efficient decisions and response to rapid changes in the traffic environment. Furthermore, in addition to making single intersection traffic signal control, we will also study the distributed traffic signal adaption control of multi-intersection to optimize area traffic signal control.

**Acknowledgements** This work was supported in part by the National Natural Science Foundation of China under Grant 62372163, Natural Science Foundation of Chongqin under Grant cstc2021jcyjmsxmX0460, The science and technology innovation Program of Hunan Province under Grant 2024RC1033.

**Data availability** The data that support the findings of this study are openly published by the University of Pennsylvania and Shanghai Jiao Tong University at https://traffic-signal-control.github.io/#open-datasets.

## References

1. Wei H, Zheng G, Gayah V, Li Z (2019) A survey on traffic signal control methods. CoRR. **abs/1904.08117**
2. Haydari A, Yilmaz Y (2022) Deep reinforcement learning for intelligent transportation systems: a survey. IEEE Trans Intell Transport Syst 23(1):11–32. https://doi.org/10.1109/TITS.2020.3008612
3. Yong Q (1989) Urban traffic control. China Communications Press, Beijing
4. Zhang J, Deng S (2005) Application study and remedy on theory and method by webster calculation in signal timings. Hebei Jiaotong Science and Technology
5. Spall JC, Chin DC (1997) Traffic-responsive signal timing for system-wide traffic control. In: Proceedings of the 1997 American control conference (Cat. No.97CH36041), vol 4, pp 2462–2463. https://doi.org/10.1109/ACC.1997.609205
6. Murat YS, Gedizlioglu E (2005) A fuzzy logic multi-phased signal control model for isolated junctions. Transport Res Part C 13(1):19–36
7. Liu J, Zuo X (2020) Research on fuzzy control and optimization for traffic lights at single intersection. J Syst Simul **32**(12)
8. Bi Y, Lu X, Sun Z, Srinivasan D, Sun Z (2018) Optimal type-2 fuzzy system for arterial traffic signal control. IEEE Trans Intell Transport Syst 19(9):3009–3027. https://doi.org/10.1109/TITS.2017.2762085
9. Li X, Guo Z, Dai X, Liu Y (2020) Deep imitation learning for traffic signal control and operations based on graph convolutional neural networks. In: 2020 IEEE 23rd international conference on intelligent transportation systems (ITSC), pp 1–6. https://doi.org/10.1109/ITSC45102.2020.9294215
10. Sanchez-Medina JJ, Galan-Moreno JM, Rubio-Royo E (2010) Traffic signal optimization in “la almozara” district in saragossa under congestion conditions, using genetic algorithms, traffic microsimulation, and cluster computing. IEEE Trans Intell Transport Syst 11(1):132–141. https://doi.org/10.1109/TITS.2009.2034383
11. Watkins CJCH (1989) Learning from delayed rewards
12. Yu X-H, Recker WW (2006) Stochastic adaptive control model for traffic signal systems. Transport Res Part C Emerg Technol 14(4):263–282
13. Gao J, Shen Y, Liu J, Ito M, Shiratori N (2017) Adaptive traffic signal control: deep reinforcement learning algorithm with experience replay and target network. CoRR
14. Sun H, Chen C, Liu Q, Zhao J (2020) Traffic signal control method based on deep reinforcement learning. Comput Sci 47(2):169. https://doi.org/10.11896/jsjkx.190600154
15. Liang X, Du X, Wang G, Zhu H (2019) A deep reinforcement learning network for traffic light cycle control. IEEE Trans Veh Technol 68(2):1243–1253. https://doi.org/10.1109/TVT.2018.2890726
16. Long S, Wang Z, Liu H (2020) Based on deep reinforcement learning optimization of traffic signal control at single intersection. Ind Control Comput 33(10):16–22. https://doi.org/10.3969/j.issn.1001-182X.2020.10.006
17. Li L, Lv Y, Wang F (2016) Traffic signal timing via deep reinforcement learning. IEEE/CAA J Autom Sin 3(3):247–254. https://doi.org/10.1109/JAS.2016.7508798
18. Lee J, Chung J, Sohn K (2020) Reinforcement learning for joint control of traffic signals in a transportation network. IEEE Trans Veh Technol 69(2):1375–1387. https://doi.org/10.1109/TVT.2019.2962514
19. Hua W, Chen C, Zheng G, Wu K, Gayah V, Xu K, Li Z (2019) Presslight: learning max pressure control to coordinate traffic signals in arterial network. In: Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery and amp; data mining. KDD ’19. Association for Computing Machinery, pp 1290–1298
20. Liang X, Du X, Wang G, Han Z (2019) A deep reinforcement learning network for traffic light cycle control. IEEE Trans Veh Technol 68(2):1243–1253. https://doi.org/10.1109/TVT.2018.2890726
21. Zheng G, Xiong Y, Zang X, Feng J, Wei H, Huichu Z, Li Y, Xu K, Li Z (2019) Learning phase competition for traffic signal control, pp 1963–1972. https://doi.org/10.1145/3357384.3357900
22. Zang X, Yao H, Zheng G, Xu N, Xu K, Li Z (2020) Metalight: value-based meta-reinforcement learning for traffic signal control. Proc AAAI Conf Artif Intell 34:1153–1160. https://doi.org/10.1609/aaai.v34i01.5467
23. Hua YBJ, Wang X (2023) Multi-agent deep reinforcement learning-based urban traffic signal management. Oper Res Trans 27(02):49–62
24. Xu D, Lei Zhou DWJD, Wei C (2022) Overview of reinforcement learning-based urban traffic signal control. J Transport Eng Inf 20(01):15–30. https://doi.org/10.19961/j.cnki.1672-4747.2021.04.017
25. Eom M, Kim B-I (2020) The traffic signal control problem for intersections: a review. Eur Transport Res Rev 12:1–20
26. Edwards A, Pottenger WM (2011) Higher order q-learning. In: 2011 IEEE symposium on adaptive dynamic programming and reinforcement learning (ADPRL), pp 128–134. https://doi.org/10.1109/ADPRL.2011.5967385
27. Wei H, Xu N, Zhang H, Zheng G, Zang X, Chen C, Zhang W, Zhu Y, Xu K, Li Z (2019) Colight: learning network-level cooperation for traffic signal control. In: Proceedings of the 28th ACM international conference on information and knowledge management. CIKM ’19