Received 22 January 2024, accepted 1 March 2024, date of publication 21 March 2024, date of current version 28 March 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3380454

RESEARCH ARTICLE

An Enhanced Dueling Double Deep Q-Network With Convolutional Block Attention Module for Traffic Signal Optimization in Deep Reinforcement Learning

PENG WANG[1] AND WENLONG NI[1,2], (Member, IEEE)
[1]School of Digital Industry, Jiangxi Normal University, Shangrao 334000, China
[2]School of Computer Information Engineering, Jiangxi Normal University, Nanchang 330022, China

Corresponding author: Wenlong Ni (wni@jxnu.edu.cn)

This work was supported in part by JiangXi Education Department under Grant GJJ191688.

ABSTRACT Many studies on the application of deep reinforcement learning (DRL) in the field of traffic signal control do not fully consider the influence of vehicles approaching the intersection on traffic flow. In this paper, the convolutional block attention module (CBAM) is incorporated on the basis of the Dueling Double Deep Q Network (D3QN) method to improve the sensitivity of the model to the traffic situation, which can help the model to focus more on the distribution and dynamics of vehicles near intersections. To further improve the model performance, this paper introduces the traffic light phase variable time interval based on the original D3QN method, which helps the model to take into account the traffic requirements in all directions of the intersection. In addition, Double Deep Q Network (Double DQN) and Dueling Deep Q Network (Dueling DQN) technologies are used to further improve the performance of the model. The simulation experiments using the urban traffic simulator SUMO show that the proposed method has significant advantages over D3QN, Maximum Pressure algorithm and Fixed Timing Strategy for key indicators such as mean vehicle delay time, mean queue length and average number of stops. This shows that the method proposed in this paper has great potential in practical traffic signal control applications.

INDEX TERMS Attention mechanism, deep reinforcement learning, deep Q network, traffic signal control.

I. INTRODUCTION

In the realm of urban traffic management, the continuous variability and complexity of traffic flow present an evolving challenge [1]. Despite the historical utility of fixed-time traffic signal control techniques, they have exhibited evident limitations in addressing the increasingly intricate and changeable urban traffic conditions of today. During different time intervals, traffic peak periods, and occurrences of special events, these conventional methods' inflexible scheduling strategies often struggle to adapt, resulting in a series of issues within the traffic system, including congestion, delays, fuel wastage, and environmental pollution [2]. To transcend the constraints of traditional approaches and better address the variability of traffic flow, reinforcement learning (RL) [3] technology has garnered increasing attention in recent years. RL technology, using learning and optimizing interactive decision processes, imbues traffic signal control with heightened intelligence and adaptability. In comparison to conventional fixed-time schemes, RL methods possess greater flexibility, permitting real-time adjustments in signal timing based on the environmental state, thereby achieving more optimized traffic flow control across diverse timeframes and traffic scenarios. Nevertheless, despite these merits, as the complexity of traffic conditions continues to escalate, table-based traditional Q-Learning algorithms manifest certain limitations when tackling high-dimensional state problems and continuous action scenarios.

In response to these challenges, the amalgamation of deep learning [4] and RL has emerged as a prominently

The associate editor coordinating the review of this manuscript and approving it for publication was Binit Lukose.

explored research domain in recent years, known as deep reinforcement learning (DRL) [5]. DRL employs deep neural networks to approximate state-action value functions, allowing systems to extract valuable features from raw sensory data, thereby surmounting the constraints exhibited by traditional Q-Learning algorithms in high-dimensional state and continuous action contexts. A range of widely applied deep learning architectures, including traditional fully connected neural networks (FCLN) [6], convolutional neural networks (CNN) [7], and stacked autoencoders (SAE) [8], have demonstrated robust adaptability and performance in various problem domains and scenarios.

Despite the substantial potential of DRL techniques in the field of traffic signal control, existing research frequently overlooks a pivotal factor: the influence of vehicles in proximity to intersections on traffic flow. These vehicles often wield greater weight and influence within the traffic stream, owing to their proximity to intersections, heightened sensitivity to signal variations, and substantial impact on overall traffic flow resulting from their behaviors. Consequently, this study endeavors to introduce a Convolutional Block Attention Module (CBAM) atop the Dueling Double Deep Q Network (D3QN) framework, situated on the foundation of DRL, to more accurately capture and underscore the influence of these proximate intersection vehicles on traffic flow.

The contributions of this work can be summarized as follows:

1) To enhance the model's focus on critical states within the traffic environment and suppress noise or irrelevant features distant from intersections, we introduce CBAM into the D3QN algorithm. Experimental results demonstrate performance improvements.
2) The introduction of variable timing strategies enables the agent to flexibly select signal timing schemes that adapt to the current traffic conditions. This enhancement adds a layer of dynamism and adaptability to our research, contributing to more effective congestion management and improved traffic flow efficiency.
3) The neural network architecture adopts Dueling Deep Q Network (Dueling DQN) [9], and the algorithm optimization employs the Double Deep Q Network (Double DQN) [10] approach to update weight parameters. This combination enhances performance and stability in RL tasks.

II. LITERATURE REVIEW

The issue of traffic signal control has always attracted much attention, and methods for optimizing signal control have emerged in endlessly. Typical value-based RL algorithms include Q-learning and SARSA. Thorpe and Anderson [11] analyzed the performance of the SARSA under three different representations of the current traffic status. The results showed that the representation of variable length partitioning was better than the equal length partitioning, vehicle count, and fixed duration strategies. Abdulhai et al. [12] conducted tests using three different traffic profiles to evaluate the performance of Q-Learning. Richter et al. [13] proposed a stochastic ascending policy gradient process based on the Actor-Critic algorithm to solve distributed road traffic problems. This method is essential research in traffic management, aiming to optimize road traffic flow, reduce traffic congestion, and improve traffic efficiency.

Traditional RL methods are generally more suitable for situations with more superficial state characteristics. When the dimension of the state space increases, these traditional methods face a severe challenge; that is, the dimension of the Q-value table or value function increases dramatically, which is often called the "curse of dimensionality." Given this problem, the latest research trend is to combine deep learning with RL and use neural network methods to extract abstract features of high-dimensional data automatically. This approach allows us to handle complex state spaces more efficiently and avoid the curse of dimensionality. This fusion approach is often called DRL.

Many current studies are based on DRL. Tan et al. [14] designed an adaptive traffic signal control framework based on DRL and improved the reward function. Experimental results show that this framework significantly improves traffic performance compared to baseline models, including predefined signal control and fully actuated signal control. Wang et al. [15] proposed a traffic light timing optimization method EP-D3QN based on D3QN, Max Pressure, and self-organizing traffic light (SOTL). Experimental results show that compared with four benchmark models, EP-D3QN has superior performance in both low traffic flow and large traffic flow scenarios. Wei et al. [16] proposed a more effective DRL model for traffic light control and tested it on a large-scale actual traffic data set obtained from surveillance cameras. The implementation results proved the effectiveness of the algorithm. Liu and Ding [17] introduced a traffic signal control (TSC) method based on DRL at isolated intersections. The proposed algorithm was compared with both fixed-time and adaptive TSC under various traffic conditions. The results show that the DRL-based strategy exhibits optimal results in both average delay time and pollutant emission environmental indicators.

III. SYSTEM DESIGN

In this section, We will model the traffic signal problem as a Markov Decision Process (MDP) to effectively address this issue. Let's progressively define the state space, action space, reward function, and action selection policy.

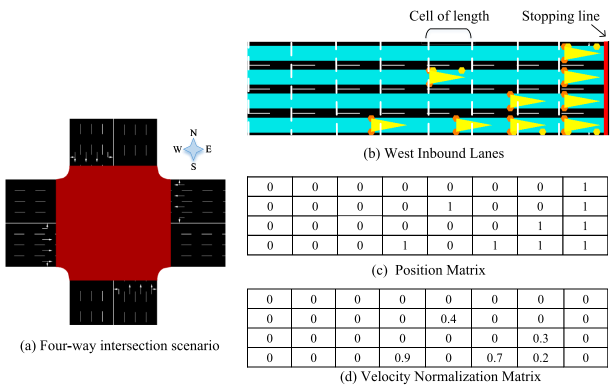
A. STATE DEFINITION

In RL, the state space constitutes a collection describing possible environmental conditions, specifically the various states of a traffic intersection.

As illustrated in Figure 1 (a), this intersection is a four-way crossroads, encompassing the directions of east, west, south, and north. Each approach road in these directions accommodates traffic flows from five different directions.

Notably, the leftmost lane permits exclusively left turns, the two middle lanes solely allow straight-ahead movement, while the rightmost lane permits both straight-ahead and right-turn movements.

The state comprises three primary components: vehicle positions, vehicle velocities, and traffic signal states. This study employs Discrete Traffic State Encoding (DTSE) [18] to represent traffic state information. All incoming lanes within the intersection are partitioned into equally-sized cells to denote vehicle information, with cell size equal to the vehicle length plus the minimum distance between adjacent vehicles. This arrangement ensures that each cell can accommodate at most one vehicle. Figure 1 (b) provides an illustrative division of the westbound approach lane. As depicted in Figure 1 (c), a cell is assigned a value of 1 if a vehicle occupies it; otherwise, it is set to 0. Furthermore, Figure 1 (d) demonstrates the representation of vehicle velocities as normalized values. The state dimensions are denoted as (C, H, W), with C representing the position matrix and normalized velocity matrix, H denoting the number of approach lanes, and W signifying the number of divided cells. Traffic signal states are represented using a one-hot encoding scheme, utilizing a vector containing eight elements, each element having a binary value of 0 or 1. If the currently selected action phase corresponds to the first phase, the state representation is [1, 0, 0, 0, 0, 0, 0, 0].



0	0	0	0	0	0	0	1
0	0	0	0	1	0	0	1
0	0	0	0	0	0	1	1
0	0	0	1	0	1	1	1

(c) Position Matrix

0	0	0	0	0	0	0	0
0	0	0	0	0.4	0	0	0
0	0	0	0	0	0	0.3	0
0	0	0	0.9	0	0.7	0.2	0

(d) Velocity Normalization Matrix

FIGURE 1. Intersection scene and status representation.

B. ACTION SPACE

In DRL, upon perceiving the current environmental state, an agent utilizes computational tools such as neural networks to assess the value of all potential actions commonly expressed as Q-values. These feasible actions collectively form what is known as the action space. As depicted in Figure 2, this study encompasses a total of eight non-conflicting phases, which are 1) the first phase (go straight and turn right in the north-south direction); 2) the second phase (turn left north-south); 3) the third phase (go straight and turn right in east-west direction); 4) the fourth phase (turn left east-west); 5) the fifth phase (south direction open to traffic); 6) the sixth phase (east direction open to traffic); 7) the seventh phase (north direction open to traffic); 8) the eighth phase (west direction open to traffic). Furthermore, this study considers traffic signal phases with variable time intervals, including 5 seconds, 10 seconds, and 15 seconds. Each traffic signal phase offers three alternative time interval possibilities. Consequently, the agent has a total of 24 actions at its disposal.

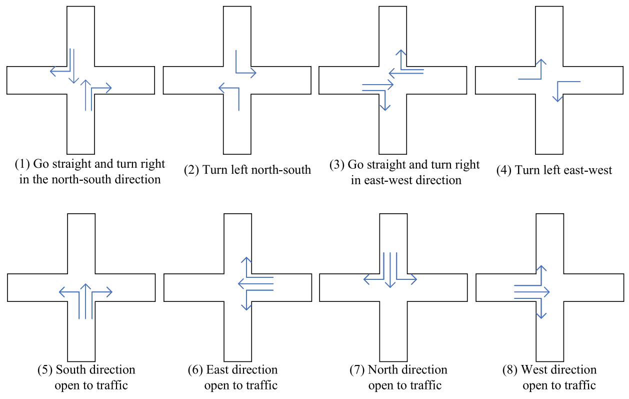

(1) Go straight and turn right in the north-south direction
(2) Turn left north-south
(3) Go straight and turn right in east-west direction
(4) Turn left east-west
(5) South direction open to traffic
(6) East direction open to traffic
(7) North direction open to traffic
(8) West direction open to traffic

FIGURE 2. Traffic light phase diagram.

C. REWARD FUNCTION

In RL, the primary objective of an agent typically revolves around the maximization of cumulative rewards. The reward function defines the immediate rewards an agent receives for taking different actions in various states. In this paper, we will employ a multi-metric coefficient-weighted approach to define the reward function, facilitating the optimization of multiple traffic metrics. This approach assists us in balancing the significance of different traffic metrics in RL, thereby guiding the agent's decision-making more effectively. This paper will simultaneously optimize three metrics: vehicle delay time, queue length, and the number of stopped vehicles.

Let w_t represent the sum of accumulated delay time for all inbound lanes at the intersection at decision point t, while w_{t-1} denotes the sum of accumulated delay time for all inbound lanes at decision point $t-1$. Let r_w signify the difference in accumulated delay time between adjacent decision points.

$$r_w = w_{t-1} - w_t \tag{1}$$

Let q_t denote the sum of vehicle queue lengths for all inbound lanes at the intersection at decision point t. Here, L represents the number of inbound lanes, and queue_l signifies the queue length for the l-th inbound lane.

$$q_t = \sum_{l=1}^{L} \text{queue}_l \tag{2}$$

Let h_t represent the sum of stopped vehicles for all inbound lanes at the intersection at decision point t. In this context, halt_l signifies the number of stopped vehicles in the l-th inbound lane.

$$h_t = \sum_{l=1}^{L} \text{halt}_l \tag{3}$$

Considering these factors and their corresponding weight coefficients w_1, w_2, and w_3, we derive the final reward function as shown in the equation (4). It is important to note that the selection of these coefficients, in this paper, is based on our discretion and the specific requirements of our experimental setup, these coefficients are set to 0.5, -1, and -1, respectively. When the agent takes action and observes the decreasing values of w_t, q_t, and h_t consecutively, this indicates that the environment provides positive feedback to the agent. Consequently, the agent continues to optimize its actions, moving toward the goal of maximizing cumulative rewards until convergence is achieved.

$$x = w_1 \times r_w + w_2 \times q_t + w_3 \times h_t \tag{4}$$

D. ACTION SELECTION POLICY

The agent utilizes an epsilon-greedy strategy for action selection. This strategy provides a balanced approach between exploration and exploitation. Its fundamental concept is that, when choosing an action each time, there is a probability of $1 - \varepsilon$ to select the action estimated to be the best in the current state (the action with the highest Q-value), and a probability of ε to randomly select an action.

In the early stages of training, we can set a relatively high value for ε to encourage exploration by the agent. This means that the agent has a higher probability of choosing random actions to have the opportunity to try different strategies, explore unknown areas in the environment, and uncover potential rewards. During this phase, the agent can accumulate more information about the environment.

As training progresses and the agent's understanding of the environment increases, we can gradually reduce the value of ε. This will lead to the agent being more inclined to choose actions based on existing experience and Q-values to maximize rewards. This strategy allows the agent to make full use of existing knowledge while maintaining a certain level of exploratory behavior to discover potentially better strategies.

In this paper, we update the value of ε in an exponentially decaying manner, where $\varepsilon_{\text{decay}}$ represents the decay rate. The equation is as follows:

$$\varepsilon = \max\left(0.01,\ \varepsilon_{\text{decay}} * \varepsilon\right) \tag{5}$$

IV. ALGORITHM MODEL

In this section, we will present a novel algorithmic model aimed at enhancing performance. This encompasses the introduced CBAM module, network Structure, the training process, and algorithmic description.

A. CBAM MODEL

CBAM is an attention mechanism introduced in convolutional neural networks, which combines channel attention and spatial attention to enhance the significance and adaptability of features.

1) CHANNEL ATTENTION MODULE

First, we integrate channel information within the feature map by applying average pooling and max pooling operations. Then, we employ a shared multi-layer perceptron to generate channel attention maps separately. Next, these generated attention maps are element-wise added. Finally, we use the Sigmoid function to obtain the weight information for each channel. The calculation equation is as follows:

$$\begin{aligned} M_c(F) &= \sigma(\text{MLP}(\text{AvgPool}(F)) + \text{MLP}(\text{MaxPool}(F))) \\ &= \sigma(W_1(W_0(F^c_{\text{avg}})) + W_1(W_0(F^c_{\max}))) \end{aligned} \tag{6}$$

The terms F^c_{avg} and $F^c_{\max}$ respectively denote average pooling and maximum pooling operations in the channel dimension. $W_0 \in \mathbb{R}^{C/r \times C}$ and $W_1 \in \mathbb{R}^{C \times C/r}$ represent the weight parameters of a shared multi-layer perceptron. σ represents the Sigmoid activation function.

2) SPATIAL ATTENTION MODULE

To compute spatial attention, we first apply average pooling and max pooling operations along the spatial dimension to obtain the spatial average and maximum values for each spatial location. Afterward, these average and maximum values are combined through concatenation. Then, convolutional operations are performed to generate a three-dimensional spatial attention map. Finally, a Sigmoid function is employed to obtain the weight information. The calculation equation is as follows:

$$\begin{aligned} M_s(F) &= \sigma(f^{7\times7}([\text{AvgPool}(F); \text{MaxPool}(F)])) \\ &= \sigma(f^{7\times7}([F^s_{\text{avg}}; F^s_{\max}])) \end{aligned} \tag{7}$$

The terms F^s_{avg} and $F^s_{\max}$ respectively denote average pooling and maximum pooling operations in the spatial dimension. $f^{7\times7}$ represents a convolution operation with a kernel size of 7×7. σ signifies the Sigmoid activation function.

B. NETWORK STRUCTURE

Inspired by the CBAM [19], as shown in Figure 3, this paper introduces this module into the Dueling DQN network architecture to enhance the neural network's perception and expressive capabilities. The integrated model, D3QN_CBAM, allows the intelligent agent to automatically focus on critical aspects in the state (such as vehicles approaching intersections) without being disturbed by noise or irrelevant features.

The state in this system consists of three components: normalized vehicle velocity information, vehicle position information, and the phase state information of the traffic light. Firstly, the normalized vehicle velocity information and vehicle position information are fed into the first convolutional layer to extract relevant features, thereby generating a feature map. Subsequently, this feature map is input into the CBAM module. Within the CBAM module, the feature map undergoes channel attention and spatial attention processing, enabling the model to focus more on important features.

The channel attention mechanism enhances features in each channel, while the spatial attention mechanism captures important information at different positions.

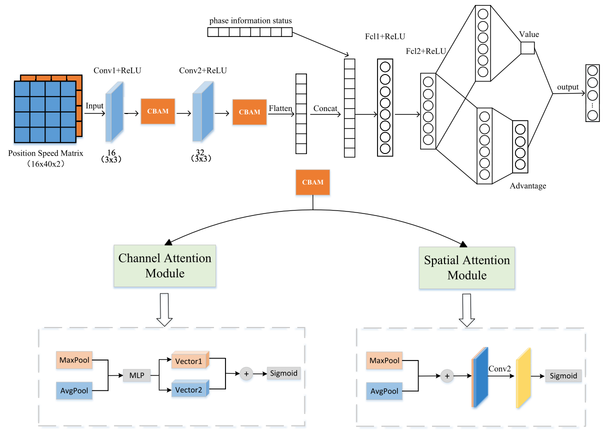


FIGURE 3. The structure of the Q-network.

Next, the feature map processed by the CBAM module is passed to the second convolutional layer for further processing. Afterward, these features obtained through convolutional operations are once again fed into the CBAM module to enhance attention feature weights. Subsequently, the output data is flattened and concatenated with the phase state information, which consists of binary values (0 and 1) and has a size of 8. Finally, these concatenated data are processed through two fully connected layers and a ReLU operation to produce the output, which is used by two subsequent branches. The first branch is used to compute the state value $V(s)$, while the second branch is used to compute the advantage value $A(a)$ for each action, with its dimensionality matching that of the output layer. The Q value of each action is calculated as the sum of the state value and the centralized advantage value, as shown in the following equation:

$$V(s) = V(s;\eta,\alpha)$$
$$A(a) = [A(s,a;\eta,\beta) - \frac{1}{|A|}\sum_a A(s,a;\eta,\beta)]$$
$$Q(s,a;\eta,\alpha,\beta) = V(s) + A(a) \tag{8}$$

where η represents the parameters shared in the common section of the network, α denotes the parameters exclusive to the value network, and β signifies the parameters exclusive to the advantage network.

C. TRAINING PROCESS

This study utilizes the Double DQN core concept to train model parameters. The fundamental idea behind Double DQN is to mitigate the problem of overestimation by introducing two independent neural networks, thereby enhancing the performance of Q-Learning. One neural network is used for selecting the best action, while the other is used to estimate the Q-values associated with the selected action. The advantage of this design is the effective reduction of overestimation in target Q-values, leading to a more accurate and robust training process. The model can more reliably guide training through the dual-network structure, improving stability and efficiency in RL tasks.

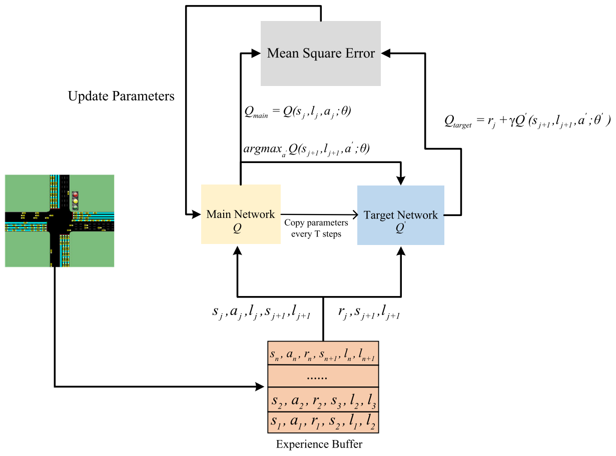

Mean Square Error
Update Parameters
$Q_{main} = Q(s_j, l_j, a_j; \theta)$
$argmax_a Q(s_{j+1}, l_{j+1}, a'; \theta)$
$Q_{target} = r_j + \gamma Q'(s_{j+1}, l_{j+1}, a'; \theta')$
Main Network Q
Copy parameters every T steps
Target Network Q'
$s_j, a_j, l_j, s_{j+1}, l_{j+1}$
r_j, s_{j+1}, l_{j+1}
$s_n, a_n, r_n, s_{n+1}, l_n, l_{n+1}$
......
$s_2, a_2, r_2, s_3, l_2, l_3$
$s_1, a_1, r_1, s_2, l_1, l_2$
Experience Buffer

FIGURE 4. Model training process.

As depicted in Figure 4, the model training process involves the interaction between the agent and the environment, generating experiences (including the current state, executed action, reward value, and next state) stored in an experience replay pool. When the experience pool reaches its maximum capacity, the "first in, first out" principle is applied, with new data replacing the oldest data. This ensures that the experience pool retains a certain amount of the most recent data for subsequent training. Both the main and target networks share the same network structure, but their parameters are independent. During Double DQN training, random batch sampling is typically employed to reduce the correlation between samples and enhance training stability and efficiency. Specifically, a batch of experience samples is randomly extracted from the experience replay buffer, and these samples are then used to update the parameters of the neural networks.

The prediction of the current state's Q-value in the main network, given the current state information (s_j, l_j), is computed as follows:

$$Q_{\text{eval}}^{\text{DDQN}} = Q\left(S_j, a_j; \theta_t\right) \tag{9}$$

Input the next state information (s_{j+1}, l_{j+1}) into the main network to compute the optimal action a' for the subsequent state, which can be expressed as $\text{argmax}_{a'} Q(s_{j+1}, l_{j+1}, a'; \theta)$. Subsequently, transmit the selected action and the next-state information (s_{j+1}, l_{j+1}) to the target network for the calculation of the target Q-value. The equation for computing the target Q-value is as follows:

$$Q_{\text{target}}^{\text{DDQN}} = R_j + \gamma Q'\left(S_{j+1}, \text{argmax}_{a'} Q\left(S_{j+1}, a'; \theta_t\right); \theta_t'\right) \tag{10}$$

Among them, Q and Q' represent the main network and the target network, respectively. S_j and a_j represent the current state and action, respectively. R_j represents the immediate reward value obtained after executing the action a_j in the current state S_j. S_{j+1} and a' respectively represent the state and action at the next moment. θ_t, $\theta_t^{'}$ respectively represent the main network weight and the target network weight at time t.

In the selection of a loss function, Mean Square Error (MSE) was employed as the chosen metric. MSE is a frequently utilized method for assessing disparities between model predictions and actual values. It computes the squared differences between the predicted values and actual values for each data point, subsequently averaging these differences to obtain a measure of overall error. The calculation equation is as indicated in equation (11).

$$\text{Loss}(\theta) = \frac{1}{n}\sum_{t=1}^{n}\left(Q_{\text{target}}^{\text{DDQN}} - Q\Big(S_t, A_t; \theta_t\Big)\right)^2 \tag{11}$$

D. ALGORITHM DESCRIPTION

The pseudocode for the training algorithm of the model used in this paper is as shown in Algorithm 1.

V. EXPERIMENTS

In this section, we will introduce the relevant aspects of the experiments, including the experimental platform, traffic flow Settings, comparative algorithms, and simulation results.

A. EXPERIMENTAL PLATFORM

In this study, we utilized SUMO (Simulation of Urban Mobility) [20] as the simulation platform to evaluate the performance and effectiveness of the proposed method. SUMO is an open-source microscopic traffic simulation tool, and we employed it to simulate certain scenarios within the urban traffic system, generating diverse traffic flows. The software and hardware platforms used in this simulation experiment are illustrated in Table 1.

B. TRAFFIC FLOW SETTINGS

This study employed the Weibull distribution for data generation to simulate real-world traffic conditions. The arrival time of vehicles follow a Weibull distribution, and its probability density function is represented by equation (12). Here, x represents one of the possible values that the random variable can take, λ is the scale parameter with a value of 1, and a is the shape parameter with a value of 2.

$$f(x;\lambda,a) = \begin{cases} \frac{a}{\lambda}\left(\frac{x}{\lambda}\right)^{a-1} e^{-\left(\frac{x}{\lambda}\right)^a} & x \geqslant 0 \\ 0 & x < 0 \end{cases} \tag{12}$$

From the histogram of the traffic flow distribution in Figure 5, it can be observed that the total simulation duration is 5200 seconds, with a peak traffic flow from 0 to 2500 seconds and a low traffic flow from 2500 seconds to 5200 seconds. Thereby simulating the high and low peaks of real life. The vehicles have an acceleration of 0.8 m/s^2, a deceleration of 4.5 m/s^2, a length of 5 meters, a minimum spacing of 2.5 meters, and a maximum speed of 13 m/s. The probabilities of vehicles entering the road network for straight, right-turn, and left-turn movements are 70%, 15%, and 15%, respectively. The number of vehicles is specified in Table 2.

Algorithm 1 D3QN_CBAM for Traffic Signal Control

```
1  Initialize: main network $Q$, target network $Q'$,
   experience replay memory $M$, discount factor $\gamma$,
   batch size $B$, target network update frequency $F$,
   training epochs $E$, training duration per epoch $T$,
   exploration rate $\varepsilon$, exploration decay rate $\varepsilon_{decay}$
2  for episode $= 1, E$ do
3      Initialize observation $s_1, l_1$
4      for $t = 1, T$ do
5          With probability $\varepsilon$ select a random action $a_t$
6          Otherwise select $a_t = \arg\max_a Q(s_t, l_t, a; \theta)$
7          Execute action $a_t$
8          Observe reward $r_t$ and next state $s_{t+1}, l_{t+1}$
9          Store transition $(s_t, l_t, a_t, r_t, s_{t+1}, l_{t+1})$ in $M$
10         Set sample counter $C = C + 1$
11         Set $s_t = s_{t+1}, l_t = l_{t+1}$
12         if $C \geq B$ then
13             Sample random minibatch of transitions
14             $Q_{eval} = Q(s_j, l_j, a_j; \theta)$
15             $a' = \arg\max_{a'} Q(s_{j+1}, l_{j+1}, a'; \theta)$
16             $Q_{next} = Q'(s_{j+1}, l_{j+1}, a'; \theta')$
17             $Q_{target} = r_j + \gamma Q_{next}$
18             Calculate Loss $= \text{MSE}(Q_{target}, Q_{eval})$
19             Update the parameters $\theta$
20             Set $L \leftarrow L + 1$
21             if $L\%F == 0$ then
22                 update $\theta' = \theta$
23             end
24         end
25     end
26     Set $\varepsilon = \max(0.01, \varepsilon_{decay} * \varepsilon)$
27 end
```

TABLE 1. Software and hardware platforms.

Name	Type
CPU	AMD Ryzen 7 5800H with Radeon Graphics
RAM	32GB
GPU	NVIDIA GeForce RTX 3060
OS	Windows 11
Python	3.7.3
PyTorch	1.13+cu116
SUMO	1.14.1

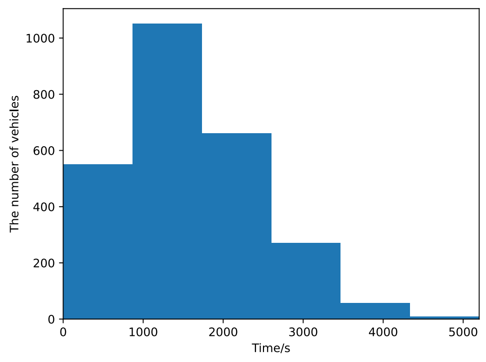


FIGURE 5. Traffic flow distribution histogram.

TABLE 2. The numbers of vehicles.

number of high-traffic vehicles			number of low-traffic vehicles		
straight	left	right	straight	left	right
1540	332	335	274	59	60

C. COMPARISON ALGORITHMS AND PARAMETER SETTINGS

To validate the efficacy of the methodology presented in this paper, it is imperative to commence by contrasting it with three benchmark algorithms. These three benchmark algorithms are delineated as follows:

- **Fixed Time Control (FTC)** [21]: FTC constitutes a set of predefined timing schemes grounded in the Webster timing method. This approach aims to regulate traffic signals according to a fixed timetable, irrespective of the actual traffic flow. In the context of our comparative study, we will scrutinize whether the approach presented in this paper outperforms this conventional timetable-based control method in the realm of traffic flow management.
- **Max Pressure Signal Control (MP)** [22]: MP represents an algorithm designed to enhance traffic management and optimize the efficiency of the road network. Its central objective is to mitigate traffic congestion as much as possible and optimize traffic fluidity through dynamic adjustments of traffic signal control. The MP algorithm introduces the concept of "pressure," which is computed based on the disparity between the queue lengths at entry lanes and exit lanes. During each signal phase selection, the MP algorithm activates the signal phase with the maximum pressure value.
- **Traffic Signal Control based on D3QN**: Similar to the approach outlined in this study, this benchmark algorithm also leverages DRL, with parameters such as state representation, action selection, reward function, and training algorithm aligning with those of the methodology presented in this paper. The key distinction lies in the absence of an attention mechanism in this benchmark algorithm. In our comparative analysis, we will assess the performance disparity between the approach detailed in this paper and this benchmark algorithm to ascertain whether the inclusion of an attention mechanism yields significant improvements in the task of traffic signal control.

The training parameters of the algorithm are as depicted in Table 3:

TABLE 3. Algorithm training parameters.

Parameter name	Parameter value
Learning Rate α	0.001
Discount Rate γ	0.75
Maximum Exploration Rate $\varepsilon_{\max}$	1
Exploration Decay Rate $\varepsilon_{\text{decay}}$	0.96
Minimum Exploration Rate $\varepsilon_{\min}$	0.01
Target Network Update Frequency T	100
Batch Size B	64
Action Space Size N	24
Training Rounds	150
Training Time(s) Per Iteration	5200
Optimizer	Adam

D. SIMULATION RESULTS

In this study's training phase, 150 training iterations were conducted, with each training session lasting 5200 seconds. As can be seen from Figure 6 (a), with an increase in training iterations, we observed a noticeable upward trend in the average reward value, indicating a gradual improvement in DRL performance. Additionally, the D3QN_CBAM algorithm outperforms the standard D3QN in terms of convergence and average reward values after training stabilization.

Figure 6 (c) depicts the comparative results between the D3QN_CBAM algorithm and three benchmark algorithms in terms of average vehicle delay time. From these figures, it is evident that as the number of training iterations increases, the D3QN_CBAM and D3QN algorithms show significant reductions in average vehicle delay time. This underscores the algorithm's effectiveness in controlling traffic flow and optimizing vehicle waiting time. Furthermore, the D3QN_CBAM algorithm outperforms the D3QN algorithm, the MP algorithm, and the fixed timing strategy in terms of convergence and performance regarding average delay time.

Further examination of Figure 6 (b) reveals that the loss value gradually decreases as training progresses and eventually approaches zero. In the early stages of training, due to the high exploration rate of the agent's action selection, the data exhibits significant fluctuations. However, as the number of training iterations increases, the exploration rate decreases, and the agent learns how to select optimal actions in the current state, leading to stabilized data.

In summary, our training results demonstrate that the algorithm presented in this paper can better learn traffic signal timing strategies, significantly reducing traffic congestion and waiting time. By introducing this attention mechanism, our model can adaptively focus on more critical state components, enhancing perception and achieving superior performance in traffic signal control tasks. During the testing process, this study conducted comparative experiments using

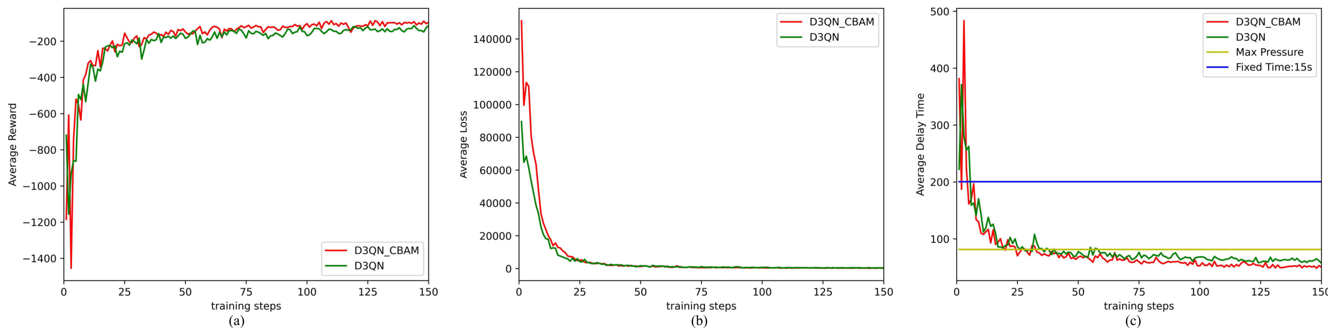


FIGURE 6. The evolution of various metrics for each algorithm over 150 iterations.

TABLE 4. Comparison of the indicators under different methods.

Method	Average Delay Time	Average Queue Length	Average Number of Stops
Fixed Time:15s	193.6	141.2	3.47
Max Pressure	76.4	43.1	1.53
D3QN	54.4	26.7	0.96
D3QN_CBAM	**49.2**	**22.4**	**0.9**

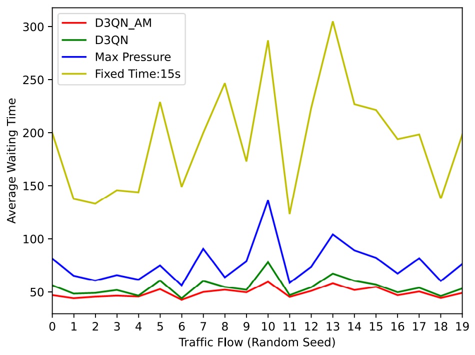


FIGURE 7. Test process.

twenty sets of traffic flows generated with different random seeds to validate the effectiveness of the algorithm proposed in this paper. Each experiment had a simulation time of 5200 seconds, Figure 7 depicts the comparison results between the D3QN_CBAM algorithm and three benchmark algorithms under different random traffic flows in terms of average vehicle delay time. The average delay duration in this study's testing phase is defined as the sum of the delay time at the intersection for all vehicles passing through, divided by the number of vehicles. The results indicate that the proposed D3QN_CBAM algorithm outperforms the three benchmark algorithms, namely, D3QN, MP, and Fix Time 15s, in terms of average vehicle delay time. We can clearly observe that the D3QN algorithm outperforms both the MP algorithm and the fixed-timing strategy of 15 seconds.

To enhance the comprehensiveness of the experiment, this study additionally incorporates comparative tests of two traffic indicators under the D3QN_AM and three benchmark algorithms during the testing phase. The metrics in question are the average queue length and the average number of stops. The average queue length refers to the mean of the total queue lengths for each inbound lane at every step during a testing round. The average number of stops is calculated as the total number of stops made by all vehicles at the intersection during a testing round, divided by the number of vehicles. We summarized and calculated the average of the three traffic indicators in Table 4.

Compared to the original D3QN algorithm, the algorithm proposed in this paper reduces average delay time by 9.6%, average queue length by 16.1%, and the average number of stopped vehicles by 6.25%. When compared to Max Pressure, our algorithm reduces average delay time by 35.6%, average queue length by 48%, and the average number of stopped vehicles by 41.2%. Furthermore, in contrast to Fix Time 15s, our algorithm decreases average delay time by 74.6%, average queue length by 84.1%, and the average number of stopped vehicles by 74.1%.

Based on the above data comparisons, the algorithm proposed in this research paper, D3QN_CBAM, demonstrates remarkable performance in the training and testing phases. Its performance surpasses that of the three benchmark algorithms. It suggests incorporating CBAM into D3QN, as presented in this paper, holds innovative significance. It significantly enhances the feature representation capability of neural networks and increases the weight of vehicles near intersections, thereby improving traffic flow control effectiveness.

VI. CONCLUSION

Vehicles near intersections play a crucial role in overall traffic flow control in typical scenarios. Therefore, this paper introduces an innovative algorithm named D3QN_CBAM to address the issue of emphasizing the weight of vehicles near intersections. Building upon the D3QN algorithm, this approach incorporates CBAM model, including channel and spatial attention, to automatically assign higher weights to vehicles close to intersections. The rigorous experimental analysis demonstrates that D3QN_CBAM outperforms D3QN, Max Pressure, and fixed time duration strategies in both training and testing phases, thus confirming the effectiveness of this algorithm. This innovation could bring significant improvements to the field of traffic management.

ACKNOWLEDGMENT

The authors would like to thank the anonymous referees for their valuable comments and suggestions, which helped a lot in improving the quality of this paper.

REFERENCES

[1] D. Zhao, Y. Dai, and Z. Zhang, "Computational intelligence in urban traffic signal control: A survey," *IEEE Trans. Syst., Man, Cybern., C*, vol. 42, no. 4, pp. 485–494, Jul. 2012.

[2] M. Alsabaan, W. Alasmary, A. Albasir, and K. Naik, "Vehicular networks for a greener environment: A survey," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 3, pp. 1372–1388, 3rd Quart., 2013.

[3] R. S. Sutton and A. G. Barto, "Reinforcement learning: An introduction," *IEEE Trans. Neural Netw.*, vol. 9, no. 5, p. 1054, Sep. 1998.

[4] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 7553.

[5] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing Atari with deep reinforcement learning," 2013, *arXiv:1312.5602*.

[6] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, Oct. 1986.

[7] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[8] J. Gehring, Y. Miao, F. Metze, and A. Waibel, "Extracting deep bottleneck features using stacked auto-encoders," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, May 2013, pp. 3377–3381.

[9] Z. Wang, T. Schaul, M. Hessel, H. Hasselt, M. Lanctot, and N. Freitas, "Dueling network architectures for deep reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1995–2003.

[10] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double Q-learning," in *Proc. AAAI Conf. Artif. Intell.*, 2016, vol. 30, no. 1, pp. 1–7.

[11] T. L. Thorpe and C. W. Anderson, "Traffic light control using sarsa with three state representations," Citeseer, Tech. Rep., 1996.

[12] B. Abdulhai, R. Pringle, and G. J. Karakoulas, "Reinforcement learning for true adaptive traffic signal control," *J. Transp. Eng.*, vol. 129, no. 3, pp. 278–285, May 2003.

[13] S. Richter, D. Aberdeen, and J. Yu, "Natural actor-critic for road traffic optimisation," in *Proc. NIPS*, 2006, pp. 1–8. [Online]. Available: https://api.semanticscholar.org/CorpusID:15806534

[14] K. Liang Tan, S. Poddar, A. Sharma, and S. Sarkar, "Deep reinforcement learning for adaptive traffic signal control," 2019, *arXiv:1911.06294*.

[15] B. Wang, Z. He, J. Sheng, and Y. Chen, "Deep reinforcement learning for traffic light timing optimization," *Processes*, vol. 10, no. 11, p. 2458, Nov. 2022. [Online]. Available: https://www.mdpi.com/2227-9717/10/11/2458

[16] H. Wei, G. Zheng, H. Yao, and Z. Li, "IntelliLight: A reinforcement learning approach for intelligent traffic light control," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*. New York, NY, USA: Association for Computing Machinery, Jul. 2018, pp. 2496–2505, doi: 10.1145/3219819.3220096.

[17] B. Liu and Z. Ding, "A distributed deep reinforcement learning method for traffic light control," *Neurocomputing*, vol. 490, pp. 390–399, Jun. 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S092523122101818X

[18] W. Genders and S. Razavi, "Using a deep reinforcement learning agent for traffic signal control," 2016, *arXiv:1611.01142*.

[19] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "CBAM: Convolutional block attention module," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 3–19.

[20] M. Behrisch, L. Bieker, J. Erdmann, and D. Krajzewicz, "SUMO—Simulation of urban mobility: An overview," in *Proc. 3rd Int. Conf. Adv. Syst. Simul.*, 2011, pp. 1–6.

[21] F. V. Webster, "Traffic signal settings," Road Res., Tech. Paper, 1958, vol. 39.

[22] P. Varaiya, "Max pressure control of a network of signalized intersections," *Transp. Res. C, Emerg. Technol.*, vol. 36, pp. 177–195, Nov. 2013. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0968090X13001782

PENG WANG was born in Jiangxi, China, in 1998. He is currently pursuing the master's degree with the School of Digital Industry, Jiangxi Normal University. His research interests include swarm intelligence algorithms, deep learning, reinforcement learning, and especially the optimization of traffic signal control. His usual hobby is to use programming to solve problems in life.

WENLONG NI (Member, IEEE) received the B.Sc. degree from Tsinghua University, China, in 1999, the M.Sc. degree from Kobe University, Japan, in 2003, and the Ph.D. degree from The University of Toledo, USA, in 2008. Within last ten years, since receiving the Ph.D. degree, he has been a Research Engineer in several companies, including top 500 company like Microsoft and Sinopec. He had more than ten years of industrial experience with computer technology, including high performance computation, big data analysis, and processing. During that career period, he has made several recognized major contributions in complex wireless networks research through publications, such as in IEEE Transactions, ICDCS, Globecom, ICC, and WCNC. In 2018, he started his new position with Jiangxi Normal University as an "Oversea Elite" Professor with the School of Computer Information Engineering. His current research interests include performance evaluation, optimal control and the applications in cloud computing, big data analysis, recommendation systems, communications networks, machine learning, and reinforcement learning.