Postprint: Deep Reinforcement Learning Traffic Signal Control with State Prediction

Authors: Tang Muyao, Zhou Dake, Li Tao

Date: 2022-04-07T15:01:57+00:00

Abstract

Deep Reinforcement Learning (DRL) can be widely applied in the field of urban traffic signal control; however, in existing research, the vast majority of DRL agents utilize only the current traffic state for decision-making, resulting in limited control effectiveness under conditions of significant traffic flow variation. This paper proposes a DRL signal control algorithm incorporating state prediction. First, a concise and efficient traffic state representation is designed using one-hot encoding. Then, Long Short-Term Memory (LSTM) networks are employed to predict future traffic states. Finally, the agent makes optimal decisions based on both the current and predicted states. Experimental results on the SUMO (Simulation of Urban Mobility) simulation platform demonstrate that under various traffic flow conditions for both single and multiple intersections, the proposed algorithm achieves the best performance compared with three typical signal control algorithms across metrics including average waiting time, travel time, fuel consumption, and CO2 emissions.

Full Text

0 Introduction

With the improvement of living standards, vehicle ownership continues to grow, and urban traffic congestion has become increasingly severe. Traffic signal control is the most direct and cost-effective approach to improve road traffic efficiency and alleviate congestion. SCATS[1] and SCOOT[2] are currently widely used adaptive traffic signal control systems. The former selects signal timing plans, while the latter utilizes simplified traffic models to solve for optimal signal control strategies. However, the establishment of simplified models relies heavily on assumptions and empirical equations, making such systems less effective for complex and variable real-world traffic scenarios.

In recent years, with the development of artificial intelligence technology, reinforcement learning[3] (RL), particularly data-driven deep reinforcement learning, has demonstrated excellent application prospects in traffic signal control.

Reinforcement learning is a "trial-and-error" learning method that learns optimal policies through interaction with the environment. In traffic signal control applications, one or several intersections can be viewed as an agent. The agent observes the road network state and makes decisions, learning optimal signal timing schemes by maximizing the reward feedback from the environment. Inspired by the working mode of the human brain, deep learning[4] (DL) can combine low-level features to form more abstract high-level features, effectively handling high-dimensional data. Deep reinforcement learning (DRL) combines DL' s strong perception capabilities with RL' s strong decision-making capabilities, making it highly suitable for traffic signal control tasks.

In 2010, Arel et al.[5] first introduced DRL into the field of traffic signal control, using neural networks to approximate Q-values but lacking experience replay and target network components. Liu et al.[6] proposed the 3DQN_{PSER} algorithm, which uses Priority Sequence Experience Replay (PSER) to update the priority of sequential samples in the experience pool, enabling the agent to obtain preceding samples similar to the current traffic state and improving training efficiency. Wei et al.[7] proposed the Intellilight model, which uses a phase-gate structure to set up independent learning channels, partitions the experience pool according to phases and actions, and conducts experiments with real traffic data. Zheng et al.[8] proposed the FRAP model, which leverages the competitive relationships between different signal phases to achieve universality under symmetric situations such as flipping and rotation in traffic flow. Jin et al.[9] used Threshold Lexicographic Ordering (TLO) to adaptively select optimization objectives and compared the improvement effects of various function approximation methods based on the SARSA algorithm. Tan et al.[10] divided large-scale road networks into several sub-regions, using Per-action DQN or Wolpertinger DDPG for control in each region, and transmitted the learning policies of all agents to a global agent to achieve global learning. These DRL signal control methods are essentially first-order Markov decision processes, where agents make decisions based only on the current state, making it difficult to achieve optimal control effects in complex and variable real-world traffic scenarios. If future states can be reasonably predicted, agents can anticipate possible traffic situations in advance and learn better signal control strategies.

Xu et al.[11] proposed the DRQN model, which integrates hidden states across 8 time steps into the DRL agent input, but this significantly increases state dimensionality and can easily lead to neural network overfitting. Recurrent neural networks have short-term memory capabilities. Chu et al.[12] adopted LSTM networks in DRL agents to extract dynamic traffic information, but these networks did not directly predict future traffic states.

This paper proposes a DRL signal control algorithm called DQN_{SP} that combines state prediction. The main features are: 1) By introducing explicit traffic state prediction, the DRL agent makes optimal decisions using both current and future states. 2) The agent' s state is carefully designed to include the most important traffic information with small data volume, making it easy to predict. The effectiveness and feasibility of the proposed algorithm are verified under various traffic flow conditions at single and multiple intersections, with vehicle flow data simulating real-world peak and off-peak scenarios, demonstrating engineering application value.

1 Research Background

This section introduces the basic concepts and methods of reinforcement learning and deep reinforcement learning, as well as DRL-based traffic signal control algorithms.

1.1 Reinforcement Learning

Reinforcement learning is the third category of machine learning methods alongside supervised and unsupervised learning. An agent learns the optimal policy to achieve a certain goal through continuous interaction with the environment. The Markov Decision Process is a theoretical framework for achieving goals through interactive learning, which is flexible and abstract and can well explain the basic process of reinforcement learning. The agent executes optimal actions with a certain probability according to the current policy and interacts with the environment. The action value function $q^\pi(s,a)$ represents the expected return for the agent taking action a in state s, expressed as:

$$q^\pi(s,a) = E_\pi[G_t|S_t = s, A_t = a] = E_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \mid S_t = s, A_t = a\right]$$

After interacting with the environment, the agent learns the optimal policy. The optimal action value function $q^*(s,a)$ represents the maximum return value obtained by taking action a in state s. According to the Bellman optimality equation:

$$q^*(s,a) = E\left[R_{t+1} + \gamma \max_{a'} q^*(S_{t+1}, a') \mid S_t = s, A_t = a\right]$$

After continuously iterating the optimal action value function, the optimal policy is obtained:

$$\pi^*(a|s) = \begin{cases} 1 & \text{if } a = \arg\max_{a\in A} q^*(s,a) \\ 0 & \text{otherwise} \end{cases}$$

1.2 Deep Reinforcement Learning

DRL is the combination of RL and DL and is one of the advanced learning frameworks in current control systems. In 2013, DeepMind[13] proposed DQN.

Unlike Q-Learning, which uses a table to store all Q-values, DQN uses experience replay to update target values. Samples obtained from agent-environment interactions (s, a, r, s') are stored in an experience pool. Mini-batch samples are uniformly sampled from the experience pool, and deep neural networks are trained using stochastic gradient descent to approximate Q-values. Random sampling can break the strong correlation between samples, making training convergence stable. DQN uses the same network to select actions and calculate target Q-values, which are interdependent during iteration, which is not conducive to algorithm convergence. To solve this problem, DeepMind proposed Nature DQN[14], which uses two networks: the current network Q for action selection and parameter updating, and the target network Q^- for calculating target Q-values. The parameters of the Q^- network do not need real-time iterative updates but are copied from the current network at regular intervals. The optimization objective of the current optimal action value function is expressed as:

$$y = r + \gamma \max_{a'} Q(s', a'; w^-)$$

where w^- represents the parameters of the target value network.

The above algorithms obtain target Q-values directly through the greedy method, which can quickly make Q-values approach the optimization target but easily leads to overestimation. To alleviate the overfitting problem of the model, Van Hasselt et al.[15] proposed Double DQN, which first finds the action corresponding to the maximum Q-value in the current network Q, and then substitutes this action into the target network Q^- to calculate the target Q-value. The optimization objective is expressed as:

$$y = r + \gamma Q(s', \arg\max_{a'} Q(s', a'; w); w^-)$$

The above algorithms train deep Q-networks through experience replay, uniformly sampling samples in the experience pool. However, different samples have different TD errors, and their impact on backpropagation varies. To address this issue, Schaul et al.[16] proposed the prioritized experience replay algorithm based on DDQN, assigning priorities proportional to the absolute TD error of samples and storing these priorities in the experience pool. During training, samples with higher priorities are more likely to be sampled, avoiding worthless iterations and improving algorithm convergence speed. Wang et al.[17] optimized the neural network structure and proposed Dueling DQN, which divides the Q-network into two parts: a value function and an advantage function.

1.3 DRL-Based Traffic Signal Control

DRL-based signal control methods do not require prior knowledge of the scenario but learn optimal policies through continuous interaction with the traffic environment. In this process, an intersection or road network is viewed as an agent, the state is a description of the traffic environment, actions are changes in traffic signals, and rewards measure the change in traffic efficiency after actions.

Existing DRL signal control algorithms differ significantly in state, action, and reward definitions. State definitions fall into two categories: vehicle-based representations (such as real-time images[7,18] and DTSE forms including vehicle position or speed information[6,19,20]) and feature-based value vector representations (such as queue length[7,19,21], cumulative delay[19,20], and waiting time[7,19]). Action definitions include selecting a possible green light phase[6,20,21], maintaining the current phase or switching to the next phase[7,11,19], or changing phase duration[9,22]. The state is a feature matrix or vector of the environment, the action is a discrete selection vector, and the reward is a scalar value related to traffic data. Reward definitions mainly consider queue length[6,7,19,20], delay[7,19,20,22], etc. DRL algorithms are mainly divided into value function-based DQN[6,7,11,19,20], policy-based DDPG[10,23], AC framework-based A2C[12,18], A3C[24], etc.

Some studies have considered the temporal correlation of traffic flow. Yu et al.[23] added vehicle speed to the state representation, and Wei et al.[7] fed real-time images representing vehicle positions into CNN networks. These two methods reflect the dynamic characteristics of traffic flow through reasonable state design. Chu et al.[12] used LSTM networks to fit Q-values, leveraging the network' s memory capabilities to learn trends in traffic information changes, but did not directly predict future traffic states. To overcome DQN' s inability to remember historical information before the current input, Xu et al.[11] proposed the DRQN model, which inputs the current state and several historical states into the agent, which can be viewed as an n-order Markov decision process. Liu et al.[6] used PSER to update the priority of sequential samples in the experience pool, making sample data before the current moment more likely to be sampled. The above methods more or less consider the temporal characteristics of traffic flow but do not directly predict traffic states because microscopic state dimensionality is large, easily leading to the curse of dimensionality, and it is difficult to train satisfactory results when combined with DRL.

2 Deep Reinforcement Learning Traffic Signal Control Algorithm with State Prediction

This paper combines state prediction with the DQN algorithm in DRL, using one-hot encoding to carefully design microscopic states and LSTM to predict future states. The agent makes decisions based on both current and predicted states. This section defines the state, action, and reward, and introduces the network model of the proposed algorithm DQN_{SP}.

2.1 State Definition

This paper utilizes both current and predicted traffic states for decision-making, making state design particularly critical. Based on the DTSE method, nonuniform quantization and one-hot encoding are used to design the state vector. The intersection used for simulation in this paper is a bidirectional six-lane intersection with a length of 500 meters. Along the direction of vehicle travel, the left lane is a left-turn lane, the middle lane is a through lane, and the right lane is a through plus right-turn lane. This paper divides lanes into cells according to a certain length ratio. Figure 1 shows the cell design for the west entrance road of the intersection as an example. The two right lanes are treated as a whole for division, while the left left-turn lane is divided separately. In this way, the lanes in four directions of an intersection are divided into 80 cells. The state is represented by whether there is a vehicle in each cell. If there is a vehicle, the state value is 1; otherwise, it is 0.

As can be seen from the cell design diagram of the west entrance road in Figure 1, 10 cells are divided with 7 meters as the unit near the intersection, each cell can only accommodate one vehicle, which can accurately reflect the vehicle distribution. The cell farthest from the intersection is 230 meters long. Compared with methods that use real-time images[18] or uniformly divide lanes[19] to represent states, this method enables the agent to pay more attention to traffic conditions near the intersection, reduces data dimensionality, and shortens computation time. Using whether there is a vehicle in each cell as the state simplifies traffic information and can reflect the main features of the environment, namely the vehicle distribution near the intersection. Additionally, predicting this one-hot encoded state can be viewed as a binary classification problem, which can improve prediction accuracy compared to traditional regression prediction.

2.2 Action Definition

The agent needs to select appropriate actions to guide traffic based on traffic states. In this paper, the action is defined as selecting a possible signal phase. The action set $A = \{NSG, NSLG, EWG, EWLG\}$ represents green light for straight and right-turn in north-south direction, green light for left-turn in north-south direction, green light for straight and right-turn in east-west direction, and green light for left-turn in east-west direction, respectively. The minimum duration of each phase is set to 10 seconds. For safety, there is a 3-second yellow light during the transition between green and red lights.

2.3 Reward Definition

The agent observes the environment state s_t at time t, executes action a_t, and receives feedback r_t from the environment, which measures the quality of the action and is critical for DRL convergence and effectiveness. In this paper, the reward r_t is defined as the difference in queue lengths of all lanes between adjacent time steps:

$$r_t = \alpha(q_t - q_{t+1})$$

where q_t represents the sum of queue lengths of all lanes in the road network at time t, q_{t+1} represents the sum of queue lengths at the next time step, and α is a coefficient determined through multiple experiments.

2.4 DRL Signal Control Algorithm with State Prediction (DQN_{SP})

The proposed algorithm DQN_{SP} in this paper uses LSTM to predict future microscopic state s_p and concatenates it with the current state s as an augmented state input to the DRL agent. The DRL algorithm uses traditional DQN[13], aiming to verify the effectiveness and feasibility of the algorithm after combining state prediction. The network structure of DQN_{SP} is shown in Figure 2. The optimization objective of the optimal action value function is expressed as:

$$y = r + \gamma \max_{a'} Q(s_p, s, a'; w^-)$$

The algorithm flow of DQN_{SP} is as follows:

```
1. Initialize deep Q-network, LSTM network, and experience pool
2. For episode = 1 to M do
3. Initialize road network environment and import traffic flow data
4. For t = 1 to T do
5. Observe current state $s_t$
6. Use LSTM to predict microscopic state $s_p$ after $n$ time steps
7. Concatenate current state and predicted state, then input to DQN agent
8. Agent executes action $a_t$ based on $\epsilon$-greedy policy
9. Execute action $a_t$, observe reward $r_t$ and next state $s_{t+1}$
10. Store transition $(s_t, a_t, r_t, s_{t+1}, s_p)$ in experience pool
11. End for
12. Calculate optimization objective according to equation (7), update deep
    Q-network using mean squared error loss function
13. Update LSTM network parameters using binary cross-entropy loss func-
    tion
14. End for
```

3 Experimental Results and Analysis

This section first introduces the experimental simulation environment and algorithm hyperparameters, then introduces the benchmark algorithms FTC, SOTL, and DQN, and finally verifies the effectiveness of algorithm DQN_{SP} under various traffic flow conditions at single and multiple intersections.

3.1 Simulation Environment and Hyperparameter Settings

The Traci (Traffic Control Interface) can interact online with various development environments to achieve traffic signal control. This paper uses Ubuntu with GeForce RTX 2080 GPU as the hardware environment. The algorithm is implemented through the deep learning framework Keras and conducts simulation experiments under SUMO v1.6.0.

Intersection Settings: This paper conducts simulations in two scenarios: single intersection and multiple intersections. The intersection consists of 4 perpendicular roads, each 500 meters long, with bidirectional six lanes. Along the direction of vehicle travel, the left side is the left-turn lane, the middle is the through lane, and the right side is the through plus right-turn lane. The multiple intersections scenario is a 2$×$2 grid network composed of 4 identical intersections, with the same intersection configuration as the single intersection scenario.

Traffic Flow Settings: The vehicle generation method has an important impact on traffic signal control. In this paper, vehicle generation follows a Weibull distribution, with the probability density function:

$$f(x;\lambda,k)=\begin{cases}\frac{k}{\lambda}\left(\frac{x}{\lambda}\right)^{k-1}e^{-(x/\lambda)^k} & x\geq 0\\ 0 & x<0\end{cases}$$

where λ is the scale parameter set to 1, and k is the shape parameter set to 2. The majority of vehicles enter the road network within a certain period, simulating real-world peak and off-peak scenarios. Vehicles enter the road network from any entrance, with a 75% probability of going straight, 12.5% probability of left-turn, and 12.5% probability of right-turn. Vehicle length is 5 meters, acceleration is $2m/s^2$, minimum spacing between vehicles is 2.5 meters, entry speed is 36 km/h, and maximum speed is 50 km/h.

Hyperparameter Settings: Referring to literature [7,9,19] and combined with experiments, hyperparameters are set as follows. The number of training episodes is set to 100. The algorithm uses DNN to evaluate Q-values, with 5 hidden layers, width of 400, Adam optimizer, learning rate of 0.001, batch size of 80, 800 iterations per episode, and mean squared error as the loss function. The prediction network uses 6 LSTM units, each with 3 LSTM layers, 80 neurons, Adam optimizer, batch size of 128, 1 iteration per episode, and binary cross-entropy as the loss function. The RL experience pool size is minimum 600 and maximum 50000, discount factor is 0.75, and ϵ-greedy algorithm is used to output actions.

3.2 Experimental Evaluation and Results Analysis

This paper conducts experiments in both single intersection and multiple intersections scenarios. For the single intersection, the simulation duration is 5400 seconds, with 500, 1000, and 1500 vehicles entering the road network, corresponding to low, medium, and high traffic flow conditions, respectively. For multiple intersections, the simulation duration is also 5400 seconds, with 2000 and 3000 vehicles entering the road network, corresponding to low and high traffic flow conditions, respectively. For each traffic flow condition, 20 groups of traffic flow data are generated using random seeds. The average waiting time, average travel time, average fuel consumption, average CO2 emissions, and average cumulative reward of vehicles under the 20 groups of data are used as performance metrics. Among them, average waiting time mainly comes from the time consumed when vehicles are queued, which has the strongest correlation with the defined reward and is the primary metric, while average travel time, fuel consumption, and CO2 emissions are secondary metrics. The proposed algorithm predicts states after 1, 5, and 10 time steps, denoted as DQN_{SP}1, DQN_{SP}5, and DQN_{SP}*{10}, respectively. To verify the effectiveness of prediction, DQN*{SP} is compared with the following benchmark algorithms:

Fixed-time Control (FTC)[25]. FTC predefines a set of timing plans based on the classic Webster timing method and is widely used in real-world traffic scenarios.

Self-organizing Traffic Lights (SOTL)[26]. When the queue length in the red light direction reaches a threshold, the signal in that direction turns green. If there are too many vehicles within a certain distance in the green light direction, the green light duration is extended.

DQN-based Traffic Signal Control. This uses the same DQN algorithm[13] as the proposed DQN_{SP}, with the only difference being that it does not predict future states, so the network input dimension is halved. Other hyperparameter settings and definitions of state, action, and reward are the same as DQN_{SP}.

Figure 3 shows the comparison of cumulative rewards and average vehicle waiting times during training and testing under medium traffic flow conditions at a single intersection. Figure 3(a) presents the cumulative reward comparison between DQN_{SP} and DQN during training under medium traffic flow conditions at a single intersection, showing little difference between the two. This indicates that adding state prediction does not reduce algorithm convergence speed or weaken algorithm stability. Figure 3(b) shows the average vehicle waiting time comparison between DQN_{SP} and the three benchmark algorithms. In the initial training stage, due to too few samples in the experience pool, the agent has not yet learned the correct control strategy, so the average waiting time increases significantly. As training progresses, intersection traffic conditions gradually improve and eventually stabilize.

The trained model is tested on 20 groups of randomly generated traffic flow data, with average performance shown in Table 1. It can be seen that whether predicting states after 1, 5, or 10 steps, DQN_{SP} performance is superior to FTC, SOTL, and DQN. Moreover, DQN_{SP}5 shows the most improvement in the primary metric, reducing average vehicle waiting time by 6.06% and increasing cumulative reward by 5.61% compared to DQN. However, in the three secondary metrics of travel time, fuel consumption, and CO2 emissions, DQN_{SP}1 shows the most significant improvement. Figure 3(c) shows the cumulative reward comparison between DQN_{SP}5 and DQN in 20 tests, and Figure 3(d) shows the average vehicle waiting time comparison between DQN_{SP}5 and the three benchmark algorithms. The results show that compared with traditional FTC and SOTL signal control methods, DRL-based methods are significantly effective in reducing vehicle waiting time, and DQN_{SP}5 outperforms DQN in 18 out of 20 tests.

This paper also conducts experiments in the multiple intersections scenario, where each intersection signal is controlled by one agent. This paper aims to verify the effectiveness of DRL combined with state prediction, so a simple multi-agent collaboration strategy is used: a spatial discount factor is adopted to weaken rewards from other intersections, with the current intersection reward weight at 0.5, neighbor intersections at 0.2, and diagonal intersections at 0.1. The simulation duration is 5400 seconds, with 2000 and 3000 vehicles entering the road network, corresponding to low and high traffic flow, respectively. Tables 4 and 5 list the average performance of algorithms in 20 tests. Under high traffic flow conditions, SOTL performs poorly because vehicle-driven control methods are difficult to work when traffic flow is highly random. Under low traffic flow conditions, DQN_{SP}5 shows the best improvement effect, reducing average waiting time by 8.82% and increasing cumulative reward by 8.11% compared to DQN. However, under high traffic flow conditions, DQN_{SP}_{10} shows the best improvement effect, reducing average waiting time by 4.92% and increasing cumulative reward by 4.59%. This shows that as traffic volume increases, it is necessary to predict states further into the future to more effectively learn traffic variation trends and improve traffic capacity.

In summary, compared with benchmark algorithms, DQN_{SP} can learn better signal control strategies in both single and multiple intersection scenarios, effectively alleviating traffic congestion and reducing fuel consumption and pollution emissions. As traffic volume increases, predicting states further into the future is needed to achieve better control effects.

4 Conclusion

This paper leverages the temporal correlation of traffic data and proposes a deep reinforcement learning traffic signal control algorithm DQN_{SP} that combines state prediction. By extracting high-dimensional traffic features and predicting future microscopic states, better signal control effects are achieved in single intersections, multiple intersections, and various traffic flow conditions. Compared with FTC, SOTL, and DQN algorithms, DQN_{SP} shows improvements in average waiting time, travel time, fuel consumption, and CO2 emissions. Future work will further investigate combining state prediction with more advanced DRL algorithms (such as TD3, SAC, etc.) and validate with real traffic data.

References

[1] Sims A G, Finlay A B. SCATS, splits and offsets simplified (SOS) [J]. Australian Road Research, 1984, 12 (4): 17-33.

[2] Hunt P B, Robertson D I, Bretherton R D, et al. The SCOOT on-line traffic signal optimisation technique [J]. Traffic Engineering & Control, 1982, 23 (4): 190-192.

[3] Sutton R S, Barto A G. Reinforcement learning: an introduction [M]. MIT Press, 2018.

[4] LeCun Y, Bengio Y, Hinton G. Deep learning [J]. Nature, 2015, 521 (7553): 436-444.

[5] Arel I, Liu C, Urbanik T, et al. Reinforcement learning-based multi-agent system for network traffic signal control [J]. IET Intelligent Transport Systems, 2010, 4 (2): 128-135.

[6] Liu Zhi, Cao Shipeng, Shen Yang, et al. Signal control of single intersection based on improved deep reinforcement learning method [J]. Computer Science, 2020, 47 (12): 226-232. (Liu Zhi, Cao Shipeng, Shen Yang, et al. Signal control of single intersection based on improved deep reinforcement learning method [J]. Computer Science, 2020, 47 (12): 226-232.)

[7] Wei Hua, Zheng Guanjie, Yao Huaxiu, et al. Intellilight: a reinforcement learning approach for intelligent traffic light control [C]// Proc of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. New York: ACM Press, 2018: 2496-2505.

[8] Zheng Guanjie, Xiong Yuanhao, Zang Xinshi, et al. Learning phase competition for traffic signal control [C]// Proc of the 28th ACM International Conference on Information and Knowledge Management. New York: ACM Press, 2019: 1963-1972.

[9] Jin Junchen, Ma Xiaoliang. A multi-objective agent-based control approach with application in intelligent traffic signal system [J]. IEEE Trans on Intelligent Transportation Systems, 2019, 20 (10): 3900-3912.

[10] Tan Tian, Bao Feng, Deng Yue, et al. Cooperative deep reinforcement learning for large-scale traffic grid signal control [J]. IEEE Trans on Cybernetics, 2019, 50 (6): 2687-2700.

[11] Xu Ming, Wu Jianping, Huang Ling, et al. Network-wide traffic signal control based on the discovery of critical nodes and deep reinforcement learning [J]. Journal of Intelligent Transportation Systems, 2020, 24 (1): 1-12.

[12] Chu Tianshu, Wang Jie, Codecà L, et al. Multi-agent deep reinforcement learning for large-scale traffic signal control [J]. IEEE Trans on Intelligent Transportation Systems, 2019, 21 (3): 1086-1095.

[13] Mnih V, Kavukcuoglu K, Silver D, et al. Playing atari with deep reinforcement learning [J]. arXiv preprint arXiv: 1312.5602, 2013.

[14] Mnih V, Kavukcuoglu K, Silver D, et al. Human-level control through deep reinforcement learning [J]. Nature, 2015, 518 (7540): 529-533.

[15] Van Hasselt H, Guez A, Silver D. Deep reinforcement learning with double q-learning [C]// Proc of the 30th AAAI Conference on Artificial Intelligence. Palo Alto, CA: AAAI Press, 2016, 30 (1): 2094-2100.

[16] Schaul T, Quan J, Antonoglou I, et al. Prioritized experience replay [C]// Proc of the 4th International Conference on Learning Representations. San Juan, Puerto Rico, 2016: 322-355.

[17] Wang Z, Schaul T, Hessel M, et al. Dueling network architectures for deep reinforcement learning [C]// Proc of the 33rd International Conference on Machine Learning. New York: ACM Press, 2016: 1995-2003.

[18] Mousavi S S, Schukat M, Howley E. Traffic light control using deep policy-gradient and value-function-based reinforcement learning [J]. IET Intelligent Transport Systems, 2017, 11 (7): 417-423.

[19] Sun Hao, Chen Chunlin, Liu Qiong, et al. Traffic signal control method based on deep reinforcement learning [J]. Computer Science, 2020, 47 (2): 169-174. (Sun Hao, Chen Chunlin, Liu Qiong, et al. Traffic signal control method based on deep reinforcement learning [J]. Computer Science, 2020, 47 (2): 169-174.)

[20] Van der Pol E, Oliehoek F A. Coordinated deep reinforcement learners for traffic light control [C]// Proc of the 30th Conference on Neural Information Processing Systems. Cambridge, MA: MIT Press, 2016: 1-9.

[21] Wang Xiaoqiang, Ke Liangjun, Qiao Zhimin, et al. Large-scale traffic signal control using a novel multiagent reinforcement learning [J]. IEEE Trans on Cybernetics, 2020, 51 (1): 174-187.

[22] Touhbi S, Babram M A, Nguyen-Huu T, et al. Adaptive traffic signal control: exploring reward definition for reinforcement learning [J]. Procedia Computer Science, 2017, 109: 513-520.

[23] Yu Bingquan, Guo Jinqiu, Zhao Qinpei, et al. Smarter and safer traffic signal controlling via deep reinforcement learning [C]// Proc of the 29th International Conference on Information & Knowledge Management. New York: ACM Press, 2020: 3345-3348.

[24] Genders W, Razavi S. Evaluating reinforcement learning state representations for adaptive traffic signal control [J]. Procedia Computer Science, 2018, 130: 26-33.

[25] Webster F V. Traffic signal settings, road research technical [J]. Road Research Laboratory, 1958, 39.

[26] Cools S B, Gershenson C, D’ Hooghe B. Self-organizing traffic lights: a realistic simulation [M]// Advances in Applied Self-organizing Systems. London: Springer, 2013: 45-55.