

A deep reinforcement learning model for large-scale traffic signal control based on graph meta-learning using local subgraphs

Zhicheng ZHOU^{1,2}, Ya ZHANG^{1,2*} & Xinde LI^{1,2*}

¹*School of Automation, Southeast University, Nanjing 210096, China*

²*Key Laboratory of Measurement and Control of Complex Systems of Engineering, Ministry of Education, Nanjing 210096, China*

Received 8 August 2023/Revised 4 March 2024/Accepted 3 June 2024/Published online 24 March 2025

Abstract This paper investigates the problem of traffic signal control in large-scale road networks. A deep reinforcement learning model based on graph meta-learning using local subgraphs is proposed to control the traffic signal. The entire traffic network is represented as a graph by defining traffic lights as nodes and treating connections between intersections as edges. A graph neural network is used to enhance cooperation and communications between agents since information about neighbors is aggregated. To overcome the challenges in large-scale road networks, the proposed model employs a graph neural network on local subgraphs to reduce the difficulty of training in large-scale road networks. The model trained in small-scale traffic networks is transferred to a large-scale traffic network. Agent knowledge acquired from local subgraphs during the training of a small-scale road network confers advantages to the training of large-scale road networks under the resemblance between the structures of local subgraphs in small- and large-scale road networks. Furthermore, meta-learning is used to facilitate the model's rapid adaptability to unseen large-scale road networks. The advantage of the double Q-learning network is taken to reduce overestimation. In experiments, real-world road networks and synthetic road networks comprising more than 1000 intersections are given to evaluate the effectiveness of the proposed model.

Keywords deep reinforcement learning, traffic signal control, multiagent system, graph neural network

Citation Zhou Z C, Zhang Y, Li X D. A deep reinforcement learning model for large-scale traffic signal control based on graph meta-learning using local subgraphs. *Sci China Inf Sci*, 2025, 68(7): 172203, <https://doi.org/10.1007/s11432-023-4280-6>

1 Introduction

Traffic congestion considerably impacts society and the economy [1, 2]. Traffic signal control (TSC) represents a potential solution for mitigating congestion within urban traffic networks. The selection of an appropriate signal control strategy can alleviate traffic congestion [3].

The primary objective of TSC is to facilitate the safe and efficient movement of vehicles through intersections while optimizing key measures such as travel time and throughput, which are commonly adopted to quantify the effectiveness of a given signal control strategy [4]. TSC systems such as SCATS [5] and SCOOT [6] that rely on expert knowledge may be inefficient despite their widespread adoption. Thus, an increasing number of researchers are exploring intelligent TSC schemes that leverage advanced technologies, such as deep reinforcement learning (DRL) within the field of artificial intelligence (AI), to enhance the efficiency of TSC. Reinforcement learning (RL) explores the learning of knowledge by an agent through its interactions with the environment [7]. The remarkable success of deep Q network (DQN) has given rise to a trend toward the use of DRL, which combines the function approximation capabilities of deep learning (DL) with the strategy generation abilities of RL [8].

When RL is adopted in TSC, every traffic light in the road network is controlled by the agent. The state is composed of queue length, waiting time, and phase, and the reward function usually depends on queue length, waiting time, and throughput [4]. Pol et al. [9] used the DQN framework to address the challenge of TSC. They adopted an image-like representation of the state, which encompassed the positions of vehicles as well as information on traffic lights. Multiple studies have concentrated on enhancing DQN [10–12],

* Corresponding author (email: yazhang@seu.edu.cn, xindeli@seu.edu.cn)

and these efforts have been considered in TSC [13]. An increasing number of novel technologies in DL are being employed within the domain of TSC. A graph neural network (GNN) [14] represents a novel type of neural network architecture that incorporates factors from graph theory. It is suitable to apply a GNN in TSC because of the structure of road networks, which can be represented as a graph. Devailly et al. [15] introduced an inductive graph RL approach that leveraged graph convolutional networks (GCNs) to control traffic signals. Velickovic et al. [16] introduced the attention mechanism to GNNs. Graph attention network (GAT) [16] was used in CoLight [17] to incorporate the temporal and spatial influences of neighboring intersections into target intersections. Yan et al. [18] proposed the graph cooperation Q-learning network traffic signal control (GCQN-TSC) model to enhance collaboration among agents. The proposed model is a graph cooperation network with an embedded self-attention mechanism. Moreover, novel technologies like meta-learning, which can expedite the learning process in new scenarios, were combined with RL to control traffic signals [19, 20].

Despite the extensive research on TSC, relatively few studies have focused on the challenges of managing large-scale road networks that encompass over one thousand traffic lights and must contend with heavy traffic flows. In this paper, the TSC problem of large-scale road networks is studied. Currently, the GNN is a pivotal tool for TSC because of its proficient handling of traffic graph structures, enabling comprehensive capture of the topology and spatial relationships within a traffic network. However, large-scale traffic networks often comprise numerous intersections, exhibiting intricate topological structures. The GNN applied directly for large-scale TSC usually exhibits limited performance and learning efficiency. Moreover, it is difficult to train many agents directly in large-scale traffic networks. Therefore, how to expedite the learning process of the DRL model by leveraging prior experiences should be studied.

Given the aforementioned challenges, in this paper, a novel DRL model based on graph meta-learning using local subgraphs for large-scale TSC is proposed. First, the TSC problem is modeled as a partially observable Markov decision process (POMDP), and the entire road network is represented as a graph with traffic lights defined as nodes and connections between intersections treated as edges. Second, multi-hop neighboring nodes of target nodes are extracted to construct local subgraphs, and then small- and large-scale road networks are depicted by multiple small local subgraphs. A GNN, which can enhance cooperation and communications between agents since information about neighboring nodes is aggregated, is employed on local subgraphs rather than the whole graph to reduce the difficulty of training in large-scale road networks. The model forward architecture comprises the graph information extraction module, the intermediate information processing module, and the final Q-value prediction part. To avoid direct training in large-scale traffic networks from the ground up, the model trained in small-scale traffic networks is transferred to a large-scale traffic network. Furthermore, meta-learning is used to enable the transferred model to quickly adapt to unseen large-scale road networks that include similar local subgraphs with small-scale traffic networks, thereby enhancing its training performance. Finally, experiments on large-scale road networks with high traffic flow show the strong performance of the proposed model when facing a previously unseen large-scale road network.

The main contributions of the paper are as follows. (1) A novel DRL-based TSC model using local subgraphs is proposed to reduce the difficulty of training in large-scale road networks and improve traffic efficiency, where a GNN is used on local subgraphs instead of the entire graph, and the model is trained in small-scale traffic networks and then transferred to a large-scale traffic network. (2) Network training is conducted under the meta-learning framework to facilitate the model's rapid adaptability to unseen large-scale road networks.

The remainder of the paper is organized as follows. Section 2 presents a literature review. Section 3 presents the problem formulation. The proposed model is described in detail in Section 4. Experiments and evaluations are presented in Section 5. Finally, the paper is concluded in Section 6.

2 Related work

Before the application of RL, several classic methods for TSC have been used. SCATS [5], which has been widely adopted in many metropolises, chooses traffic signals according to the predefined performance measurements. Max-pressure control [21] determines traffic signals on the basis of the principle of minimizing the “pressure” of phases at an intersection. Max-queue length greedily selects the phase with the maximum queue length [22].

Recently, important advancements in DL and RL have been made from different research focuses.

Multitasking data in offline RL have been studied. Pessimistic value iteration on the shared offline dataset is conducted to execute multitasking data sharing when the dataset is limited [23]. A diffusion-based method multi-task diffusion model (MTDiff) [24] was proposed in modeling large-scale multi-task offline data. Regarding safe offline policy, Bai et al. [25] proposed the monotonic quantile network for risk-averse policy learning, and the method addresses the distribution shift problem. To generalize policies across diverse domains, Xu et al. [26] proposed the value-guided data filtering algorithm. Furthermore, in terms of exploration, Hao et al. [27] presented several key challenges in efficient exploration and surveyed numerous exploration-related methods. Potential safety issues in RL cannot be ignored. Sun et al. [28] introduced two novel adversarial attack techniques to stealthily and efficiently attack the DRL agents. RL has enhanced the cognitive abilities of decision-making systems, thereby greatly accelerating the development of some domains such as game intelligence [29].

Given these advancements, numerous studies have investigated the TSC problem using AI-based approaches. Deep imitation learning was used to learn expert knowledge through the historical record of manipulations by traffic operators [30]. Nonetheless, this model is less intelligent than RL-based models, as it relies on imitation rather than true learning. Some studies based on RL models focus on traffic scenarios of only one intersection. Li et al. [31] employed a deep stacked autoencoder neural network to approximate the Q-function. Notably, the experimental settings were relatively modest in complexity, as only two phases were considered, while right-turn and left-turn maneuvers were omitted. IntelliLight [32] addresses a simplified one-intersection case. The novel design of memory structure that is beneficial to the training in RL has also been discussed. IntelliLight, which improved the balance and accuracy of Q-value estimations, set up a separate experience pool for each phase/action combination, and the same number of samples was selected from each pool during training. Liang et al. [13] proposed a DRL model that incorporated dueling, target, and double Q-learning networks and prioritized experience replay to control traffic light signals. Moreover, they defined actions as plus or minus 5 s for one and only one phase of the cycle to make the model more flexible.

Although prior studies have predominantly focused on optimizing traffic flow at an isolated intersection, the recent emergence of TSC systems has used multiagent reinforcement learning (MARL) models to optimize across entire road networks. In multi-intersection scenarios, it is a common practice to directly expand single-agent RL to MARL, where each agent is trained and executed independently while leveraging parameter sharing [33–36]. Furthermore, MARL is often implemented within a value-based RL framework [17, 37, 38]. In devising an MARL model for TSC, it is valid to incorporate intuitive principles that are known to be effective in TSC [34, 37–39]. Max-pressure control [21] was considered in the design of state representation and reward function [37]. Wu et al. [38] proposed efficient pressure and considered it the traffic state. The correlation degree based on the number of vehicles waiting between two intersections was introduced in the observation and reward function to facilitate communications between agents [34]. Zhang et al. [39] included the demand for each phase, which is measured in terms of the number of vehicles traveling on the incoming lanes within the effective range, as a state representation.

However, most studies focus on scenarios in which the number of intersections is limited to fewer than 100. In a practical road network, the total count of intersections typically exceeds one thousand. Extensive road networks with more than one thousand traffic lights are infrequently analyzed because of the intricacies of the traffic conditions involved. Although CoLight [17] pioneered the use of GAT to perform experiments on a large-scale road network comprising hundreds of traffic signals, it considered relatively few intersections, i.e., 196, compared with a practical large-scale road network, which involves more than 1000 traffic lights that must be considered. MPLight [33] used max-pressure control [21] to conduct experiments in a Manhattan road network comprising 2510 traffic lights. Nonetheless, the traffic volume utilized was relatively light, and a transferring model that is trained in small-scale road networks to large-scale road networks to improve learning efficiency was ignored. An RL model based on GCNs was explored to generalize to new road networks, and a Manhattan road network involving 3971 traffic lights was considered [15]. However, traffic of two distinct regimes utilized was relatively light, and using a GNN on the entire graph in a 3971 large-scale road network was inefficient.

Furthermore, meta-learning, an innovative AI technique, has been integrated with RL in TSC to enable rapid adaptation to new tasks by leveraging prior experience [19, 20]. Zang et al. [19] employed a gradient-based meta-learning algorithm that involved periodically alternate individual- and global-level adaptation. This approach facilitated the transfer to new tasks, with training and testing scenarios spanning multiple cities and traffic flow conditions. However, GNNs were not used to extract graph information from road networks, and the experimental scenarios were relatively limited in scale; notably, the potential of meta-

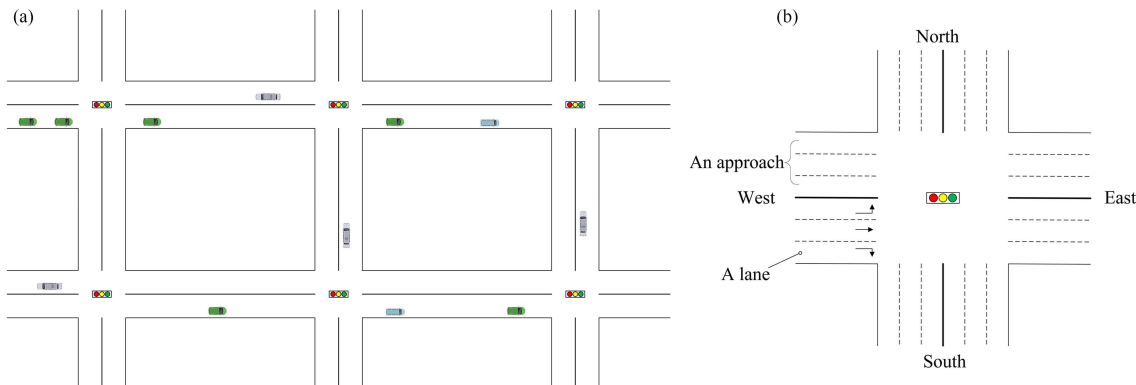


Figure 1 (Color online) Illustration of road networks. (a) Multi-intersection road network; (b) one intersection road network.

learning in tackling TSC on large-scale road networks was not explored. Wang et al. [20] developed a graph neural framework that incorporates a GAT and a long short-term memory (LSTM) network to extract spatial and temporal information. Meta-learning facilitates the graph network in effectively capturing the dynamically evolving characteristics of the intersections by dynamically learning the weights in the GNN. This approach falls under model-based meta-learning. Specifically, the authors trained two meta-knowledge learners to generate weights for GAT and LSTM. Nonetheless, experimental road networks contained relatively few intersections, and meta-learning was not used to facilitate the adaptability to unseen large-scale road networks.

In this work, we design a DRL model based on graph meta-learning using local subgraphs to rapidly adapt to unseen large-scale road networks by utilizing knowledge obtained in the training of a small-scale road network. A GNN is used on local subgraphs instead of the entire graph to fully capture large-scale road network structures and reduce the difficulty of training in large-scale road networks. In addition, network training is conducted under the meta-learning framework to facilitate the model's rapid adaptability to unseen large-scale road networks. In experiments, a 25×40 large-scale road network with high traffic flow is studied using the proposed model.

3 Problem formulation

This paper studies the TSC problem in the context of multiple intersections. Road networks with multiple intersections and single intersections are shown in Figure 1. Figure 1(a) illustrates a multi-intersection road network with six intersections, where traffic signals of multiple intersections must be coordinated simultaneously to regulate traffic flow. The travel time of a vehicle is equal to the time difference between its entering and leaving the road network. The average travel time of the road network is defined as the average travel time of all vehicles that have entered and left the system. TSC aims to minimize the travel time of vehicles by coordinating their movements at intersections. Figure 1(b) shows a typical intersection with eight approaches, including four incoming approaches and four outgoing approaches. Each approach comprises three lanes. The queue length of a lane is defined by the number of waiting vehicles in the lane. A vehicle traverses the intersection from an incoming approach to an outgoing approach, which is called a traffic movement. A movement signal is defined with a green signal indicating that the corresponding traffic movement is allowed and a red signal indicating that the movement is prohibited. As shown in Figure 2(a), eight traffic movement signals are employed, wherein right-turning vehicles are permitted to pass through the intersection irrespective of the traffic signal. A traffic signal phase p is defined as a combination of multiple movement signals. As illustrated in Figures 2(b) and (c), four phases and eight phases are commonly used, respectively. A TSC scheme that comprises a set of phases can be defined as $\{(p_1, t_1), (p_2, t_2), \dots, (p_i, t_i), \dots\}$, where p_i represents the i -th phase and t_i represents the corresponding start time.

We model the TSC problem as a POMDP with $\langle \mathcal{G}(\mathcal{V}, \mathcal{E}), \mathcal{S}, \mathcal{O}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \pi, \gamma \rangle$. $\mathcal{G}(\mathcal{V}, \mathcal{E})$ is used to represent the road network as a graph, where \mathcal{V} and \mathcal{E} denote the node set and edge set, respectively, with intersections as nodes and connections between intersections as edges. Any edge in \mathcal{E} is undirected because lanes are bidirectional. Assume the road network has N intersections, and each intersection is

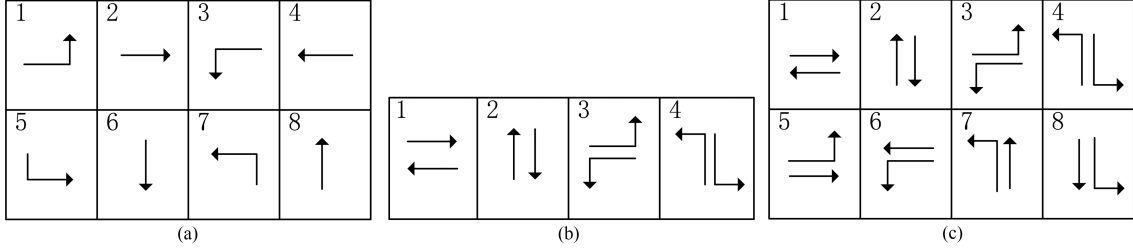


Figure 2 Illustration of movement signals and phases. (a) Movement signals; (b) four phases; (c) eight phases.

controlled by an agent. \mathcal{S} is the state space, and \mathcal{O} is the observation space. An agent can only observe a partial state of the entire system. The observation of agent i , which is defined as $o_i \in \mathcal{O}$, only includes information of intersection i . Observation information o_i can be used by N_i , which denotes the neighbor agents of agent i . \mathcal{A}_i represents the action space of agent i , and \mathcal{A} is the joint action space of all agents, where $\mathcal{A} = \langle \mathcal{A}_1, \mathcal{A}_2, \dots, \mathcal{A}_i, \dots, \mathcal{A}_N \rangle$. The system arrives at a new state after all agents have taken action. \mathcal{P} denotes the transition probability, where $P(s'|s, a)$ represents the probability of the transition to state s' when taking action a at state s with $s, s' \in \mathcal{S}$ and $a \in \mathcal{A}$. $\mathcal{R} : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$ represents the reward function. Agent i takes action $a \in \mathcal{A}_i$ when it receives observation $o_i \in \mathcal{O}$ in accordance with policy $\pi_i(a|o_i)$, and π is the joint policy where $\pi = \langle \pi_1, \pi_2, \dots, \pi_i, \dots, \pi_N \rangle$. γ is the discount factor.

3.1 States

The selection of an appropriate state representation that accurately reflects the environment's condition is crucial in TSC. In this paper, the current traffic light phase p and the queue lengths of the incoming lanes of each intersection are used as the state. The observation o_i of agent i is described as a vector $[m_1, \dots, m_j, \dots, m_8, q_1, \dots, q_l, \dots, q_L]$, where binary value m_j represents the j -th traffic movement signal, with 0 indicating permission and 1 prohibition, q_l represents the queue length of lane l , and L is the number of incoming lanes of the intersection.

3.2 Actions

Action a_i of agent i is chosen as phase p from phase sequence $\langle p_1, \dots, p_k, \dots \rangle$, where p_k represents the k -th phase in the configuration of four phases or eight phases.

3.3 Rewards

The reward function is a critical element for the RL model in TSC. The negative sum of queue lengths of all incoming lanes of intersections is employed as the reward in the proposed model. The reward r_i of intersection i and the reward r of the road network are designed as follows:

$$r_i = - \sum_{l \in \mathcal{L}_i} q_l, \quad (1)$$

$$r = \frac{1}{N} \sum_{i=1}^N r_i, \quad (2)$$

where q_l is the queue length of lane l , \mathcal{L}_i is the set of incoming lanes of intersection i , and N is the number of intersections of the road network.

In addition, using the queue length as the reward and incorporating it as part of the state representation, optimization of the reward can be facilitated [22]. The agent aims to maximize the reward.

4 Proposed model

In this section, the proposed DRL model based on graph meta-learning using local subgraphs is described in detail.

DRL models are difficult to implement for multi-intersection TSC because they can extract and effectively utilize the information of the complicated road network structure. Particularly, in a large-scale

traffic road network where more than one thousand traffic lights need to be controlled, abundant agents and a complex road network structure complicate the training of the RL model. Such scenarios result in a considerable reduction in training performance. In the proposed model, a GNN is employed on local subgraphs to reduce the difficulty of training in large-scale road networks and to extract the information on road networks adequately. Using meta-learning and local subgraphs, better performance can be obtained through less training in large-scale road networks after transferring from small-scale road networks.

First, how each traffic network is represented as an entire graph and how local subgraphs are generated from this graph are elucidated. Second, a GNN employed on local subgraphs to capture graph structure and conduct observation information processing is explicated. Third, the final representation is described as each local subgraph that is used to predict Q-values. Finally, double DQN training under a meta-learning framework is elucidated.

4.1 Graph representation of the road network

We represent the road network as a graph where each intersection and node is considered a node and edge, respectively. Figure 3 presents an example of a graph representation of the road network, wherein each node is depicted as a circle, and each road is illustrated as a line.

4.2 Local subgraphs

GNN representations may fail to entirely capture the intricate structure of large and complex graphs [40, 41]. In large-scale road networks, especially where more than one thousand traffic lights need to be controlled, it is inefficient to directly use a GNN on the entire graph. In the proposed model, we apply a GNN on local subgraphs instead of the entire graph. \mathcal{V} is the set of nodes in the entire graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$. The generated local subgraph of node $i \in \mathcal{V}$ is defined as $U_i = (\mathcal{V}^i, \mathcal{E}^i)$, where \mathcal{V}^i represents a set of nodes $\{k | d(k, i) \leq h\}$ and \mathcal{E}^i denotes a set of edges between these nodes, $d(k, i)$ represents the shortest distance between nodes k and i in the entire graph, and h defines the size of the local subgraph. Node i is defined as the central node of the local subgraph U_i . In addition, the representation of the local subgraph U_i is defined by the representation of node i . The number of local subgraphs in an entire graph equals the number of nodes in the entire graph. Figure 3 shows an example of the generation of local subgraphs U_i and U_j , where h is 2, U_i is marked in purple, U_j is marked in green, and arrows represent directions of generating local subgraphs. A node may belong to two or more different local subgraphs. As shown in Figure 3, k is a special node that belongs to U_i and U_j simultaneously. Useful information can be preserved when the GNN is applied on the local subgraphs compared with the entire graph [42]. We employ the GNN on local subgraphs to fully capture the structure of large-scale road networks and reduce the difficulty of training. The utilization of agent knowledge acquired from local subgraphs during the training of a small-scale road network confers advantages to the training of large-scale road networks by virtue of the resemblance between the structures of local subgraphs in small- and large-scale road networks.

4.3 Model architecture

In this section, the architecture of the proposed model is elucidated. In the proposed model, each intersection is controlled by an agent, and the parameters are shared among agents. Parameters sharing among agents can reduce the overall number of parameters, thus streamlining the learning process for individual agents. The same parameters are updated across multiple local subgraphs, thereby enabling the proposed model to generalize effectively to previously unseen large-scale road networks. The proposed model is trained directly in a small-scale road network and then transferred and trained in a large-scale road network.

The main architecture of the proposed model is illustrated in Figure 4 and includes the graph information extraction module, the intermediate information processing module, and the final Q-value prediction part.

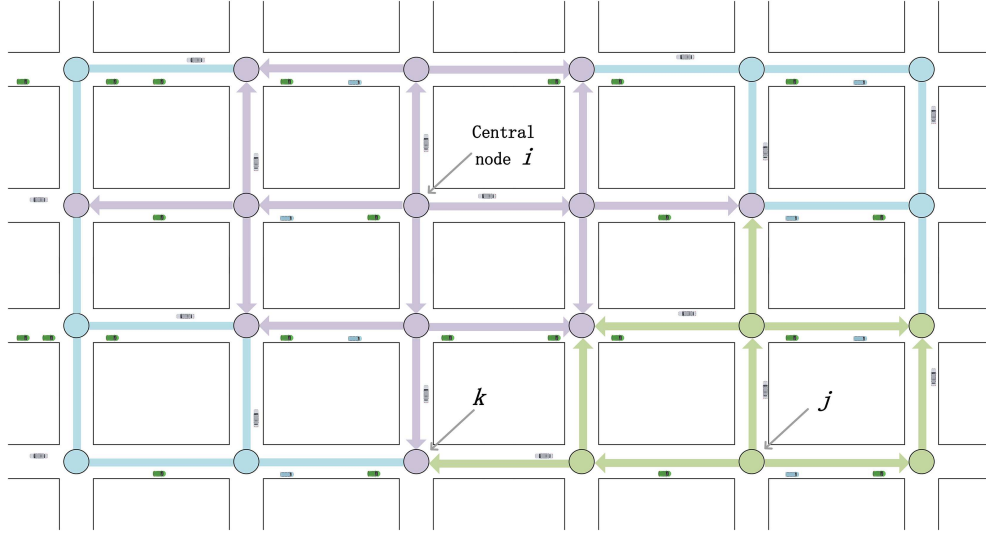


Figure 3 (Color online) Illustration of the representation of the traffic network.

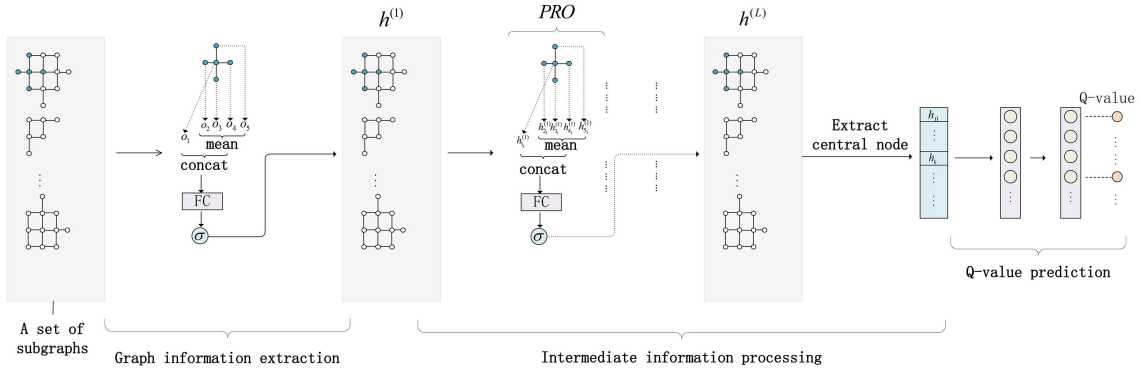


Figure 4 (Color online) Main framework of the model.

4.3.1 Graph information extraction

In this part, GNN is employed to capture graph structure and conduct observation information processing. We first process observation information of agents using a GraphSAGE layer [43]:

$$m_{i_k}^{(0)} = \text{mean}(\{o_j, \forall j \in N_k(i)\}), \quad (3)$$

$$h_{i_k}^{(1)} = \sigma(\text{concat}(o_i, m_{i_k}^{(0)})W_0 + b_0), \quad (4)$$

where $m_{i_k}^{(0)}$ represents the aggregation of the observation information of i 's neighbors in U_k , $o_j \in \mathbb{R}^e$ denotes the observation of intersection j , e represents the feature dimension of observation, $N_k(i)$ is the set of adjacent intersections of i in the local subgraph U_k , $\text{mean}(\cdot)$ denotes the mean aggregator [43], $h_{i_k}^{(1)}$ is the first processed feature of intersection i in local subgraph U_k , $W_0 \in \mathbb{R}^{2e \times n}$ and $b_0 \in \mathbb{R}^n$ are the weight matrix and bias vector to be learned, respectively, n denotes the number of neurons in the hidden layer, $\text{concat}(\cdot)$ denotes the operation of concatenating, and σ denotes a ReLU function. As mentioned in Subsection 4.2, a node probably belongs to two or more different local subgraphs simultaneously. In this case, the processed feature $h_{i_k}^{(1)}$ of intersection i in U_k differs from that in other local subgraphs. The embeddings of initial nodes encode the observable features of intersections.

4.3.2 Intermediate information processing

In this section, final representations of local subgraphs are obtained:

$$m_{i_k}^{(1)} = \text{mean}(\{h_{j_k}^{(1)}, \forall j \in N_k(i)\}), \quad (5)$$

$$h_{i_k}^{(2)} = \sigma \left(\text{concat} \left(h_{i_k}^{(1)}, m_{i_k}^{(1)} \right) W_1 + b_1 \right). \quad (6)$$

Eqs. (5) and (6) are defined as $\text{PRO}_k^1(h_{i_k}^{(1)})$, where $m_{i_k}^{(1)}$ represents the first aggregation of the first processed features of i 's neighbors in U_k , $h_{i_k}^{(2)}$ represents the second processed feature of intersection i in local subgraph U_k , and $W_1 \in \mathbb{R}^{2n \times n}$ and $b_1 \in \mathbb{R}^n$ are the weight matrix and bias vector to be learned, respectively.

The equation $h_{i_k}^{(p+1)} = \text{PRO}_k^p(h_{i_k}^{(p)})$ can be represented as

$$m_{i_k}^{(p)} = \text{mean} \left(\left\{ h_{j_k}^{(p)}, \forall j \in N_k(i) \right\} \right), \quad (7)$$

$$h_{i_k}^{(p+1)} = \sigma \left(\text{concat} \left(h_{i_k}^{(p)}, m_{i_k}^{(p)} \right) W_p + b_p \right), \quad (8)$$

where $m_{i_k}^{(p)}$ represents the p -th aggregation of the p -th processed features of i 's neighbors in U_k , $W_p \in \mathbb{R}^{2n \times n}$ and $b_p \in \mathbb{R}^n$ are weight matrix and bias vector in the p -th calculation.

Then, the final representation of node i in U_k can be obtained as

$$\begin{aligned} h_{i_k}^{(2)} &= \text{PRO}_k^1(h_{i_k}^{(1)}), \\ &\dots \\ h_{i_k}^{(L+1)} &= \text{PRO}_k^L(h_{i_k}^{(L)}), \end{aligned} \quad (9)$$

where $L = h - 1$, and h is the size of the local subgraph, as mentioned in Subsection 4.2.

Afterward, the final representation of the local subgraph U_k is obtained:

$$h_k = h_{k_k}^{(L+1)}, \quad (10)$$

where h_k denotes the final representation of U_k .

4.3.3 Q-value prediction

The final representation h_k of local subgraph U_k is used to predict Q-value:

$$q_k = h_k W_q + b_q, \quad (11)$$

where $W_q \in \mathbb{R}^{n \times v}$ and $b \in \mathbb{R}^v$ are the weight matrix and bias vector to be learned, respectively, v denotes the number of actions, and q_k represents the predicted Q-value of node k .

4.3.4 Network training under a meta-learning framework

Network training is conducted under a meta-learning framework. A meta-learning framework facilitates learning and improves the model's rapid adaptability to previously unseen large-scale road networks. In this paper, the network training process is divided into two parts: inner loop training (ILT) and outer loop training (OLT). Moreover, two sets of trainable parameters of the neural network are updated at the corresponding training process, including inner loop parameters of the value network θ_{inner} and inner loop parameters of the target network θ_{inner}^- , outer loop parameters (or meta parameters) of the value network θ_{meta} , and outer loop parameters of the target network θ_{meta}^- . An agent makes decisions in the proposed model with the latest θ_{inner} . The value of θ_{inner} is always replaced with the value of the latest θ_{meta} after OLT, and they are initialized with the value of the latest θ_{meta} before the start of each episode. The value of θ_{meta}^- is replaced with θ_{meta} every few OLTs. Moreover, the value of θ_{inner}^- is also replaced with the value of θ_{meta}^- when θ_{inner} is replaced or initialized. Notably, θ_{inner}^- remains unchanged during ILT before the next OLT.

Differences between ILT and OLT are expatiated as follows. First, the training data used in ILT is the experience collected during that training episode, while the data used in OLT includes not only experience collected during that training episode but also data sampled from the pool of stored samples. Second, the training frequency of ILT is higher than that of OLT. Several ILTs are conducted before the next OLT.

During each training, double DQN [11] is used. This method alleviates the problem of overestimation in the proposed model. The network training is conducted as follows:

$$y_i^t = \frac{r_i^t}{\beta} + \gamma Q \left(\bar{o}_i^{t'}, \arg \max_{a_i^{t'}} Q \left(\bar{o}_i^{t'}, a_i^{t'}, U_i; \theta \right), U_i; \theta^- \right), \quad (12)$$

$$\mathcal{L}(\theta) = \frac{1}{T} \sum_{t=1}^T \sum_{i \in W_t} \|y_i^t - Q(\bar{o}_i^t, a_i^t, U_i; \theta)\|^2, \quad (13)$$

$$\theta = \theta - \alpha \nabla_{\theta} \mathcal{L}(\theta), \quad (14)$$

where T denotes the total number of time steps in the updating, W_t represents the set of nodes in the updating at time step t , y_i^t represents the target value, o_k^t represents the observation of node k at time step t , \mathcal{V}^i represents the node set of local subgraph U_i , $\bar{o}_i^t = \{o_k^t, \forall k \in \mathcal{V}^i\}$ is the set of observations of nodes in subgraph U_i , $\bar{o}_i^{t'}$ denotes the next observation for \bar{o}_i^t , a_i^t is the action of node i at time step t , θ contains all trainable parameters of the value network in the proposed model, $Q(\bar{o}_i^t, a_i^t, U_i; \theta) = q_i(a_i^t)$, q_i is referred in (11), $q_i(a_i^t)$ represents the Q-value under action a_i^t , r_i^t denotes the reward of node i at time step t , β represents a standardized coefficient, θ^- represents all trainable parameters of the target network in the proposed model, \mathcal{L} represents the loss function, and α is the learning rate. Eq. (14) is conducted by the adaptive moment estimation (Adam) [44].

Finally, during parameters updating, ILT is conducted using the equation below:

$$\theta_{\text{inner}} = \theta_{\text{inner}} - \omega \nabla_{\theta_{\text{inner}}} \mathcal{L}(\theta_{\text{inner}}), \quad (15)$$

where ω is the learning rate of ILT. Furthermore, OLT is conducted using the following equation:

$$\theta_{\text{meta}} = \theta_{\text{meta}} - \lambda \nabla_{\theta_{\text{meta}}} \mathcal{L}(\theta_{\text{meta}}), \quad (16)$$

where λ is the learning rate of OLT.

The training process of the proposed model is given in Algorithm 1. All training on small- or large-scale road networks is given. Notably, the input θ_{meta} should be initialized randomly when training on a small-scale road network; then, the output θ_{meta} is transferred to training on large-scale road networks.

In the following discussion, we discuss the computational complexity of the proposed model. The proposed algorithm introduces additional overhead mainly due to local subgraphs and meta-learning. In terms of local subgraphs, each node i corresponds to a local subgraph U_i whose center node is i , which requires additional space to store compared with the GNN on the entire graph because some nodes and edges are simultaneously present in different subgraphs. Additionally, more time is required than the GNN on the entire graph because of some repeated computations in different local subgraphs, such as some aggregations. Despite the additional storage space and time required for local subgraphs, their incorporation considerably enhances performance, reduces training difficulty, and confers advantages to the training of large-scale road networks by virtue of the resemblance between the structures of local subgraphs in small- and large-scale road networks. In terms of meta-learning, additional space is required because two sets of parameters, namely, inner loop parameters $\theta_{\text{inner}}, \theta_{\text{inner}}^-$ and outer loop parameters $\theta_{\text{meta}}, \theta_{\text{meta}}^-$, require the double amount of space. ILT and OLT require more computation time than algorithms without meta-learning. Despite incurring additional overhead, meta-learning facilitates the model's rapid adaptability to unseen large-scale road networks.

5 Experiments

We conduct experiments on CityFlow [45]. Small-scale road networks and a large-scale road network with 1000 traffic lights are used to validate the proposed model. Ablation experiments are conducted to ascertain the effectiveness of local subgraphs and meta-learning in the proposed model. The computations on graphs were performed using DGL [46], an efficient and scalable Python package for DL on graphs. All experiments were conducted five times using different random seeds, and the average outcomes were used for evaluation.

Algorithm 1 Training process of the proposed model.

Input: ω, λ , initial greedy factor ϵ , minimum value of greedy factor ϵ_{\min} , ϵ decay rate τ , two empty pools D_1, D_2 , ILT frequency t_{inner} , OLT frequency t_{outer} , minimum and maximum amount of stored samples for D_1 : B_{\min} and B_{\max} , maximum amount of stored samples for D_2 : B'_{\max} , maximum amount of training samples in D_2 for OLT: $B_{\text{outer}}, \theta_{\text{meta}}$, road networks;

Output: Optimized meta parameters θ_{meta} ;

Create the entire graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ and local subgraphs $\{U_i = (\mathcal{V}^i, \mathcal{E}^i), \forall i \in \mathcal{V}\}$;

Replace the value of $\theta_{\text{inner}}, \theta_{\text{inner}}^-, \theta_{\text{meta}}^-$ with θ_{meta} ;

for episode = 1, ... **do**

 Empty D_1 ;

 Replace the value of θ_{inner} with θ_{meta} , replace θ_{inner}^- with θ_{meta}^- ;

for time step $t = 1, \dots$ **do**

 Receive observations $\{o_i^t, \forall i \in \mathcal{V}\}$ from the environment;

for $i \in \mathcal{V}$ **do**

 Choose actions a_i^t using ϵ greedy according to $Q(\bar{o}_i^t, a_i, U_i; \theta_{\text{inner}})$;

end

 Take actions $\{a_i^t, \forall i \in \mathcal{V}\}$ and get rewards $\{r_i^t, \forall i \in \mathcal{V}\}$, next observations $\{o_i^{t+1}, \forall i \in \mathcal{V}\}$;

 Add $\{o_i^t, a_i^t, r_i^t, o_i^{t+1}, \forall i \in \mathcal{V}\}$ into D_1, D_2 ;

if $|D_1| > B_{\max}$ **then**

 Remove the oldest experience in D_1 ;

end if

if $|D_2| > B'_{\max}$ **then**

 Remove the oldest experience in D_2 ;

end if

if $t \bmod t_{\text{inner}} = 0$ **and** $|D_1| \geq B_{\min}$ **then**

do randomly draw samples from D_1 ;

 Use (15) to update θ_{inner} ;

until all samples in D_1 have been drawn;

end if

if $t \bmod t_{\text{outer}} = 0$ **and** $|D_1| \geq B_{\min}$ **then**

if episode = 1 **then**

do randomly draw samples from D_1 ;

 Use (16) to update θ_{meta} ;

until all samples in D_1 have been drawn;

else

do randomly draw samples from $D_1 \cup \min(B_{\text{outer}}, |D_2|)$;

 Use (16) to update θ_{meta} ;

until all samples in $D_1 \cup \min(B_{\text{outer}}, |D_2|)$ have been drawn;

end if

 Update θ_{meta}^- , replace the value of θ_{inner} with θ_{meta} , replace θ_{inner}^- with θ_{meta}^- ;

end if

$\epsilon \leftarrow \epsilon \times \tau$ but make sure $\epsilon \geq \epsilon_{\min}$.

5.1 Settings of the traffic simulator

CityFlow is an MARL environment for large-scale urban traffic scenarios, and it can provide various road networks and flexible traffic flow scenarios. During interactions between the agents and CityFlow, CityFlow sends real-time state data to the agents, after which the agents make decisions and send actions back to CityFlow. Subsequently, each traffic light is assigned a phase, and the simulator executes the traffic signal action for 10 s. Then, each green signal is followed by a three-second yellow signal and a two-second all-red signal. Afterward, the agents receive rewards and new real-time state data, and the decision-making process is repeated as before. A time step is defined as the interval where all intersections have taken action once. In the experiments, each time step is set to 15 s. Vehicles in the traffic road network can travel straight or turn left or right at intersections. Traffic lights control the movements of going straight and turning left, while turning right is unrestricted. Vehicles can turn right at intersections without being affected by traffic lights. Vehicles entering the road network travel according to the predetermined speed and routes until they leave the road network.

5.2 Datasets

5.2.1 Synthetic datasets

A synthetic 25×40 large-scale road network is used. Each intersection in the traffic road network consists of four incoming approaches and four outgoing approaches, each approach comprising three lanes. The three lanes of an incoming approach represent going straight, turning left, and turning right, respectively. A vehicle must choose the corresponding lane according to its route after it crosses the intersection. Traveling to any lane of outgoing approaches is permissible. Each lane is 300 m in length and 4 m in width. The settings of vehicles settings are shown in Table 1. The predefined routes of vehicles

Table 1 Settings of vehicles in synthetic datasets.

Parameter	Value
Length	5.0 m
Width	2.0 m
Maximum acceleration	2.0 m/s ²
Maximum deceleration	4.5 m/s ²
Usual acceleration	2.0 m/s ²
Usual deceleration	4.5 m/s ²
Minimum acceptable gap with leading vehicle	2.5 m
Maximum cruising speed	16.67 m/s
Desired headway time with leading vehicle	1.5 s

encompass various directions, including northbound, southbound, westbound, and eastbound, as well as combinations of alternating right and left turns and alternating left and right turns. The traffic flow of vehicles whose routes encompass northbound, southbound, westbound, and eastbound is 19500 vehicles per 300 s, and that of vehicles whose routes include combinations is 1200 vehicles per 300 s. The traffic flow is approximately 2096 vehicles per 300 s in the large-scale road network used in MPLight [33] and is 300 vehicles per 300 s in IG-RL [15]. These flows are lighter than the 20700 vehicles per 300 s used in this paper.

5.2.2 Real-world datasets

Real-world datasets¹⁾, including the 4×4 Hangzhou road network and the 28×7 New York City road network, are used. Different real-world traffic flow data are used in Hangzhou and New York.

Experiments are divided into two sections. The first is to train directly on each traffic flow situation of each road network, excluding the 25×40 road network, and the second is to transfer the model trained on the small-scale 4×4 road network to the large-scale 25×40 road network. In our experimental settings, each episode is defined as a period of 3600 s.

5.3 Compared methods

- Fixedtime [47]: Phases are executed according to a predetermined plan that includes a fixed sequence of phases and their corresponding durations.
- CoLight [17]: CoLight is a DRL method that uses GAT [16] to incorporate temporal and spatial influences of neighboring intersections to target intersections.
- QL-DQN [22]: QL-DQN is an adaptation of vanilla DQN [8] for TSC, where the queue length and current phase are used as the state, and the queue length serves as the reward. An agent controls each intersection, and parameters between agents are shared. An agent cannot observe neighbor information.
- GCN [48]: On the basis of QL-DQN, neural network architecture is replaced with two layers of GCN [48] and a fully connected layer. Information on adjacent intersections is extracted in the GCN model. Other settings, such as parameter sharing, are the same as those of QL-DQN.
- GraphSAGE [43]: Neural network architecture is replaced with two layers of GraphSAGE [43] and a fully connected layer based on QL-DQN. The aggregator in the GraphSAGE layer is the mean type. Other settings are the same as those of QL-DQN.
- GMSLight: The proposed model, which is an intelligent TSC model based on graph meta-learning using local subgraphs, is named GMSLight.

Intelligent control methods apply the configuration of four phases, whereas Fixedtime employs the configuration of eight phases.

5.4 Evaluation metrics

We use two metrics to evaluate different models. First, the reward of the road network is used because it shows the optimization objective of agents. The reward curve explicitly displays the learning quality of agents. This metric is only employed in the 4×4 road network because the reward is improper for large-scale road networks. Second, average travel time is used to evaluate models, as described in Section 3. This metric has been widely used in numerous studies [17, 33, 35].

1) <https://traffic-signal-control.github.io>.

Table 2 Hyperparameter settings.

Hyperparameter	Value
The number of neurons in a hidden layer	64
The size of the local subgraph h	2
Epochs of ILT for the road network of 16 nodes	5
Epochs of OLT for the road network of 16 nodes	30
Epochs of ILT for road networks of 196 nodes and 1000 nodes	2
Epochs of OLT for road networks of 196 nodes and 1000 nodes	10
Time steps in one episode	240
Learning rate of ILT and OLT	0.001
ILT frequency t_{inner}	20
OLT frequency t_{outer}	60
θ_{meta}^- update frequency	5
γ	0.8
Initial greedy factor ϵ	0.8
Minimum value of greedy factor ϵ_{min}	0.2
ϵ decay rate τ	0.95

5.5 Performance comparison

In this section, we compare the proposed model with other models listed in Subsection 5.3 in terms of the reward and the average travel time. The hyperparameter setting is presented in Table 2. The batch size of ILT in scenarios of 16 nodes and 196 nodes at time step t_{step} is $3 \times t_{\text{step}}$. The batch size of OLT in scenarios of 16 nodes and 196 nodes at time t_{step} is

$$\begin{cases} 3 \times t_{\text{step}}, & \text{episode} = 1, \\ 3 \times t_{\text{step}} + 720, & \text{episode} > 1. \end{cases} \quad (17)$$

The batch size of ILT in scenarios of 1000 nodes at time step t_{step} is

$$\begin{cases} 1000 \times t_{\text{step}} \div 20, & t_{\text{step}} \leq 150, \\ 1000 \times 150 \div 20, & t_{\text{step}} > 150, \end{cases} \quad (18)$$

where 1000 represents the number of nodes and 20 represents the number of batches. The batch size of OLT in scenarios of 1000 nodes at time step t_{step} is

$$\begin{cases} (1000 \times t_{\text{step}}) \div 20, & t_{\text{step}} \leq 150, \text{episode} = 1, \\ (1000 \times 150) \div 20, & t_{\text{step}} > 150, \text{episode} = 1, \\ (1000 \times t_{\text{step}} + 150000) \div 20, & t_{\text{step}} \leq 150, \text{episode} > 1, \\ (1000 \times 150 + 150000) \div 20, & t_{\text{step}} > 150, \text{episode} > 1. \end{cases} \quad (19)$$

One sample in the batch is $\langle o_i^t, a_i^t, r_i^t, o_i^{t+1} \rangle$, which is mentioned in Algorithm 1. To ensure fair comparisons between different models, we keep the amount of training data D_{episode} per episode approximately constant across all models. D_{episode} refers to the number of $\langle o_i^t, a_i^t, r_i^t, o_i^{t+1} \rangle$ mentioned in Algorithm 1. In experiments of the 4×4 road network, D_{episode} is 3840 for GMSLight in the first episode and 7680 after the first episode. Moreover, D_{episode} is 3840 for other methods in the first episode, 7680 in the second episode, and 8000 after the second episode. In experiments of the 28×7 road network, D_{episode} is 47040 for QL-DQN in the first episode and 80000 after the first episode. Furthermore, D_{episode} is 47040 for other methods in the first episode and 94080 after the first episode. D_{episode} is 300000 for all methods after the first episode on the 25×40 road network. After an episode is trained, the evaluation is to be done after resetting the CityFlow environment with a greedy policy. Notably, only ILT is kept in the test for GMSLight. Other methods do not undergo training during the testing process. For the training data used in an episode for GMSLight, 240 sample sets $\{\langle o_i^t, a_i^t, r_i^t, o_i^{t+1} \rangle, \forall i \in \mathcal{V}\}$ are used for ILT, 240 sample sets are used for OLT in the first episode, and 480 sample sets are used after the first episode during the training in the 4×4 road network and the 28×7 road network. Furthermore, the number of used sample sets is 150 for ILT, 150 for OLT in the first episode, and 300 after the first episode in the 25×40 road network.

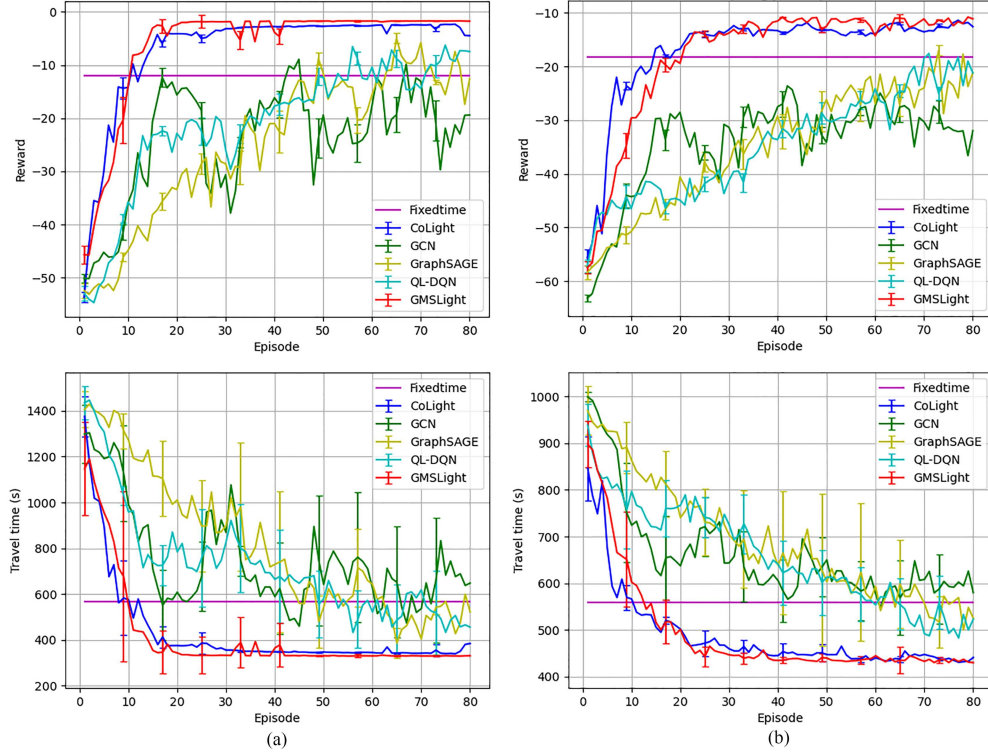


Figure 5 (Color online) Learning directly on Hangzhou dataset. (a) Hangzhou A; (b) Hangzhou B.

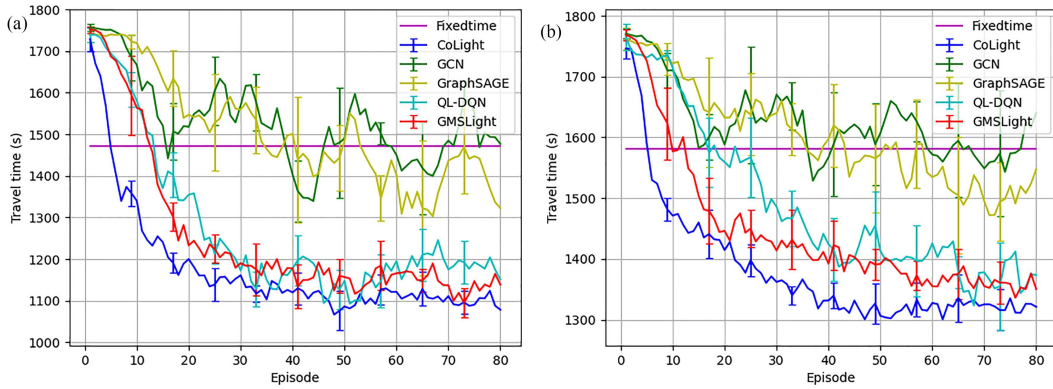


Figure 6 (Color online) Learning directly on New York dataset. (a) New York A; (b) New York B.

5.5.1 Performance of direct learning

In this section, we compare these models in 4×4 and 28×7 road networks. In Figure 5, GMSLight achieves a higher reward and shorter travel time than other methods. Among all methods, GMSLight and CoLight demonstrate the fastest learning. Figure 5(a) shows that GMSLight learns faster than CoLight after 10 episodes and displays strong stability after 42 episodes. Figure 6 shows that GMSLight outperforms all other methods except for CoLight when training directly on the 28×7 road network. The overall inferior performance of GMSLight compared with that of CoLight in New York A and New York B can be attributed to the specific intersection locations and traffic conditions in the New York dataset. These factors pose challenges for GMSLight in effectively handling particular traffic patterns, causing it to underperform CoLight in such scenarios. However, as shown in Figure 7, when the size of local subgraphs is 1, GMSLight achieves lower travel time than CoLight in certain episodes, and overall, GMSLight can achieve a lower optimal travel time than CoLight. This comparison indicates that GMSLight still has the potential to effectively handle traffic patterns in New York.

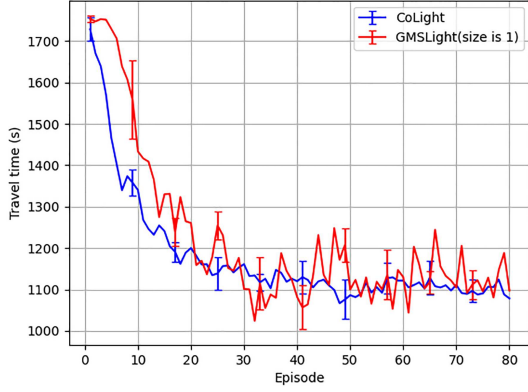


Figure 7 (Color online) Learning directly on New York dataset with local subgraph size being 1.

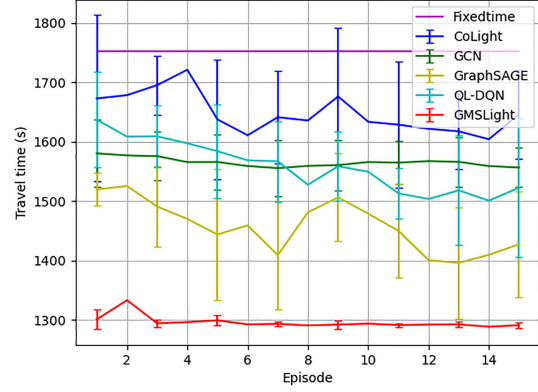


Figure 8 (Color online) Transferring to 25×40 road network.

Table 3 Time consumed for the experiments with GMSLight.

Scenario	Training time (h)	Testing time (h)	Other time (h)	Total time (h)
Hangzhou A (80 episodes)	5.29 (69.24%)	1.45 (18.98%)	0.90 (11.78%)	7.64
Hangzhou B (80 episodes)	5.68 (68.60%)	1.60 (19.32%)	1.00 (12.08%)	8.28
New York A (80 episodes)	29.63 (55.67%)	13.25 (24.90%)	10.34 (19.43%)	53.22
New York B (80 episodes)	28.78 (55.67%)	12.55 (24.27%)	10.37 (20.06%)	51.70
25×40 road network (15 episodes)	9.71 (31.41%)	8.79 (28.44%)	12.41 (40.15%)	30.91

5.5.2 Performance of transferring

In this section, all methods are evaluated in the 25×40 road network using parameters obtained from the training in the 4×4 road network as initial parameters. The final parameters of the training in the first real-world traffic flow data are used as initial parameters. Figure 8 illustrates that GMSLight outperforms the other methods, and notably, GMSLight displays superior performance after only one training episode. This model's rapid adaptability to a previously unseen road network benefits from mechanisms of meta-learning and local subgraphs in the proposed model. Training on local subgraphs of a small-scale road network facilitates the acquisition of knowledge that can be applied to train on unseen large-scale road networks with similar local subgraphs.

Figures 5 and 8 show the learning efficiency because different methods have basically the same amount of training data D_{episode} between two episodes. GMSLight learns more efficiently than other methods when transferring to large-scale road networks.

5.5.3 Training time

The time consumed for the experiments with GMSLight using two AMD EPYC 7302 16C/32T CPUs and a GPU of NVIDIA RTX 3080TI is shown in Table 3; the values in parentheses indicate the proportion of the corresponding time relative to the total time. Testing time refers to the time consumed during the testing process. Other time includes simulator running time and data processing time during the interaction between agents and the environment. Simulator running time and data processing time during the testing process are included in the testing time. The total number of episodes is 15 for the 25×40 road network and 80 for other scenarios. As the number of agents increases, the proportion of other time becomes larger, while the proportion of training time becomes smaller because the simulator running and data processing in large-scale scenarios are more time-consuming. In the 25×40 road network, the training time is only 9.71 h, accounting for 31.41% of the total time of 30.91 h.

5.5.4 Ablation comparisons

We perform ablation experiments to verify the effectiveness of meta-learning and local subgraphs used in the proposed model. Two configurations are compared with GMSLight. Configuration 1 excludes meta-learning, and configuration 2 excludes local subgraphs. Figure 9(a) shows that the mechanisms of meta-learning and local subgraphs facilitate learning and enhance model stability. Figure 9(b) shows that

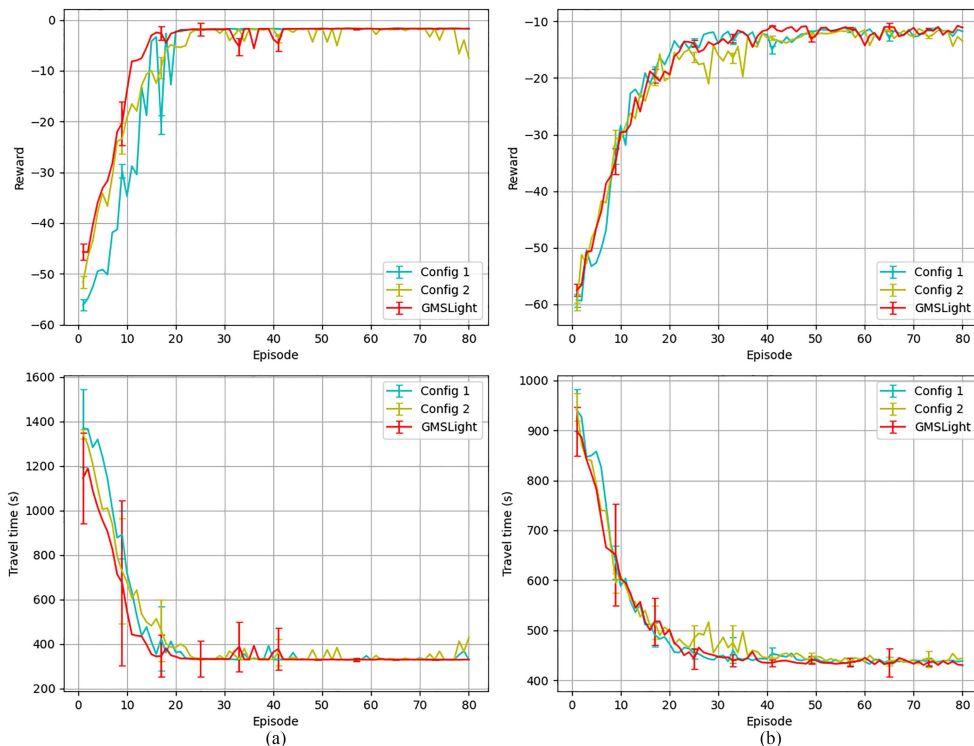


Figure 9 (Color online) Ablation study on learning directly on Hangzhou dataset. (a) Hangzhou A; (b) Hangzhou B.

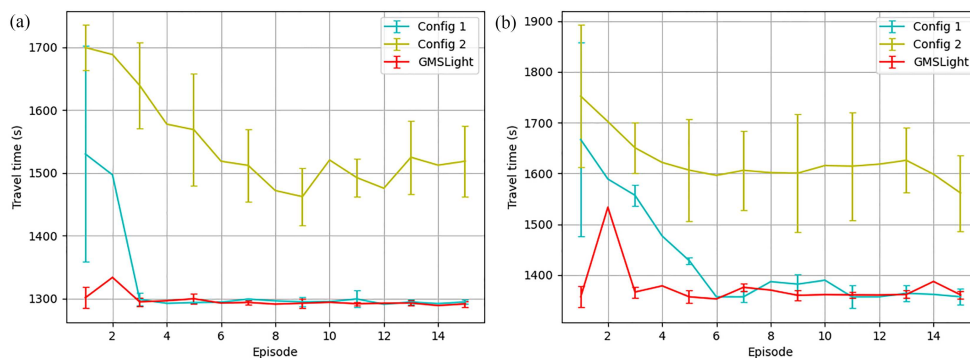


Figure 10 (Color online) Ablation study on the network transferring. (a) 25×40 road network; (b) 35×50 road network.

meta-learning and local subgraphs contribute to the improved performance of the proposed model. An extra 35×50 road network is used to conduct the ablation experiment. The traffic flow of the 35×50 road network is 26700 vehicles per 300 s, while other settings remain consistent with those of the 25×40 road network except for the number of intersections. Figure 10 demonstrates that the application of meta-learning improves learning efficiency when transferring to large-scale road networks. Although the 35×50 road network has 750 more intersections and 6000 more vehicles per 300 s than the 25×40 road network, meta-learning still enables the model to converge rapidly. The mechanism of local subgraphs contributes to less travel time because knowledge of local subgraphs is used to improve training on previously unseen large-scale road networks, and the GNN on local subgraphs extracts information on large-scale road network structures adequately.

5.5.5 Size of local subgraphs

An ablation study has been included to discuss the optimal size of local subgraphs. This study is conducted with a size ranging from 1 to 3. As shown in Figures 11 and 12, the convergent rewards in Hangzhou or New York road networks are similar when the size is equal to 1, 2, and 3. In Hangzhou B, Figure 11(b) shows that a size of 1 achieves slightly better convergent performance despite exhibiting

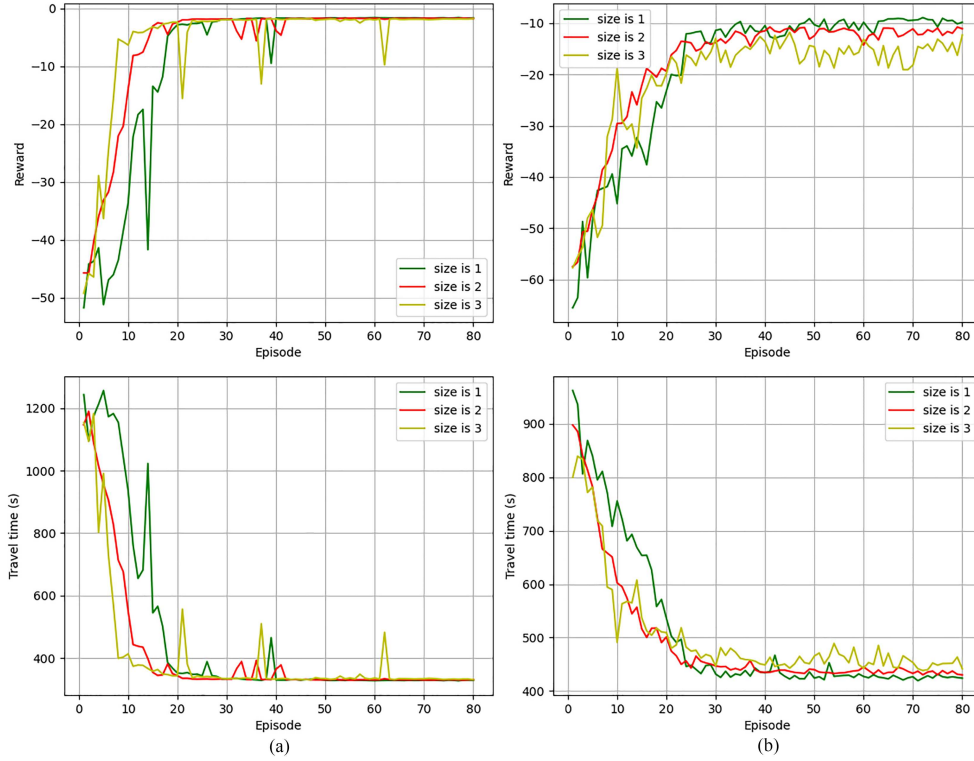


Figure 11 (Color online) Ablation study on the local subgraph size on Hangzhou dataset. (a) Hangzhou A; (b) Hangzhou B.

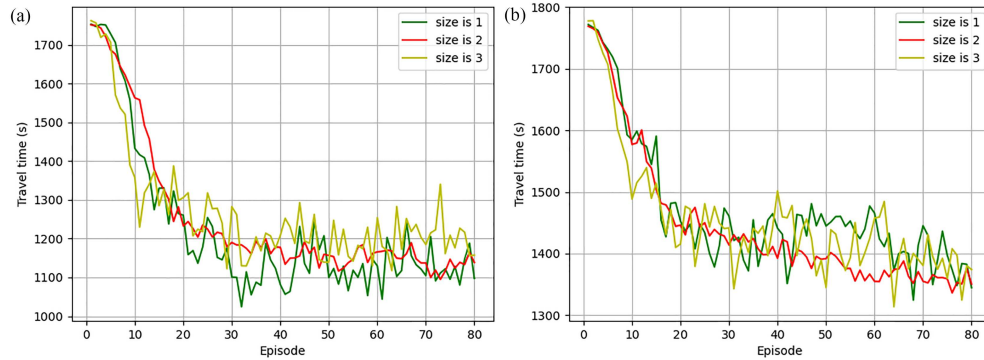


Figure 12 (Color online) Ablation study on the local subgraph size on New York dataset. (a) New York A; (b) New York B.

relatively slow learning speed in the early stage. In New York A, as shown in Figure 12(a), a size of 1 achieves the best performance. A size of 2 in New York B, as depicted in Figure 12(b), demonstrates basically the most optimal performance. For large-scale road networks, Figure 13 shows that sizes 2 and 3 achieve relatively optimal performance and exhibit the highest learning speed in the 25×40 road network and the 35×50 road network, respectively. Size 1 performs poorly in large-scale road networks because of the insufficiency of local subgraphs of this small size in extracting and effectively using the intricate structural information of the large-scale road network. Overall, a size of 1 is optimal in small road networks, whereas sizes 2 and 3 exhibit superior performance in relatively large road networks. The optimal size may increase with an increase in road networks.

6 Conclusion

In this paper, we propose a DRL model to solve large-scale TSC problems. Specifically, a GNN is used on local subgraphs to reduce the difficulty of training in large-scale road networks and to extract the information on large-scale road network structures adequately. In addition, it is used to promote the

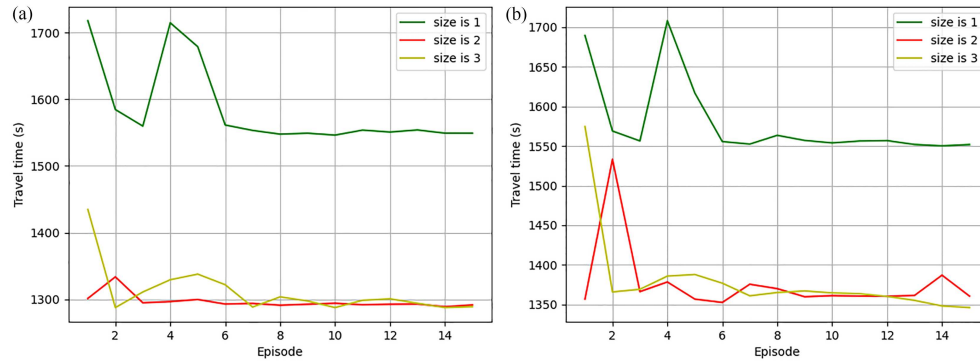


Figure 13 (Color online) Ablation study on the local subgraph size after transferring. (a) 25×40 road network; (b) 35×50 road network.

training of the transferred model. Meta-learning is introduced to facilitate training when transferring to previously unseen large-scale road networks. In the future, certain research directions will be of interest to us. First, the setting of actions can be more flexible by considering the integration of choosing phase and changing duration. Hybrid action space methods can be taken to address mixed control problems flexibly [49]. Second, concepts related to multiagent systems focusing on facilitating collaboration can be considered to enhance performance [50,51]. Finally, how to improve algorithm efficiency in terms of time and space must be determined.

Acknowledgements This work was supported by National Science and Technology Major Project (Grant No. 2021ZD0112702) and National Natural Science Foundation of China (Grant Nos. 62373100, 62233003).

References

- Grote M, Williams I, Preston J, et al. Including congestion effects in urban road traffic CO₂ emissions modelling: do local government authorities have the right options? *Transp Res Part D-Transp Environ*, 2016, 43: 95–106
- Sweet M. Traffic congestion's economic impacts: evidence from US metropolitan regions. *Urban Studies*, 2014, 51: 2088–2110
- Yue W, Li C, Chen Y, et al. What is the root cause of congestion in urban traffic networks: road infrastructure or signal control? *IEEE Trans Intell Transp Syst*, 2022, 23: 8662–8679
- Wei H, Zheng G, Gayah V, et al. A survey on traffic signal control methods. 2019. ArXiv:1904.08117
- Lowrie P R. SCATS, Sydney Co-Ordinated Adaptive Traffic System: A Traffic Responsive Method of Controlling Urban Traffic. Darlinghurst: Roads and Traffic Authority NSW, 1990
- Hunt P, Robertson D, Bretherton R, et al. The scoot on-line traffic signal optimisation technique. *Traffic Eng Control*, 1982, 23: 190–192
- Sutton R S, Barto A G. Reinforcement Learning: An Introduction. Cambridge: MIT Press, 2018
- Mnih V, Kavukcuoglu K, Silver D, et al. Human-level control through deep reinforcement learning. *Nature*, 2015, 518: 529–533
- Pol E, Oliehoek F A. Coordinated deep reinforcement learners for traffic light control. In: *Proceedings of Advance Neural Information Processing Systems*, 2016
- Wang Z, Schaul T, Hessel M, et al. Dueling network architectures for deep reinforcement learning. In: *Proceedings of the 33rd International Conference on Machine Learning*, 2016. 1995–2003
- van Hasselt H, Guez A, Silver D. Deep reinforcement learning with double q-learning. In: *Proceedings of the 30th AAAI Conference on Artificial Intelligence*, 2016. 2094–2100
- Schaul T, Quan J, Antonoglou I, et al. Prioritized experience replay. 2015. ArXiv:1511.05952
- Liang X, Du X, Wang G, et al. A deep reinforcement learning network for traffic light cycle control. *IEEE Trans Veh Technol*, 2019, 68: 1243–1253
- Scarselli F, Tsoi A C, Gori M, et al. Graphical-based learning environments for pattern recognition. In: *Structural, Syntactic, and Statistical Pattern Recognition*. Berlin: Springer, 2004. 42–56
- Devailly F X, Larocque D, Charlin L. IG-RL: inductive graph reinforcement learning for massive-scale traffic signal control. *IEEE Trans Intell Transp Syst*, 2022, 23: 7496–7507
- Velickovic P, Cucurull G, Casanova A, et al. Graph attention networks. *Stat*, 2017, 1050: 20
- Wei H, Xu N, Zhang H, et al. CoLight: learning network-level cooperation for traffic signal control. In: *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, 2019. 1913–1922
- Yan L, Zhu L, Song K, et al. Graph cooperation deep reinforcement learning for ecological urban traffic signal control. *Appl Intell*, 2023, 53: 6248–6265
- Zang X, Yao H, Zheng G, et al. MetaLight: value-based meta-reinforcement learning for traffic signal control. In: *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020. 34: 1153–1160
- Wang M, Wu L, Li M, et al. Meta-learning based spatial-temporal graph attention network for traffic signal control. *Knowledge-Based Syst*, 2022, 250: 109166
- Varaiya P. The max-pressure controller for arbitrary networks of signalized intersections. In: *Proceedings of Advances in Dynamic Network Modeling in Complex Transportation Systems*, 2013. 27–66
- Zhang L, Wu Q, Deng J. AttentionLight: rethinking queue length and attention mechanism for traffic signal control. 2021. ArXiv:2201.00006
- Bai C, Wang L, Hao J, et al. Pessimistic value iteration for multi-task data sharing in offline reinforcement learning. *Artif Intelligence*, 2024, 326: 104048
- He H, Bai C, Xu K, et al. Diffusion model is an effective planner and data synthesizer for multi-task reinforcement learning. 2023. ArXiv:2305.18459
- Bai C, Xiao T, Zhu Z, et al. Monotonic quantile network for worst-case offline reinforcement learning. *IEEE Trans Neural Netw Learn Syst*, 2024, 35: 8954–8968

- 26 Xu K, Bai C, Ma X, et al. Cross-domain policy adaptation via value-guided data filtering. 2023. ArXiv:2305.17625
- 27 Hao J, Yang T, Tang H, et al. Exploration in deep reinforcement learning: from single-agent to multiagent domain. *IEEE Trans Neural Netw Learn Syst*, 2024, 35: 8762–8782
- 28 Sun J, Zhang T, Xie X, et al. Stealthy and efficient adversarial attacks against deep reinforcement learning. In: *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020. 5883–5891
- 29 Hao J Y, Shao K, Li K, et al. Research and applications of game intelligence (in Chinese). *Sci Sin Inform*, 2023, 53: 1892–1923
- 30 Li X, Guo Z, Dai X, et al. Deep imitation learning for traffic signal control and operations based on graph convolutional neural networks. In: *Proceedings of the 23rd International Conference on Intelligent Transportation Systems (ITSC)*, 2020. 1–6
- 31 Li L, Lv Y, Wang F Y. Traffic signal timing via deep reinforcement learning. *IEEE CAA J Autom Sin*, 2016, 3: 247–254
- 32 Wei H, Zheng G, Yao H, et al. IntelliLight: a reinforcement learning approach for intelligent traffic light control. In: *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018. 2496–2505
- 33 Chen C, Wei H, Xu N, et al. Toward a thousand lights: decentralized deep reinforcement learning for large-scale traffic signal control. In: *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020. 3414–3421
- 34 Zhang C, Tian Y, Zhang Z, et al. Neighborhood cooperative multiagent reinforcement learning for adaptive traffic signal control in epidemic regions. *IEEE Trans Intell Transp Syst*, 2022, 23: 25157–25168
- 35 Zhang W, Yan C, Li X, et al. Distributed signal control of arterial corridors using multi-agent deep reinforcement learning. *IEEE Trans Intell Transp Syst*, 2023, 24: 178–190
- 36 Fang B, Zheng C, Wang H, et al. Two-stream fused fuzzy deep neural network for multiagent learning. *IEEE Trans Fuzzy Syst*, 2023, 31: 511–520
- 37 Boukerche A, Zhong D, Sun P. A novel reinforcement learning-based cooperative traffic signal system through max-pressure control. *IEEE Trans Veh Technol*, 2022, 71: 1187–1198
- 38 Wu Q, Zhang L, Shen J, et al. Efficient pressure: improving efficiency for signalized intersections. 2021. ArXiv:2112.02336
- 39 Zhang L, Wu Q, Shen J, et al. Expression might be enough: representing pressure and demand for reinforcement learning based traffic signal control. In: *Proceedings of International Conference on Machine Learning*, 2022. 26645–26654
- 40 Bose A J, Jain A, Molino P, et al. Meta-graph: few shot link prediction via meta learning. 2019. ArXiv:1912.09867
- 41 Xu K, Hu W, Leskovec J, et al. How powerful are graph neural networks? 2018. ArXiv:1810.00826
- 42 Huang K, Zitnik M. Graph meta learning via local subgraphs. In: *Proceedings of Advance Neural Information Processing Systems*, 2020. 33: 5862–5874
- 43 Hamilton W, Ying Z, Leskovec J. Inductive representation learning on large graphs. In: *Proceedings of Advance Neural Information Processing Systems*, 2017, 30
- 44 Kingma D P, Ba J. Adam: a method for stochastic optimization. 2014. ArXiv:1412.6980
- 45 Zhang H, Feng S, Liu C, et al. CityFlow: a multi-agent reinforcement learning environment for large scale city traffic scenario. In: *Proceedings of World Wide Web Conference*, 2019. 3620–3624
- 46 Wang M, Zheng D, Ye Z, et al. Deep graph library: a graph-centric, highly-performant package for graph neural networks. 2019. ArXiv:1909.01315
- 47 Koonce P, Rodegerdts L, Lee K, et al. *Traffic Signal Timing Manual*. Technical Report, FHWA-HOP-08-024, Kittelson & Associates, Inc., 2008
- 48 Kipf T N, Welling M. Semi-supervised classification with graph convolutional networks. 2016. ArXiv:1609.02907
- 49 Li B, Tang H, Zheng Y, et al. HyAR: addressing discrete-continuous action reinforcement learning via hybrid action representation. 2021. ArXiv:2109.05490
- 50 Li P, Tang H, Yang T, et al. PMIC: improving multi-agent reinforcement learning with progressive mutual information collaboration. 2022. ArXiv:2203.08553
- 51 Yang Y, Hao J, Liao B, et al. Qatten: a general framework for cooperative multiagent reinforcement learning. 2020. ArXiv:2002.03939