Multi-Agent Transfer Reinforcement Learning With Multi-View Encoder for Adaptive Traffic Signal Control

Hongwei Ge, Dongwan Gao, Liang Sun, Yaqing Hou, *Member, IEEE*, Chao Yu, Yuxin Wang, and Guozhen Tan

***Abstract*— Multi-agent reinforcement learning (MARL) based methods for adaptive traffic signal control (ATSC) have shown promising potentials to solve the heavy traffic problems. The existing MARL methods adopt centralized or distributed strategies. The former only models the environment as an agent and suffers from the exponential growth of action and state space. The latter extends the independent reinforcement learning methods, such as DQN, to multiple interactions directly or propagates information, such as state and policy, without taking their qualities into account. In this paper, we propose a multi-agent transfer reinforcement learning method to enhance the performance of MARL for ATSC, which is termed as multi-agent transfer soft actor-critic with the multi-view encoder (MT-SAC). The MT-SAC combines centralized and distributed strategies. In MT-SAC, we propose a multi-view state encoder and a transfer learning paradigm with guidance. The encoder processes input states from multiple perspectives and uses an attention mechanism to weigh the neighborhood information. While the paradigm enables the agents to handle different conditions for improving generalization abilities by transfer learning. Experimental studies on different scale road networks show that the MT-SAC outperforms the state-of-the-art algorithms and makes the traffic signal controllers more collaborative and robust.**

***Index Terms*— Adaptive traffic signal control, multi-agent reinforcement learning, self-attention, tranfer learning, soft actor critic.**

I. Introduction

With the development of modern cities, the number of vehicles is continuously increasing. The traffic congestions have become a major dilemma that affects the urban economy and advancement. Traffic signal control plays an important role in improving congestions and reducing traffic accidents [1]. Adaptive traffic signal control (ATSC) can dynamically adjust the phases by considering real-time traffic conditions to optimize the traffic of the regional road network and alleviate congestion [2].

There are many methods to control traffic lights. The traditional fixed time methods periodically cycle the established phase settings [3]. They usually lead to insufficient flexibility, long traffic delays, etc. The ATSC was proposed as early as the 20th century, and the representative methods include the SCOOT [4] and the SCATS [5]. They take the information obtained by infrastructure-based detectors as real-time inputs to generate optimal traffic signal control strategies. Many of the traditional ATSC methods still rely on manually designed traffic signal plans. With the flourishing of machine learning, a large number of machine learning based ATSC methods have been proposed, for instance, evolutionary algorithms [6]–[8], neural networks [9]–[11], fuzzy theory [12]–[14].

Traffic signal control is essentially a sequential decision-making process, which is suitable to be solved by Reinforcement learning (RL) [15]. In the past few years, many researchers have shifted to solve ATSC using RL [16], [17]. The RL formulates the problem as a Markov decision process (MDP). It takes the intersection as an agent to maximize the cumulative reward when the agent interacts with the environment. The RL can learn the optimal strategy with less expert knowledge in an unsupervised manner. While applying RL for ATSC, the design of state, action and reward has a great influence on the algorithm. This paper also focuses on how to make full use of the appropriate state information for efficient collaboration.

More recently, many works equip RL with deep neural networks, which are referred to as deep reinforcement learning (DRL). The DRL enables the RL with stronger learning ability. The DRL for ATSC can be roughly divided into three categories, i.e., the value-based, the policy-based and the actor-critic methods. The value-based methods generally parameterize the state-action function, and utilize the experience replay buffer to sample transitions and update themselves through a temporal difference error. This way possesses high sample utilization and is suitable for discrete actions. The policy-based methods directly parameterize the policy, and update themselves by episode returns of the sampled trajectories.

Manuscript received 11 August 2020; revised 16 April 2021; accepted 15 September 2021. Date of publication 1 October 2021; date of current version 9 August 2022. This work was supported in part by the National Key Research and Development Program of China under Grant 2018YFB1600600; in part by the National Natural Science Foundation of China under Grant 61976034, Grant 61572104, and Grant U1808206; and the in part by Dalian Science and Technology Innovation Fund under Grant 2019J12GX035. The Associate Editor for this article was J. Sanchez-Medina. *(Corresponding authors: Hongwei Ge; Liang Sun.)*

Hongwei Ge, Dongwan Gao, Liang Sun, Yaqing Hou, Yuxin Wang, and Guozhen Tan are with the College of Computer Science and Technology, Dalian University of Technology, Dalian 116023, China (e-mail: hwge@dlut.edu.cn; liangsun@dlut.edu.cn).

Chao Yu is with the College of Computer Science and Technology, Sun Yat-sen University, Guangzhou 510275, China.

Digital Object Identifier 10.1109/TITS.2021.3115240

They can be employed in both discrete and continuous action space. Policy-based methods are often on-policy with larger variance than value-based methods. The actor-critic (AC) methods combine the advantages of the value-based and policy-based methods by parameterizing the policy and value networks, respectively. This paper employs an advanced RL algorithm, soft actor-critic (SAC) [18], that uses the concept of maximum entropy to balance the exploration and exploitation.

There are many works applying RL for ATSC under single intersection environment, termed as independent reinforcement learning (IRL). The IRL takes each intersection as an agent [19]–[21], and it is not suitable for the multiple intersection environment in the urban road network and does not take advantage of the characteristics of the multi-intersection environment. There are also some RL methods under multiple intersection environments, which can be roughly classified into the centralized methods [17] and the distributed methods [22]. The centralized methods take global state information as input to obtain a centralized optimal strategy, which treat all intersections as a global agent. It is not free from dimension explosion because of the expansion of intersections and exponentially growth of the action and state space. Furthermore, the distributed methods adopt the decentralized control strategy and formulate the multi-intersection scenario as a multi-agent system (MAS), in which each agent controls a single intersection and only observes parts of the environment. These methods belong to the category of multi-agent reinforcement learning (MARL). The simplest and common approach is to straightly apply IRL to each agent without any interactions. Since each agent regards other agents as parts of the environment, the characteristics of multi-agents are not considered. This will lead to unsatisfactory results. So it is necessary to make improvements derived from IRL to get better results.

In this paper, we propose a multi-agent transfer soft actor-critic with the multi-view state encoder (MT-SAC) for ATSC. MT-SAC combines centralized and distributed strategies. In MT-SAC, a multi-view state encoder and a transfer learning paradigm with cooperative guidance are proposed for further collaboration among agents. The encoder processes input states from multiple perspectives and uses an attention mechanism to weight the neighborhood information. The transfer learning paradigm enables the agent to handle different conditions. The contributions can be summarized as follows:

1) Multi-view state encoder: it is a specifically designed state encoder that represents the multi-view state. The encoder concatenates the encoded local 1D and 2D states of individual intersection and the global attention state of other intersections to construct a multi-view state. Specifically, the encoder encodes the local states by a CNN, and encodes the global state by attention mechanism. The multi-view encoder is designed in such a way so that the appropriate state information can be fully utilized for efficient collaboration.
2) Transfer learning paradigm with cooperative guidance: The paradigm trains individual agents at different intersections for exhibiting different abilities under their confronted states. They serve as specific experts to guide the training of a seed apprentice cooperatively. Then the seed apprentice is transferred to the individual intersections for further training. The above three stages in the paradigm are iterative. Such cooperative guidance and the iterative mechanism make the paradigm being generalized. We validate the effectiveness of the paradigm through extensive experiments.
3) Soft actor critc for ATSC: SAC is an advanced IRL, which works with the maximum entropy to improve the exploring ability of action space. To our knowledge, this is the first work that applies SAC to ATSC.

The rest of the paper is organized as follows. Section II reviews the related work. Section III gives a detailed background. The proposed MT-SAC and the implementation details are given in Section IV. The experiments and discussions are given in Section V, followed by conclusions in Section VI.

II. Related Work

Reinforcement learning is a target-oriented method based on the interactions between the agent and the environment. Deep learning (DL) has strong hierarchical feature extraction capabilities and nonlinear approximation capabilities. The combination of them, i.e., the deep reinforcement learning, has many attractive features, thus the applications of DRL to ATSC have attracted the attentions of many researchers.

Many RL-based methods for ATSC are designed for single intersection scenarios [19], [23]. Specifically, Ref. [24] not only applies RL in ATSC, but also analyzes the influences of the representations of state, action and reward. The DRL methods have also been used in recent years, many of them adopt different variants of DQN [25]–[27]. As the road networks in cities involve multiple intersections. Many methods will face the problem of "curse of dimensionality" in multi-intersection environment. Multi-agent reinforcement learning (MARL) has become a powerful tool for this problem. Generally, the existing MARL methods for ATSC work under the independent control mode or the coordinated control mode.

Under the independent control mode, the individual agents of MARL at each intersection select their optimal action by observing their current local states. Ref. [28] proposes a model-based tabular RL method, and reveals that the RL based adaptive strategy is better than the fixed time non-adaptive strategy. In [29]–[31], each agent controls the traffic signals of an intersection by observing the local state. Ref. [29] uses Q-learning algorithm, while Refs. [30] and [31] use deep Q learning algorithm with replay buffer and target network. The MARLs that adopt independent control mode are effective in small-scale intersection scenarios. However, when they are applied to large-scale multi-intersection scenarios, because each intersection learns and acts simultaneously and there is no coordination among agents, it is difficult for the agents to obtain an optimal joint strategy.

Under the coordinated control mode, the individual agents in MARL work collaboratively by sharing their states, policies, etc. The coordinated control strategy helps the agents to yield the optimal joint strategy. Some works consider traffic conditions of adjacent intersections to extend the input

for collaboration, for examples, Ref. [32] is based on the actor critic method. Besides, some other methods not only consider the states of other agents, but also consider the neighborhood policies to enhance collaboration [33]–[35]. In Ref. [22], the strategies and states of neighborhoods are considered in an individual agent, and a spatial-temporal reward is designed to facilitate training. Meanwhile, Q-value is transferred in the training process to enhance the evaluation ability of the Q-value network [36]. Ref. [37] proposes a collaboration method with traffic flow prediction in which the agents share their traffic information with other adjacent agents. In [38] and [39], the improved tabular Q learning methods are proposed. Besides, for better collaboration of agents, a heuristic neighborhood communication strategy and a max-sum communication strategy are proposed, respectively.

As reviewed above, the MARL methods under the coordinated control mode are more practical than those under the independent control mode. For coordinating MARL, an essential problem raised is how to balance the exploration and exploitation. However, many existing methods have not fully considered this problem. Besides, many methods that prompt collaboration only directly merge or share information without considering the importance of the information.

III. Background

This section introduces the basic concepts of deep reinforcement learning, multi-agent reinforcement learning, problem formulation for ATSC in MARL and transfer learning, which are the foundations of this paper.

A. Deep Reinforcement Learning

Reinforcement learning is a class of algorithms that concerns with how an agent makes sequential actions by observing feedbacks given by an environment. The Markov decision process (MDP) is frequently used to model the RL system, which can be formally represented by a tuple $\langle S, A, R, P, \gamma \rangle$, where S is a finite set of environment states, A is a finite set of available actions, R is a reward function, P is a state transition probability function, and $\gamma \in [0, 1]$ is a discount factor used to adjust the reward, the closer γ is to 1, the more long-term benefits are considered. At each time step t, the agent observes its state $s_t \in S$ and selects an action $a_t \in A$ based on the observation. In return, the agent predicts its next state $s_{t+1} \sim P(s_t, a_t)$ and acquires a reward $r_t \sim R(s_t, a_t)$ from the environment. The objective of an MDP is to find an optimal policy so that the accumulated reward can be maximized.

Generally, the DRL falls into three categories, i.e., the value-based methods, the policy-based methods, and the actor-critic methods.

1) Value-Based Methods: The DQN [40] is the most representative value-based method, which uses neural network to approximate the current state-action value. It adopts two strategies to stabilize the learning process. The first strategy adopts experience replay and the second strategy establishes a target network that copies the value network every T steps. The DQN updates its value network by temporal difference error with sampling transitions from replay buffer $\mathcal{D}$, and the loss is formulated by:

$$\mathcal{L}(\psi) = \mathbb{E}_{(s_t, a_t, r_t, s_{t+1}) \sim \mathcal{D}}[(Q_\psi(s_t, a_t) - y')^2]. \tag{1}$$

In Eq. (1), y' is defined as:

$$y' = r_t + \gamma \max_{a_{t+1}} Q_{\psi^-}(s_{t+1}, a_{t+1}) \tag{2}$$

where ψ and ψ^- are the parameters of the Q-value and the target Q-value networks, respectively. The action is chosen according to the value obtained by the value network. There have been many studies concentrate on adjusting DQN. For example, Ref. [41] adds another value network to alleviate the overestimation problem, and Ref. [42] adds a priority mechanism in the replay buffer for sample efficiency. These methods use $\varepsilon - greedy$ to make a tradeoff between exploration and exploitation.

2) Policy-Based Methods: The REINFORCE is a classic policy-based method [43]. It directly parameterize the policy without ε-$greedy$, because it can naturally explore and can be trained by maximizing the product of likelihood and return. The loss of the policy-based methods is presented in Eq. (3),

$$\mathcal{L}(\theta) = -\mathbb{E}_{\tau \sim \pi_\theta}[\sum_{t=0}^{T} log \pi_\theta(a_t|s_t) G_t] \tag{3}$$

where $G_t = \sum_{k=t}^{T-1} \gamma^{k-t} r_k$ is the discount cumulative return, T is the number of steps of an episode. It is worth noting that the G here is calculated during a period of continuous trajectory, which may bring high variance because of the relevant data. The TRPO [44] and the PPO [45] are popular policy-based methods. They update policies by manipulating the length of updating steps. All of these methods face the sample inefficiency problem.

3) Actor-Critic Methods: The actor-critic methods are combinations of value-based and policy-based methods, which include A2C [46], DDPG [47], TD3 [48]. The rules for updating the policy and value are similar to Eqs. (1) and (3).

$$\pi' = \arg \min_{\pi_k \in \Pi} D_{KL}(\pi_k(\cdot|s_t)||\frac{\exp(\frac{1}{\alpha} Q^\pi(s_t, \cdot))}{Z^\pi(s_t)}) \tag{4}$$

The soft actor-critic (SAC) [18] is a special kind of actor-critic that applies maximum entropy in reward function to prevent the policy from premature converging. Eq. (4) formulates the multimodal policy adopted in SAC, where Z is a partition function, $\alpha > 0$ is a coefficient that adjusts the entropy item and can be determined by backpropagation. The loss function for the policy network can be written as:

$$\mathcal{L}(\theta) = \mathbb{E}_{s_t \sim \mathcal{D}, a_t \sim \pi_\theta}[\alpha \log \pi_\theta(a_t|s_t) - Q_\phi(s_t, a_t)]. \tag{5}$$

The loss for Q network can be written as:

$$\mathcal{L}(\phi) = \mathbb{E}_{(s_t, a_t, s_{t+1}) \sim \mathcal{D}, a_{t+1} \sim \pi_\theta}[\frac{1}{2}(Q_\phi(s_t, a_t) - y')^2] \tag{6}$$

$$y' = r_t + \gamma \mathbb{E}_{s_{t+1}, a_{t+1}}[Q_{\phi_-}(s_{t+1}, a_{t+1}) - \alpha \log(\pi_\theta(a_{t+1}|s_{t+1}))] \tag{7}$$

The loss for α can be written as:

$$\mathcal{L}(\alpha) = -\mathbb{E}_{a_t \sim \pi_t}[\alpha(\log \pi_t(a_t \mid s_t) + \mathcal{H}_0)] \tag{8}$$

where $\mathcal{H}_0$ is a predefined minimum policy entropy threshold.

B. Multi Agent Reinforcement Learning in ATSC

MARL usually expands independent RL with some coordinating mechanism. Some strategies and mechanisms have been studied by establishing the relationship between local Q-value and global Q-value [49], [50] or by passing various information [51], [52] to improve the ability of collaboration. Some works have been studied to apply these ideas to ATSC. If MARL is used without any coordination in ATSC, the learned policy will suffer from partial observability and non-stationary problems. In this paper, we use the states of other agents in the same region to enhance the cooperation among agents.

The ATSC studied in this paper is formulated as follows. The road network is modeled by using a graph $G(V, E)$, where V and E are sets of nodes and edges respectively, and $v_i \in V$ is an intersection which is controlled by an agent, $e_{ij} \in E$ is the lane connecting v_i and v_j. Then we model the problem as a networked decentralized partially observable Markov decision process (NDec-POMDP), which can be represented by a tuple $\langle G, N, S, O, A, R, P, \gamma \rangle$, where:

- G is the graph of the road network.
- $n_i \in N$ is the set of agents which are in the observable region of v_i.
- $s_i \in S$ represents the state information of v_i.
- $o_i = O(s_i, s_{-i})$ is the observation of v_i.
- $a_i \in A$ is the action taken by v_i under o_i.
- $r_i = R(s_i, a_i)$ is the local reward of v_i, which is the feedback from environment.
- P and γ are the same with those in MDP.

The goal of NDec-POMDP is to maximize the global reward as the optimization. We assume that the global reward is decomposable, $r_{global,t} = \sum_{i=1}^{N} r_{i,t}$. The designs of S, A, R will be introduced in Section IV.D.

C. Transfer Learning

The transfer learning [53] involves a source domain and a target domain. The knowledge learned from the source domain can be transferred to the target domain to improve the learning performance. If the tasks in the source domain and target domain are similar, the process of knowledge transfer is relatively easy without considering extra differences in distribution or adaptation.

For multi-intersection road networks, the traffic flows at the intersections are different, which implies that the control strategies of traffic lights at different interactions should be different. So transfer learning can be combined with RL to construct an adaptive traffic signal controller that can adapt to complex road conditions. In transfer reinforcement learning, two strategies are usually adopted. One is transferring the parameters of learned models and the other is sharing the transitions which are produced by previously trained agents. This paper adopts the first kind of strategies to transfer information with cooperative guidance.

IV. Multi-Agent Transfer Soft Actor-Critic With the Multi-View Encoder for ATSC

In the same region of a road network, each intersection can be modeled as an agent to construct a multi-agent environment, then we can use MARL to realize cooperative control. This section proposes a multi-agent transfer reinforcement learning to enhance cooperation, which is termed as multi-agent transfer soft actor-critic with multi-view encoder (MT-SAC).

We propose a multi-view encoder and a transfer learning paradigm with cooperative guidance to stabilize and enhance the learning process. The multi-view encoder processes input state from multiple perspectives and uses attention mechanism to weight the neighborhood information. The transfer learning paradigm with cooperative guidance is motivated by the idea that many experts with different abilities can teach apprentices together and then guide them to learn independently and continuously. The paradigm is formulated in such a way so as to make the final learned agents being more comprehensive.

A. Multi-View State Encoder

As suggested by [24], the definitions of the state, action, and reward are essential for the performance of RL in ATSC. An appropriate state can provide useful information to the learning process for RL agents.

To describe the local state of the intersections, we consider two definitions, i.e., the 1D state and the 2D state. The 1D state adopts a one-dimensional vector with each element which records the data of vehicles at intersections, such as queue length, number of vehicles, waiting time or speed of vehicles. Agents can learn from the fully connected layers (FC) which takes the 1D state as input. The 2D state is extracted from the real images captured by the camera or extracted from the position and speed data collected by road sensors. It contains the spatial information that is not included in the 1D state. Since the 2D state is stored in matrix, we can use the convolutional neural network [54] to learn the lattice features.

Both the 1D state and the 2D state are at the local level. To enable the whole system to learn an better strategy, we construct a global level state by attention mechanism. The attention mechanism can extract effective features from key parts to obtain important information. This paper proposes a self-attention mechanism to preprocess the neighborhood state information because it is differentiable and easy to train in an end-to-end way. Thus, we can evaluate the neighborhood information to get a global attention state for enriching the inputs and improving the acquisition of effective information. The self-attention works as follows: 1) construct three trainable transfer matrices that are embedded in the network; 2) multiply them with the same input S to get Query (Q), Key (K) and Value (V); 3) calculate the attention coefficient w_i by:

$$w_i = softmax(\frac{Q_i K^T}{\sqrt{d_k}}) \tag{9}$$

where d_k is used to compress the product and the attention state $s_{att,i}$ is calculated by:

$$s_{att,i} = w_i V \tag{10}$$

The above procedure integrates the neighborhood information according to the importance. This forms a global level state.

By integrating the different states at the local level and global level, a multi-view state encoder is constructed,

Algorithm 1 MT-SAC

Initialize $\{\theta_i, \phi_i, \alpha_i, \eta_{\theta,i}, \eta_{\phi,i}, \eta_{\alpha,i}\}$ $for\ i \in I$
if It is the third stage **then**
  Trasfer above parameters from seed agent
end if
Initialize $D, \gamma, \tau, \varepsilon = 0, \{\phi_i^-\} \leftarrow \{\phi_i\}$ $for\ i \in I$
for episode $= 1 \rightarrow$ MAX_EPISODES **do**
  for step $= 1 \rightarrow$ MAX_STEPS **do**
    get $S_t = \{s_{t,i}\}_{i \in I}$ from env
    $\varepsilon = min\{0.9 \cdot (episode/threshold\_episode), 0.9\}$
    get $O_t = \{o_{t,i}\} = encoder_i(S_t)$
    get $A_t = \{a_{t,i}\}_{i \in I}$ from π_i with probability ε
    get $S_t' = \{s_{t,i}'\}$, $R_t = \{r_{t,i}\}$ after act A_t in env
    store (S_t, A_t, R_t, S_t') in D
    if $|D| \geq batch\_size$ **and** $step\ \%\ update\_gap = 0$ **then**
      for $i \in I$ **do**
        if It is the second stage **then**
          $\phi_i = \phi_i - \eta_{\phi,i} \nabla \mathcal{L}(\phi_i)$ //Eq.12 and Eq. 14
        else
          $\phi_i = \phi_i - \eta_{\phi,i} \nabla \mathcal{L}(\phi_i)$ //Eq.12 and Eq. 7
          $\phi_i^- = \tau \phi_i^- + (1-\tau)\phi_i$
        end if
        $\theta_i = \theta_i - \eta_{\theta,i} \nabla \mathcal{L}(\theta_i)$ //Eq. 11
        $\alpha_i = \alpha_i - \eta_{\alpha,i} \nabla \mathcal{L}(\alpha_i)$ //Eq. 13
      end for
    end if
    $step \leftarrow step + 1$
  end for
  $episode \leftarrow episode + 1$
end for

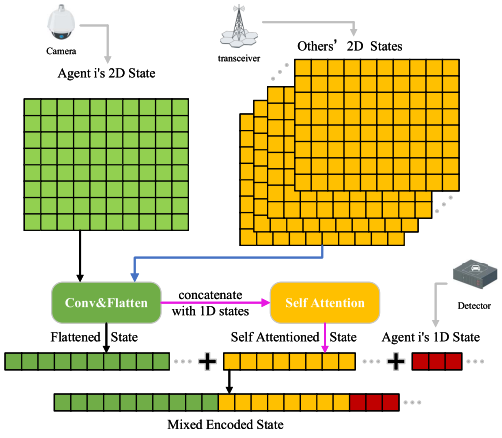


Fig. 1. The diagram of the Multi-view State Encoder. A multi-view encoded state is generated by concatenating the 1D state, encoded the 2D state, and the encoded attention state. While encoding, it employs the CNN to encode the 2D state and employs the attention matrices to encode the states of other agents. Then this mixed encoded state is used as the model input. The blue line means the data flow of 2D state of the other agents and the pink line means the generation process of the attention state. These encoded states are concatenated with the corresponding 1D state to calculate the attention states.

as shown in Fig. 1. Each agent has an encoder. The encoder generates encoded 2D states with CNN using all states of the agents in the observable region, then it flattens the encoded states to vectors. These vectors are concatenated with 1D states to calculate the attention state. Finally, the encoder takes 1D, encoded 2D, and the attention state of the current agent as the input. It not only contains its own basic and implicit state but also holds the states of other intersections. This will make the agents learn better cooperative strategies.

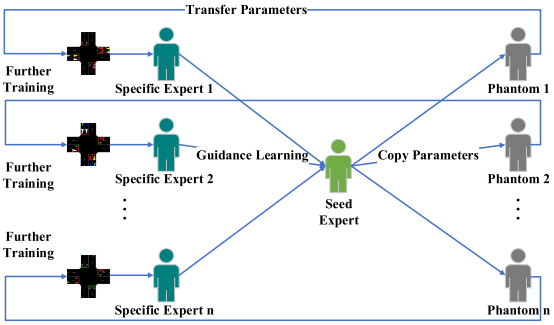


Fig. 2. The transfer learning paradigm with cooperative guidance.

B. Transfer Learning Paradigm With Cooperative Guidance

Human adaptability relies crucially on learning and merging knowledge: the teachers point out some important concepts and the apprentices fill in the gaps of their own. This is particularly effective, because possible scenarios can never be exhaustive. Motivated by the perspective, this section proposes the transfer learning paradigm with cooperative guidance.

In multi-intersection scenario, the road conditions of each intersection are generally different. Even if the agent obtains the state information of the neighbor agent, the agent can only improve the ability to deal with the situation of its corresponding intersection. It is necessary for an agent that can handle the road conditions never encountered. Moreover, it should learn adaptively to discover characteristics and regularities that help to generalize in different intersections.

Based on the above ideas, we design the paradigm with three stages. Fig. 2 presents the flowchart of the transfer learning paradigm with cooperative guidance. The first stage involves training a set of individual agents at different intersections for exhibiting different abilities under their confronted state. We refer the trained individuals as specific experts. They are specifically trained to deal with their corresponding intersections. The second stage involves training a seed apprentice with the guidance of the specific experts. Since the specific experts trained in the first stage may locate in the local optimum of the parameter space, the cooperative guidance of them provides opportunities to make the seed apprentice locate in a more appropriate position of the parameter space. The obtained seed apprentice possesses a strong generalization ability and can deal with different road situations. In the third stage, the seed apprentice is copied to the phantom experts. Then they are transferred to the different intersections for further adaptive training so that they have the ability of adapting the traffic characteristics and regularities at corresponding intersections.

The above process is iterative. The trained agents in the third stage can also be used as experts for the next iteration. Especially, the $\varepsilon - greedy$ can be used to make the learned

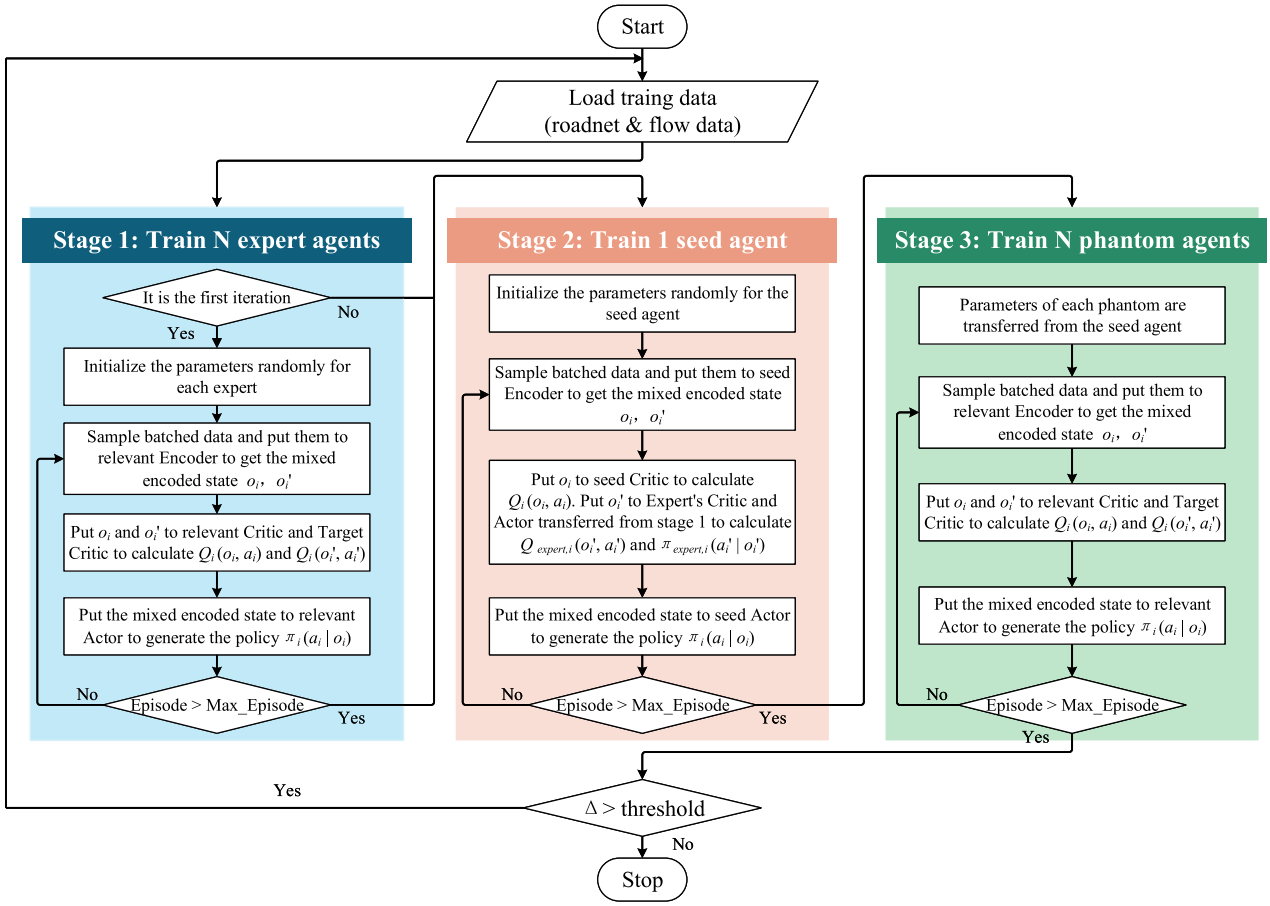


Fig. 3. The overall framework of MT-SAC. In the subsequent iterations, the experts of stage 1 are replaced by the phantoms in the previous iteration.

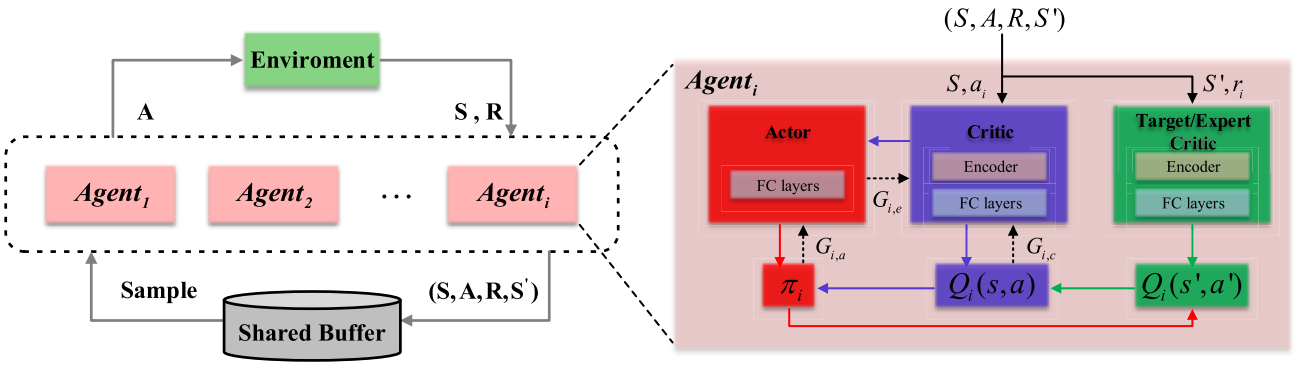


Fig. 4. The diagram of the training process of MT-SAC. The left part is the details of the training at each stage, and the right part is the details of the training for an individual agent. Each iteration has three stage, the target critic of the seed is replaced by the critics of the expert agents in the second stage, while the target critics in other stages are the copy of their own critics.

strategies being more exploratory. The paradigm can help to adapt and generalize in different intersections.

C. Multi-Agent Transfer SAC With Multi-View Encoder

The MT-SAC operates by integrating SAC with the multi-view state encoder and the transfer learning paradigm. The training has three stages. The first stage involves training the individual expert agents. The second stage involves training the seed agent with the expert guidance. In the third stage, the seed agent is copied to each intersection for further learning.

Fig. 3 illustrates the framework of the MT-SAC and Algorithm 1 illustrates the updating process of MT-SAC during the three stages in one iteration. All the agents in the three stages are modelled using SAC, which includes a policy network, two Q-value networks and a trainable parameter α. The input of the agent is set as $o_i = encoder(s_{2d,i}, s_{1d,i}, s_{-i})$,

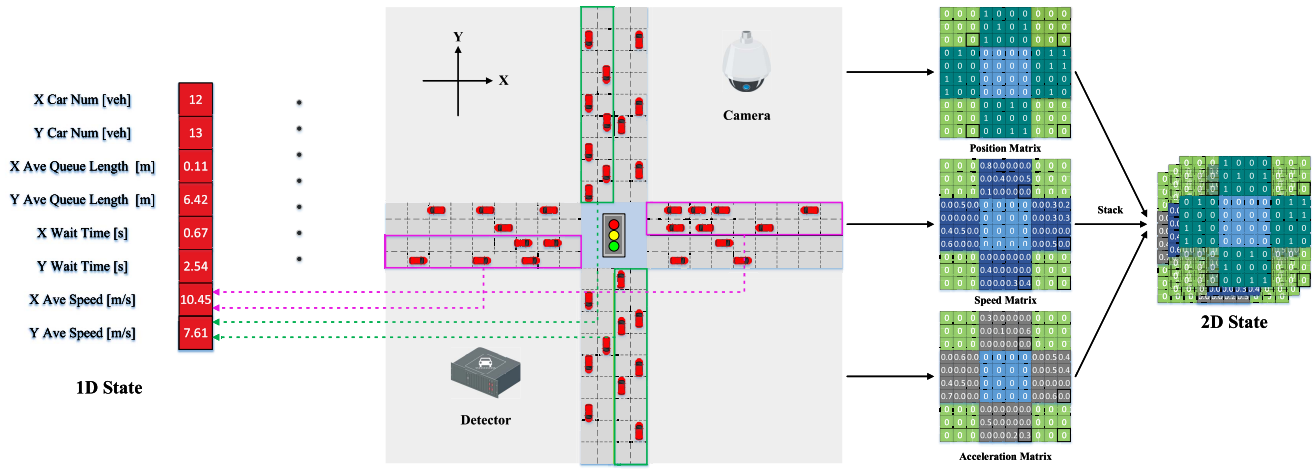


Fig. 5. The illustration of 1D and 2D states. The value of each item in the speed and acceleration matrix is the ratio of the vehicle speed versus the maximum road speed and the vehicle acceleration versus the maximum road acceleration.

where s_{-i} means the states taken from other agents. It is calculated by the multi-view state encoder. We note that all agents share a global replay buffer and receive the local reward $r_{t,i}$.

In the first stage, the networks are initialized to generate global transitions (S, A, R, S') during the learning process. During the training process, the $agent_i$ updates the $actor_i$, the $critic_i$, and the parameter α_i by Eq. (11), Eq. (12) and Eq. (13) respectively, where y_i' is the same with that in Eq. (7) and $o_{t,i}$ is generated by the encoder with sampled S_t. The batch data is sampled from the global replay buffer. The learning process and the network structure are presented in Fig. 4.

$$\mathcal{L}(\theta_i) = \mathbb{E}_{S_t\sim\mathcal{D}, a_{t,i}\sim\pi_{\theta,i}} \times [\alpha_i \log \pi_{\theta,i}(a_{t,i}|o_{t,i}) - Q_{\phi,i}(o_{t,i}, a_{t,i})] \quad (11)$$

$$\mathcal{L}(\phi_i) = \mathbb{E}_{(S_t,a_{t,i},S_{t+1})\sim\mathcal{D}, a_{t+1,i}\sim\pi_{\theta,i}} \times [\frac{1}{2}(Q_{\phi_i}(o_{t,i}, a_{t,i}) - y_i')^2] \quad (12)$$

$$\mathcal{L}(\alpha_i) = -\mathbb{E}_{a_{t,i}\sim\pi_{t,i}}[\alpha_i(\log \pi_{t,i}(a_{t,i} \mid o_{t,i}) + \mathcal{H}_0)] \quad (13)$$

In the second stage, the MT-SAC firstly initializes a seed agent termed as seed apprentice whose action can be selected based on the road conditions of other intersections. Under this scheme, all intersections share the same seed network. The MT-SAC provides a solution to guide the agent for the Actor-Critic based RL, in which the actor learns a strategy and the critic learns how to evaluate the state. While training, the actor is more affected by the critic. So the critic plays a more important role than the actor. Besides, the parameters of the critic are updated by a mean square error (MSE). Thus in the training process, we use the critics of experts to guide the critic of seed, and the seed is trained by Eq. (12) and Eq. (14).

$$y_i' = r_{t,i} + \gamma\, \mathbb{E}_{o_{t+1,i}, a_{t+1,i}}[Q_{\phi_{expert,i}}(o_{t+1,i}, a_{t+1,i}) - \alpha_i \log(\pi_{\theta_{expert,i}}(a_{t+1,i}|o_{t+1,i}))] \quad (14)$$

The training in the third stage also adopts the similar framework with the first stage. The seed is transferred to each intersection. After that the intersections are further trained to adapt their own road conditions. The above operations form one iteration, then we calculate the average vehicle number of the last 10 episodes in the first stage and the third stage, resulting in $\Delta = |car_{third} - car_{first}|/car_{first}$. If Δ is larger than the threshold K, then we treat the phantom agents as the expert agents in the next iteration. The above process repeats until the stop criterion is met.

Convergence Analysis: For SAC algorithm, we have the soft policy iteration theorem [18], which clarifies that the repeated application of soft policy evaluation and soft policy improvement from any $\pi \in \Pi$ converges to a policy π^* such that $Q^{\pi^*}(s_t, a_t) \geq Q^{\pi}(s_t, a_t)$ for all $\pi \in \Pi$ and $(s_t, a_t) \in S \times A$, assuming $|A| < \infty$.

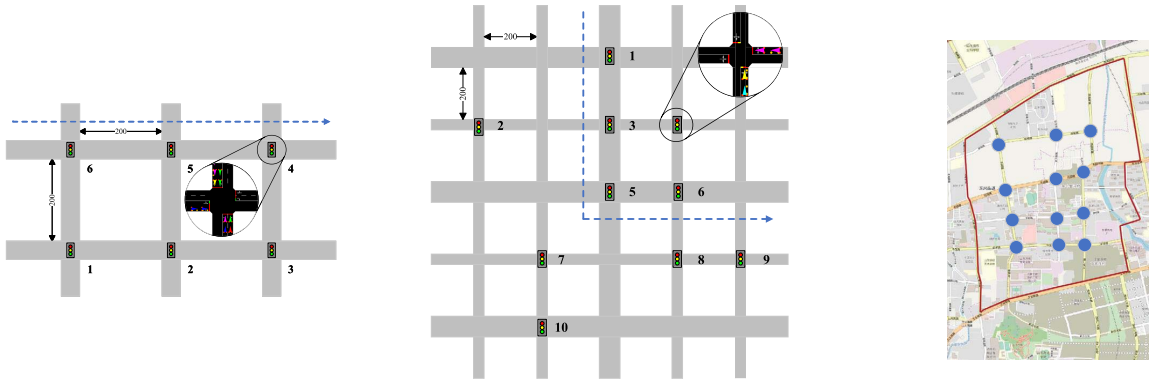
The MT-SAC involves 3 stages. In the first stage, the training is formulated in a decentralized way. Each agent takes the states of its neighbors in the input vector, and the training is the same as that of SAC. Definitely, the volume of action set of the agent satisfies $|A| < \infty$. According to the soft policy iteration theorem, it can guarantee convergence. In the second stage, the training of the critic is carried out in a supervisory way, so it will converge as long as the training of the first stage converge. In the third stage, the working process can be considered as fine-tuning on the parameters of the second stage. In summary, according to the soft policy iteration theorem, the entire algorithm can guarantee convergence.

D. Details of NDec-POMDP

For applying MT-SAC in ATSC, the problem is modeled as NDec-POMDP. We simulate the multi-intersection environment using the traffic simulator SUMO 1.8.0 [55]. The state, action, and reward are defined as follows.

1) State: 1D and 2D state representations are required for multi-view state encoder. The lanes that connect the intersections are categorized into horizontal (X) lanes and vertical (Y) lanes. Fig. 5 presents an illustration of 1D and 2D states.

To construct 1D state, we consider the incoming lanes in X and Y directions of the intersection. The car number, average



(a) (b) (c)

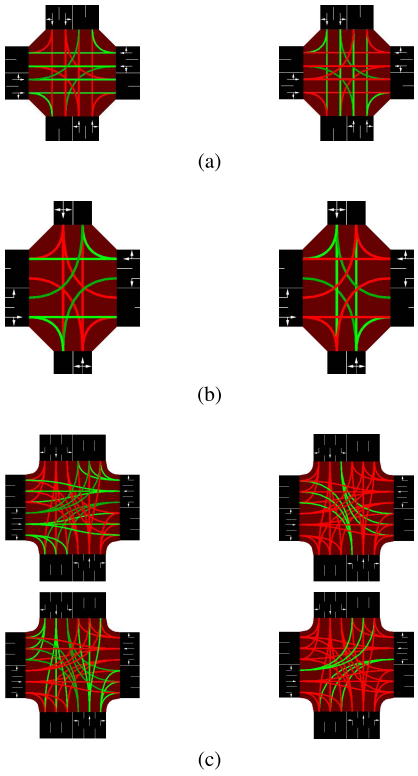
Fig. 6. The multi intersections environments: (a) shows the small-scale scenario with 6 intersections; (b) shows the large-scale scenario with 25 intersections; (c) shows the Dongfeng Sub-district of Jinan in China scenario with 12 intersections.

queue length, average waiting time, and average speed of vehicles are employed. The elements are assumed to be taken from the sensors deployed in the intersection.

To construct 2D state, we consider the snapshot of the intersection. The snapshot is divided into $H \times W$ grids, and the width of each grid is the width of a lane, and the length of each grid approximates the average length of the vehicles. For the $H \times W$ grids, we employ 3 matrices, which records the position, speed, and acceleration of vehicles, respectively. For each grid, if there is a vehicle, the corresponding element in the position matrix is set as 1, the corresponding element in the speed matrix is set as the ratio of speed to maximum speed, and the corresponding element in the acceleration matrix is set as the ration of acceleration to maximum acceleration. If there isn't any vehicle, the corresponding elements in the three matrices are set as 0. The 2D state is obtained by stacking the three matrices into a $H \times W \times 3$ tensor.

2) Action: We employ different phases as actions of agents. Here, we define the same action space for each individual intersection as those defined in SUMO.

Fig. 6 shows three multi-intersecion scenarios. Fig. 7 shows illustraive examples of the actions in different scenarios, where the green lines represent the routes that allow the vehicles to pass by, and the red lines represent the routes that forbid the vehicles to pass by. The actions of each agent are not exactly the same, but we limit the number of the actions to be the same. We require the actions with the same index have similar meanings and don't require they have identical content. For example, in a large-scale scenario illustrated in Fig. 6 (b), the intersection are heterogeneous. We adjust the action space of each intersection to $\{0, 1\}$. $Agent_5$ and $Agent_6$ have the topologies like Fig. 7 (a) and Fig. 7 (b), the actions of $Agent_5$ are "rrrrGGGgrrrrGGGg" and "GGGgrrrrGGGgrrrr" whose length is 16, and the actions of $Agent_6$ are "rrrGGgrrrGGg" and "GGgrrrGGgrrr" whose length is 12. It can be seen that the actions of $Agent_5$ and $Agent_6$ are not identical, but the actions with index 0 have similar meanings, they are used to adjust the traffic flow along the X direction and X→Y direction. The actions with index 1 adjust the traffic flow along the Y direction and Y→X direction.



(a) (b) (c)

Fig. 7. The action space of the three scenarios. (a) illustrates the actions of the agent which has 16 lanes in small-scale and large-scale scenaries. (b) shows the actions of the agent which has 12 lanes in large-scale scenario, while real-world scenario adopts the actions shown in (c).

3) Reward: The optimization objective at each step is to maximize the global reward, we assume that the global reward is decomposable, $r_{global,t} = \sum_{i=1}^{N} r_{i,t}$. As far as the rewarding scheme concerned, the following four optimization objectives in the incoming lanes of the intersection are considered:

- The total number of vehicles (car).
- The average speed of vehicles (speed).
- The average lengths of all vehicles whose speed is less than 0.1m/s (queue).
- The average waiting time of all vehicles whose speed is less than 0.1m/s (wait).

In summary, the reward funtion is formulated as follows:

$$\mathrm{w}_1 \cdot speed - \mathrm{w}_2 \cdot queue - \mathrm{w}_3 \cdot car - \mathrm{w}_4 \cdot wait \quad (15)$$

where $W = \{\mathrm{w}_1, \mathrm{w}_2, \mathrm{w}_3, \mathrm{w}_4\}$ mean the weights of these items. The weighted sum of these four objectives is taken as the final objective of the proposed algorithm. The summarization can be considered as a normalization method. The reason why we adopt the weighted summarization strategy is that the magnitudes of these four items are different in the early episodes for each environment. If they are added directly as a reward, the agent will only focus on the items with large magnitudes and make the algorithm fail to converge.

V. Experiments

To validate the performance of the MT-SAC and analyze its components, the experiments are carried out on three scenarios as shown in Fig. 6: a small-scale synthetic scenario with 2×3 homogeneous intersections, a large-scale synthetic scenario with 5×5 heterogeneous intersections, and a real-world scenario with 12 homogeneous intersections. We firstly carry out evaluation experiments, and compare the results with the state-of-the-art algorithms. The objective is to demonstrate the overall performance of the MT-SAC. Then we carry out ablation experiments. The objective is to demonstrate the effects played by the proposed multi-view encoder and the transfer learning mechanism. Finally, we give the complexity analysis of the MT-SAC.

A. Algorithms for Comparisons

To demonstrate the performance of the MT-SAC, we include the following algorithms for comparisons:

1) IDQN [30]: IDQN provide a solution to apply the DQN in a multi-intersection system for IRL training.
2) MA2C [22]: MA2C is a relatively new and effective distributed MARL algorithm that adopts the Actor-Critic framework. It considers the spatial discount factor to enhance the vision of the agent and adds neighbors' policies to enrich the local state.
3) Independent SAC (ISAC) [18]: We adopt the naive SAC in each of the interactions without the shared neighbor information and transfer learning method.
4) MPC [56]: It is a model predictive control algorithm for discrete action space. We use a deep neural network to approximate the dynamic function of the environment. And we take the RandomOptimizer as the optimizer of MPC.

Besides, we additionally add FIXED method which is based on the fixed time rotation in SUMO.

B. Implementation Details

In MT-SAC, each agent has three networks: actor, critic, and target critic. We note that the target critic uses the soft update method. The encoder is combined into the critic so that it can be trained while the algorithm updating critic. It should be noted that the target network in Seed Apprentice is replaced by the critic of specific experts successively. The threshold K=10%. The length of the 1D state is 8, the 2D state size is $32 \times 32 \times 3$, and the length of each grid is 5m.

TABLE I
Network Structure of the Encoder

Encoder
Conv(3,4,4,2)
Conv(4,8,2,2)
Pool(2,2)
Linear(128,64)
Linear(64,24)
Q_linear(32,16)
K_linear(32,16)
V_linear(32,16)

TABLE II
Network Structure of the MT-SAC

	MT-SAC
Critic	Encoder
	Linear(48,16)
	Linear(16,16)
	Linear(16,a_dim)
Actor	Linear(48,16)
	Linear(16,16)
	Linear(16,a_dim)
Alpha	α

The structures of the encoder and the MT-SAC are shown in Table I and Table II. The parameter a_dim represents the dimension of the action space. The Relu function is used for activation function in all algorithms. For efficient learning, the Adam function is used as the gradient optimizer.

For IDQN, it adopts the same structure of the critic of the MT-SAC. For ISAC, it adopts the same structure of the actor and critic of the MT-SAC. The encoders in IDQN and ISAC do not include the attention layers. Besides, we find MA2C with CNN can not converge, so we use two networks with four FC layers to represent actor and critic respectively. State and action space are the same for these algorithms. For MPC, the dynamic function is a network with four FC layers. And the predictive horizon of MPC is 6. The MT-SAC and the compared algorithms use the same dataset for training and the same dataset for testing. Both the data are generated randomly in synthetic scenarios, while the real-world data are taken from Dongfeng sub-district of Jinan.[1]

To provide a fair comparison, the MT-SAC, IDQN, MA2C and ISAC use the same training settings. During training, each episode has 1800 steps, and each step stands for one second. Each decision-making cycle T has an interval of 6 seconds. If T is set too short, for example, 1s, the traffic lights may change frequently, which is unreasonable and will increase the computational burden. If T is set too long, the learning speed of the agent will become slow. All algorithms are trained for 150 episodes. Such setting implies that the agents will be

[1] https://traffic-signal-control.github.io/#open-datasets

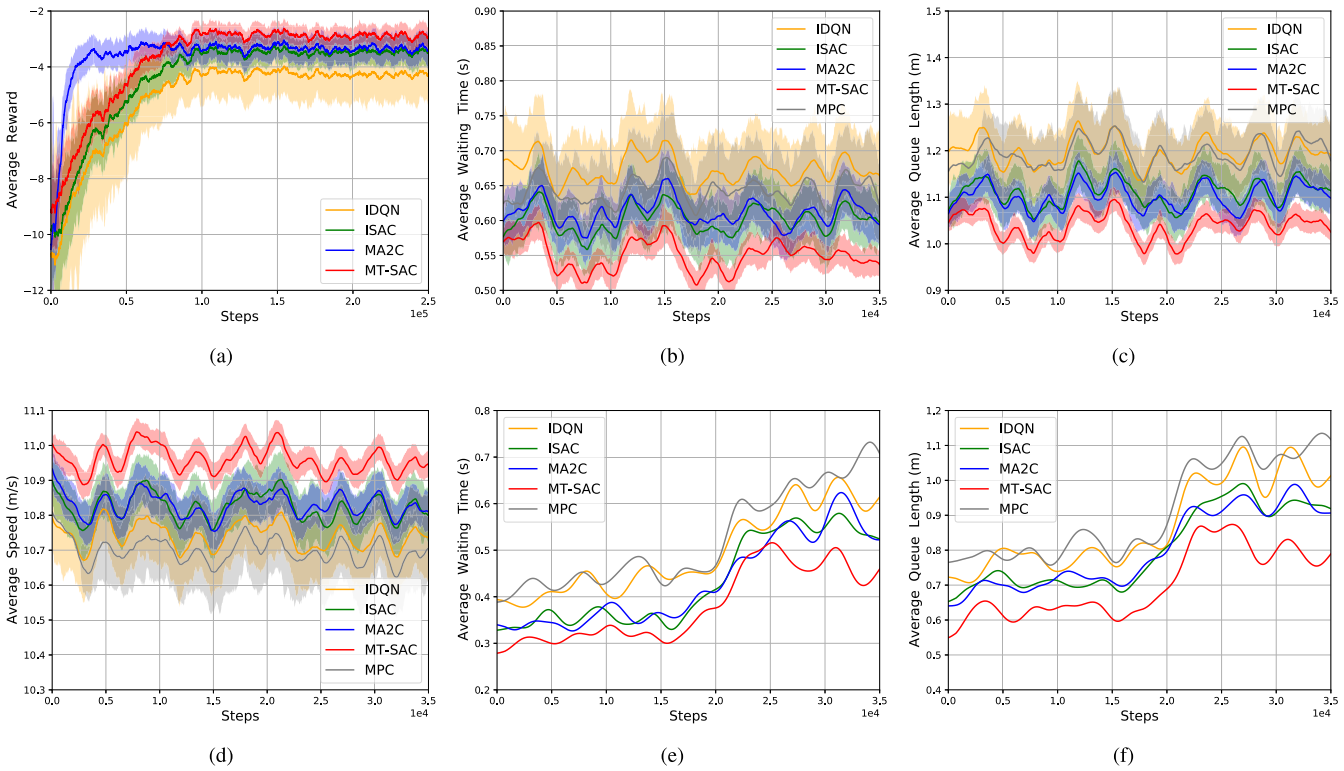


Fig. 8. The result curves of the four algorithms on the small-scale scenario. (a) represents the training curves; (b), (c), and (d) represent the average waiting time, average queue length, and average speed of the vehicles in the traffic network. (e) and (f) represent the average waiting time and average queue length curves of the vehicles of $Agent_5$ in an extra test where recurring congestion happens at the 18000th step.

trained 300 (1800/6) times in one episode. Besides, the first 5 episodes are used to generate the data for replay buffer for off-policy strategy. We note that the seed agent in the second stage of an iteration is trained 100 times in one episode for avoiding overfitting. The MT-SAC, IDQN and ISAC adopt the off-policy training strategy and MA2C adopts the on-policy training strategy. For all algorithms, the random seed for initializing the network is set the same.

The hyper-parameters used in the MT-SAC and the compared algorithms are selected through trial and test, and the configurations listed in Table III yield the best performance. More specifically, lr and γ are used in all RL-based algorithms, where 1e-4 and 1e-8 are adopted in the synthetic scenarios and real-world scenario respectively. The parameter sp is used in MA2C. The parameters τ, ϵ, batch and buffer are used in off-policy algorithms. The parameter ε ranges from 0 to 0.9, and it maintains 0.9 at the 65th episode, which means that the policy is randomly generated with a probability of 10%. During evaluation process, ε is set to 1. The training and evaluation rounds of all algorithms are taken as 5 rounds.

TABLE III

HYPERPARAMETERS SETTING OF MT-SAC AND COMPARED ALGORITHMS

parameters	description	setting
lr	learning rate	1e-4, 8e-4
γ	reward discount factor	0.99
sp	spatial discount factor	0.25
τ	coefficient for updating target network	1e-2
ϵ	exploration coefficient	$0 \sim 0.9$
batch	batch size	16
buffer	replay buffer size	10000

C. Evaluation Experiments on Small-Scale Scenario

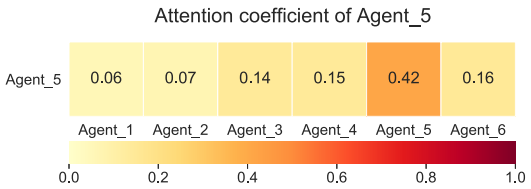
1) Detailed Settings: As shown in Fig. 6 (a), in the 2×3 synthetic scenario, each intersection has 16 lanes, with the speed limitation of 14m/s and fixed distance of 200m. We take the first 70m to construct the state matrix. The intersections are controlled by homogeneous agents with 2 actions. The vehicles have the same length of 4m and have the same acceleration and deceleration. Basic traffic density is assumed to be 7000veh/h. The weights for rewards are taken as $W = \{0.5, 0.5, 2, 0.5\}$.

2) Training Results: The training curves of IDQN, ISAC, MA2C and MT-SAC are shown in Fig. 8 (a), where the solid line represents the average value and the shadow area represents the standard deviation. MPC only trains dynamic function networks, and it does not have training curve, so it is not included in this figure. It can be seen that the curves of all RL-based methods become flat after certain numbers of episodes. Among the 4 algorithms, the MT-SAC yields the highest reward, the MA2C, ISAC yield similar reward, and the IDQN yields the lowest reward. The MA2C converges the fastest. The results show that the on policy methods converges faster than the off-policy methods.

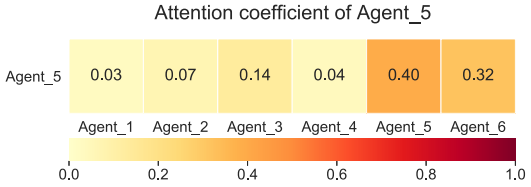
3) Evaluation Results: The objective of the MT-SAC involves optimizing a global reward, which concerns the number of vehicles, the average queue length, the average

TABLE IV
COMPARISONS OF DIFFERENT ALGORITHMS IN THREE SCENARIOS

Metrics	Small-scale scenario						Large-scale scenario					Real-world scenario		
	FIXED	IDQN	ISAC	MA2C	MT-SAC	MPC	FIXED	ISAC	MA2C	MT-SAC	MPC	FIXED	MT-SAC	MPC
sum.car [veh]	21.53	14.51	12.40	12.14	**12.02**	14.62	12.40	9.18	9.95	**8.48**	13.82	29.45	**24.02**	27.21
ave.queue [m]	5.11	1.19	1.11	1.09	**1.04**	1.19	5.00	0.52	0.61	**0.42**	0.70	2.59	**0.56**	1.75
ave.speed [m/s]	8.17	10.75	10.85	10.86	**10.96**	10.70	10.98	12.98	12.87	**13.04**	12.70	8.79	**9.89**	9.30
ave.wait [s]	15.19	0.67	0.60	0.61	**0.55**	0.64	16.58	0.35	0.51	**0.27**	0.43	7.72	**0.76**	4.42



(a) Before recurring congestion



(b) After recurring congestion

Fig. 9. The attention coefficient of $Agent_5$ before and after recurring congestion in the small-scale scenario.

speed, and average waiting time in the incoming lanes of the intersections. The results are presented in Table IV. It can be seen that the performance of the FIXED method is worse than other methods. The reason is that the fixed time method requires a lot of expert experience to obtain good results. The performance of MPC with a low training cost is similar to that of IDQN. The MT-SAC performs the best among these algorithms.

Fig. 8 (b), (c), and (d) plot the curves of average waiting time, average queue length, and average speed of the vehicles in the traffic network. From the figures, it can be observed that:

- The curves for average waiting time, average queue length of the MT-SAC locate below the curves of other algorithms, and the curve for average speed of the MT-SAC locates above the curves of other algorithms.
- The curves obtained by IDQN and MPC fluctuate with larger volatility.
- The ISAC and MA2C yield similar performance. However, the volatility of MA2C is smaller than that of ISAC.

The above results demonstrate the effectiveness of the proposed MT-SAC in a simple but high-density scenario. On the other hand, the IDQN does not have interaction mechanism and does not consider the dynamics of the environment; the MPC relies on dynamic model; the MA2C considers the neighbor policies and rewards, thus can be regarded as an interaction mechanism to some extent.

In the experiment, the training of the MT-SAC and the compared algorithms repeat for 5 rounds. For each algorithm, we select the trained model that yields the best results for further extending of current experiment. In the extended experiment, we consider the same scenario with a recurring congestion occurs at the intersection $Agent_5$. The blue dotted line in Fig. 6 (a) shows the recurring congestion traffic flow. Fig. 8 (e) and (f) show the curves of average waiting time, and average queue length in $Agent_5$ obtained by the MT-SAC and the compared algorithms. It can be seen that the curves of MT-SAC have downward trend, while other algorithms fail to deal with the recurring congestion efficiently.

To demonstrate the cooperative behavior incurred by the MT-SAC, we present the attention coefficients of $Agent_5$ at the time before and after congestion in Fig. 9. The coefficients are averaged over 900 steps. Before congestion, $Agent_5$ pays more attention to $Agent_3$, $Agent_4$, and $Agent_6$ than others. After congestion, $Agent_5$ pays more attention to $Agent_6$ and pays less attention to $Agent_4$. The reason is that the recurring congestion traffic flow passes through $Agent_6$, $Agent_5$, and $Agent_4$ successively, and $Agent_6$ is the upstream of $Agent_5$. The results demonstrate that the attention state obtained by the multi-view encoder has positive effect on the policies of agents, and demonstrate that the agents will take the states of their surrounding agents into consideration when making their own decisions. They will pay more attention to the agents that have greater impact on themselves. This is a cooperative behavior and will yield satisfactory results.

D. Evaluation Experiments on Large-Scale Scenario

1) Detailed Settings: As shown in Fig. 6 (b), in the 5×5 scenario, the intersections have different topologies, which are controlled by the agents with 2 actions. The heterogeneous intersections have 8, 12, or 16 lanes. For the streets with 2 lanes or 4 lanes, the speed limits are 11m/s or 20m/s, respectively. The vehicles have the same length of 4m and have different acceleration and deceleration. Compared with the small-scale scenario, the intersections cover a wider range, but the traffic capacity of each single intersection becomes smaller. If there are too many vehicles in the region of the intersection with 8 lanes, the influence will spread to other intersections, and will lead to uncontrollable congestion.

We randomly select 10 intersections for RL-controlled agents, so their distances are different. Considering the number of agents and road capacity, the traffic density is set as 4200veh/h, and the reward weights are set as $W = \{0.5, 0.5, 1.5, 0.1\}$. Other settings are the same as the setting of the small-scale scenario.

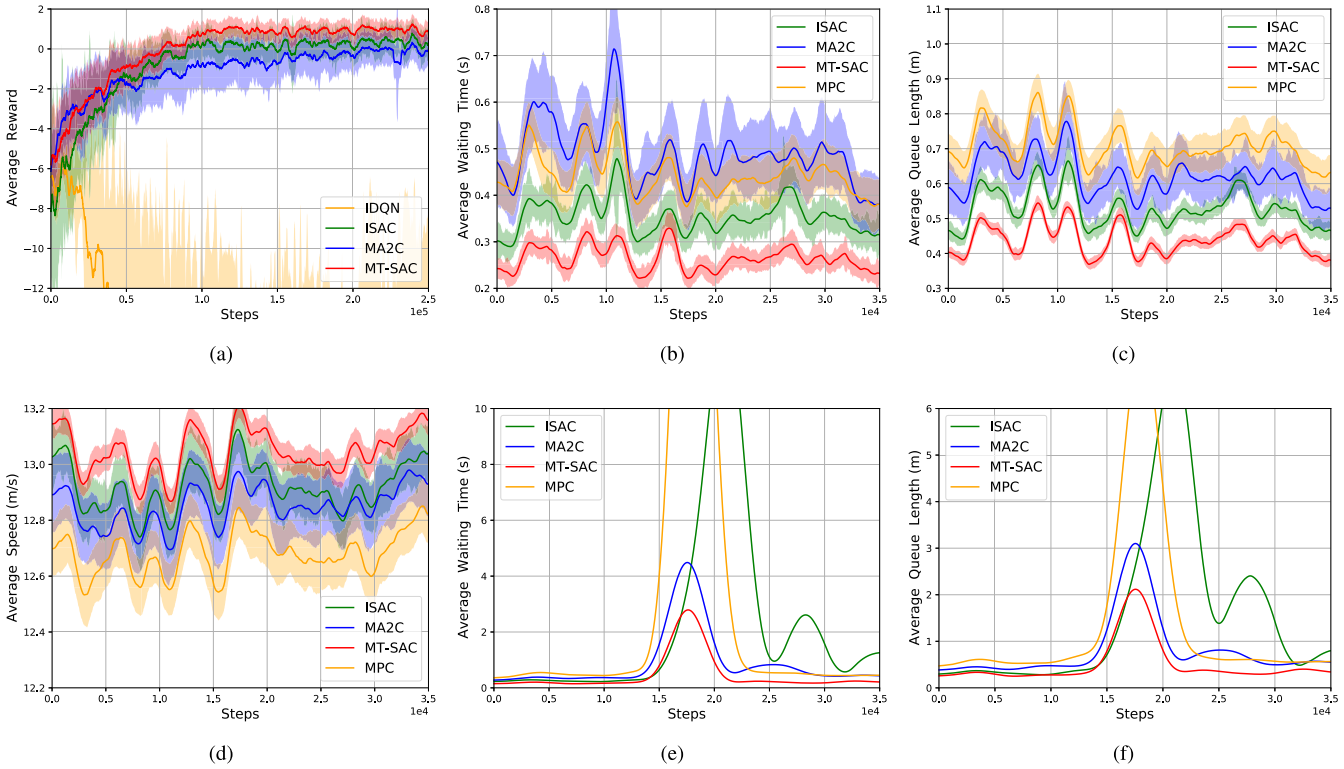


Fig. 10. The result curves of the different algorithms in the large-scale scenario. The item meanings of (a), (b), (c), and (d) are the same as those in Fig. 8. (e) and (f) represent the average waiting time and average queue length of the vehicles of $Agent_5$ in where the occasional congestion occurs between the 18000th and 19000th step.

2) Training Results: The training curves of the four methods are shown in Fig. 10 (a), and the meaning of the curves are the same as those in Fig. 8 (a). It can be seen that the IDQN does not converge, which implies that simple DQN based methods without any cooperation cannot solve complex tasks. For other methods, they converge to different optima. The MA2C learns faster than ISAC but the reward is lower, and the proposed MT-SAC yields the highest reward.

3) Evaluation Results: Fig. 10 (b), (c) and (d) plot the average waiting time, average queue length, and average speed of the traffic network respectively. Since the IDQN does not converge in the training process, to show the results more clearly, its results are not included in the figures for comparison. The curves show that the MT-SAC is more adaptive and stable than the MA2C, ISAC, and MPC. And the results in Table IV show that the MT-SAC performs the best.

To test the robustness of the algorithms, we further carry out experiments on the same large-scale scenario with occasional congestion occurs at the intersection $Agent_5$. The blue dotted line in Fig. 6 (b) shows the occasional congestion traffic flow. Fig. 10 (e) and (f) show the curves of waiting time and queue length in $Agent_5$. It can be seen that even if the ISAC can solve the occasional congestion during a certain period, it will still affect the subsequent traffic flow management. Because the subsequent states may not be experienced by the agent, it will be difficult to take appropriate actions. The MA2C, MPC, and MT-SAC can handle this situation and become stable, but the MT-SAC performs more stable and better. This indicates that the multi-view encoder and transfer learning paradigm enable MT-SAC to have a stronger ability to deal with complex traffic situations.

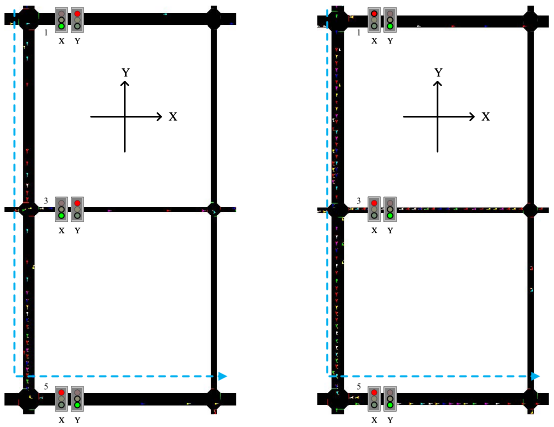
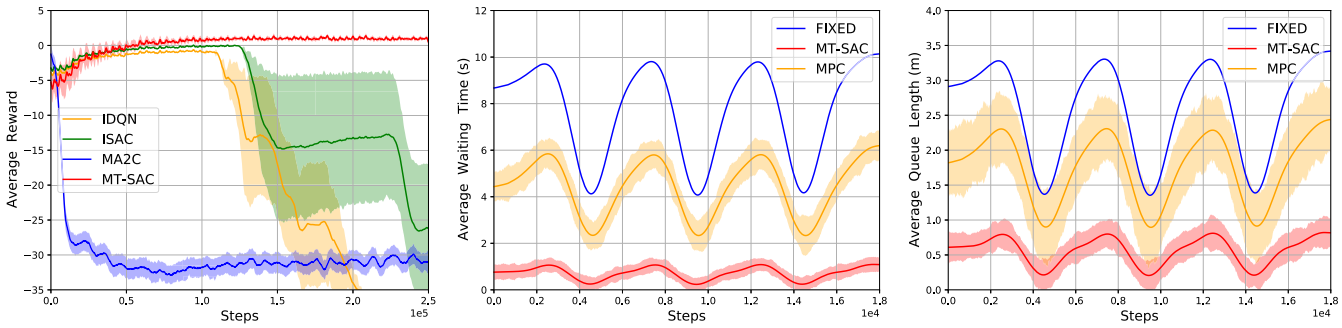


Fig. 11. The snapshots of the region near $Agent_5$ after occasional congestion in the large-scale scenario with MT-SAC and ISAC.

To give analysis of the cooperation from the dynamics of traffic light phase learned by the proposed MT-SAC. At the time when the algorithms run for 500 steps after the occasional congestion, the snapshots of the region near $Agent_5$ are presented in Fig. 11, where Fig. 11 (a) presents the snapshot results of MT-SAC, and Fig. 11 (b) presents the snapshot results of ISAC. In Fig. 11 (a), there are more vehicles in Y direction than in X direction around $Agent_3$, but the $Agent_3$ takes an action that prohibits the vehicles passing along Y



(a) (b) (c)

Fig. 12. The result curves of the different algorithms on the real world scenario. The item meanings of (a), (b), and (c) are the same as those in Fig. 8.

direction. And the $Agent_1$ takes the similar action. Such actions will alleviate the congestion burden of $Agent_5$. On the other hand, in Fig. 11 (b), the agents only consider their local rewards, and they take the action that allows vehicles to pass along the directions having more vehicles. This will lead to more serious congestion. So it can be observed that the MT-SAC has stronger cooperation ability than the ISAC.

E. Evaluation Experiments on Real-World Scenario

1) Detailed Settings: To evaluate the effectiveness of the proposed method, we employ a set of experiments in real-world scenario. The map of the scenario is taken from Dongfeng sub-district of Jinan City, China. The traffic flow data is taken from real world. There are homogeneous 12 intersections in the map, and each intersection has 24 lanes with four available actions (phases). The speed limit of each lane is 11.11m/s. The distances between two intersections along X direction (east to west) and Y direction (south to north) are 400m and 800m, respectively. The first 65m of the lanes that are close to the intersection are used to construct the state matrices of the intersections. The vehicles are assumed to have the same length of 5m. The traffic density is to 6300 veh/h. The weights for rewards are taken as $W = \{0.5, 0.1, 1, 0.2\}$.

2) Training Results: The training curves of the four methods are shown in Fig. 12 (a). It can be seen that the curves of the ISAC and IDQN are flat at first and drop down steeply after a few training steps. The curve of MA2C drops down steeply at first, then becomes flat, and the average reward continuously keeps around -30. On the other hand, the MT-SAC can improve the rewards steadily. The results indicate that the ISAC and IDQN can be effectively trained for some time, but after that, they fail to solve the subsequent road conditions. The results also indicate that the MA2C fails to deal with this kind of traffic situation. Only the MT-SAC completes the whole training well and converges to the optimal reward. The reasons that we obtain such results lie with the followings. First, the multi-view encoder encodes the state from multiple perspective, thus its make the information obtained by the agents being comprehensive. Second, the transfer learning enables the agents to handle more complex road conditions.

3) Evaluation Results: Fig. 12 (b) and (c) plots the average waiting time and average queue length of the traffic network. Since the IDQN, ISAC, MA2C yield significantly worse results, to show the results more clearly, their results are not included in the figures for comparisons. From the figure, it can be seen that the MPC has better ability to deal the real-world traffic flow than the FIXED. Compared with the RL-based methods, the MPC requires shorter training time, but it takes a longer time in the execution stage. The curves in Fig. 12(b) and (c) indicate that the MT-SAC performs best in real-world scenario and is more stable than other algorithms.

F. Ablation Results

In this subsection, we conduct a set of ablation experiments to demonstrate the roles played by the multi-view state encoder and the transfer learning paradigm with cooperative guidance.

1) The Effect of Multi-View State Encoder: The multi-view state includes the 1D state, 2D state, and attention state. We conduct four sets of experiments, with the first three experiments eliminating the 1D state, 2D state, and the attention state, respectively, and the fourth experiment taking the whole multi-view state. The results are presented in Table V. The '$-$' item means that the efficiency of the method is worse than the FIXED method.

As can be seen from Table V, the Expert that takes the whole multi-view state as input yields the best results for all experiments in all scenarios, which shows that the proposed encoder has comprehensive representation ability for the current environment states. When making comparisons between the Expert in MT-SAC with the whole multi-view state and the Expert with the 1D, 2D, attention state, respectively, the following results can be observed. First, the 1D state plays a positive role in the entire algorithm. Second, the 2D state plays the most important role in the entire algorithm, which indicates that the quality of information contained in the implicit spatial topology is high. And finally, the attention state plays a more positive role than the 1D state, which indicates that the attention state can prompt the agent to generate a better strategy.

2) The Effect of Transfer Learning Paradigm With Guidance: The MT-SAC paradigm involves three training stages. In this subsection, we conduct three ablation experiments to demonstrate the effect of each training stage. In the first ablation, we only train the specific experts for the whole entire

TABLE V
RESULTS OF MULTI-VIEW STATE ENCODER ABLATION EXPERIMENT

Metrics	Small-scale scenario				Large-scale scenario				Real-world scenario			
	w/o 1D	w/o 2D	w/o Att	Expert	w/o 1D	w/o 2D	w/o Att	Expert	w/o 1D	w/o 2D	w/o Att	Expert
sum.car [veh]	12.81	-	13.04	**12.48**	9.32	-	9.40	**9.05**	27.29	-	29.86	**25.84**
ave.queue [m]	1.12	-	1.14	**1.10**	0.58	-	0.61	**0.52**	0.93	-	1.05	**0.79**
ave.speed [m/s]	10.75	-	10.71	**10.82**	12.80	-	12.87	**12.99**	9.40	-	9.22	**9.61**
ave.wait [s]	0.67	-	0.69	**0.63**	0.38	-	0.36	**0.34**	0.94	-	0.99	**0.81**

TABLE VI
RESULTS OF TRANSFER LEARNING WITH GUIDANCE ABLATION EXPERIMENT

Metrics	Small-scale scenario				Large-scale scenario				Real-world scenario			
	Expert	Seed	Ex-Con	MT-SAC	Expert	Seed	Ex-Con	MT-SAC	Expert	Seed	Ex-Con	MT-SAC
sum.car [veh]	12.48	12.51	12.47	**12.02**	9.05	9.09	9.06	**8.48**	25.84	26.34	25.88	**24.02**
ave.queue [m]	1.10	1.11	1.10	**1.04**	0.52	0.53	0.52	**0.42**	0.79	0.83	0.80	**0.56**
ave.speed [m/s]	10.82	10.80	10.82	**10.96**	12.99	12.98	13.00	**13.04**	9.61	9.55	9.60	**9.89**
ave.wait [s]	0.63	0.62	0.63	**0.55**	0.34	0.51	0.34	**0.27**	0.81	1.21	0.84	**0.76**

TABLE VII
THE NUMBER OF PARAMETERS OF DIFFERENT MODELS

	MTSAC	IDQN	ISAC	MA2C	MPC
parameters	13.95M	11.07M	11.90M	610M	1.3M

TABLE VIII
THE TRAINING TIME OF AN EPISODE TRAINING

scenarios	MTSAC	IDQN	ISAC	MA2C
Small-scale	92s	29s	37s	20s
Large-scale	184s	-	55s	30s
Real-world	195s	-	-	-

scenario, we refer to the experiment as Expert. In the second ablation, we only train the seed phantom experts, we refer to the experiment as Seed. In particular, ε in each experiment is initialized to 0. To provide a fair comparison with the MT-SAC, we extend the number of training episodes of Expert until it is the same as that of MT-SAC, we refer to the experiment as Ex-Con. All the ablation experiments are carried out under the same traffic flow and settings.

The comparison results with the entire MT-SAC are presented in Table VI. It can be found that the Expert has the ability to handle the traffic conditions well. The comparison between the Expert and Seed shows that they have their own advantages on different metrics. However, the combination of them, i.e., the MT-SAC, yields the best results. It is because the Expert uses different networks to control the intersections, and the Expert only considers maximizing its own reward, the Seed is cooperatively guided by experts and thus it pays more attention to the generalization in different scenarios. The MT-SAC has the advantages of both the Expert and the Seed. The comparison between the Ex-Con and the MT-SAC indicates that the best result cannot be obtained by directly increasing training episodes. The above ablation results indicate that the proposed training strategies play positive roles in the MT-SAC.

G. Complexity Analysis

We carry out complexity analysis towards the proposed method from the perspective of training time involved and the number of model parameters involved.

Table VII shows the number of model parameters involved in the MT-SAC and the compared algorithms. The MA2C adopts FC layers without encoder network, so the number of the parameters is large. Table VIII shows the training time of the MT-SAC and the compared algorithms in one episode. The symbol '−' represents that the algorithm does not converge in corresponding scenarios. For the MT-SAC, it can be seen that the total training time is 92s on small-scale scenario, 184s on large-scale scenario, and 195s on real-world scenario. And the training time includes the three stages of the MT-SAC. In fact, the training time not only relies on the complexity of the model but also is influenced by the density of traffic flow in the traffic simulator and the number of agents.

It should be noted that the training process of the MT-SAC is offline, and the training time provided above is offline training time. However, in the process of decision-making in practical applications, only forward computation is needed, so the computation times in the execution phase are similar and short for all the RL-based algorithms.

VI. CONCLUSION

This paper proposes a SAC-based MARL algorithm (MT-SAC) for multi-intersection traffic signal control. A multi-view state encoder and a transfer learning paradigm with guidance are proposed to enhance the learning abilities of the agents. The multi-view encoder encodes the global state of the agents in the region by attention mechanism to generate the attention state. And the multi-view state is further constructed to make the agent learn better cooperative strategies. The transfer learning paradigm enables the agent to learn adaptively so that it can discover characteristics and regularities of the intersections for improving generalization ability. Besides, we design the state representation to enable agents to handle intersections with different topologies. Extensive experiments and analysis have been carried out in two synthetic and one real-world environment with different scales, which demonstrate the effectiveness and robustness of the MT-SAC.

In the future, we will consider other mechanisms that can facilitate collaboration in ATSC, for example, letting the agent generate a controllable language instead of the original state, action, or reward. Also, it is in demand to make the agents learn as much as possible from the data generated in other environments to obtain an RL-based algorithm that can be applied in real-world applications efficiently.

REFERENCES

[1] N. Kumar, S. S. Rahman, and N. Dhakad, "Fuzzy inference enabled deep reinforcement learning-based traffic light control for intelligent transportation system," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 8, pp. 4919–4928, Aug. 2020.

[2] H. Yang, H. Rakha, and M. V. Ala, "Eco-cooperative adaptive cruise control at signalized intersections considering queue effects," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 6, pp. 1575–1585, Jun. 2017.

[3] R. E. Allsop, "Delay at a fixed time traffic signal—I: Theoretical analysis," *Transp. Sci.*, vol. 6, no. 3, pp. 260–285, Aug. 1972.

[4] P. Hunt, D. Robertson, R. Bretherton, and M. C. Royle, "The SCOOT online traffic signal optimisation technique," *Traffic Eng. Control*, vol. 23, no. 4, pp. 190–192, 1982.

[5] J. Luk, "Two traffic-responsive area traffic control methods: SCAT and SCOOT," *Traffic Eng. Control*, vol. 25, no. 1, p. 14, 1984.

[6] H. Ceylan and M. G. H. Bell, "Traffic signal timing optimisation based on genetic algorithm approach, including drivers' routing," *Transp. Res. B, Methodol.*, vol. 38, no. 4, pp. 329–342, 2004.

[7] K. T. K. Teo, W. Y. Kow, and Y. K. Chin, "Optimization of traffic flow within an urban traffic light intersection with genetic algorithm," in *Proc. IEEE 2nd Int. Conf. Comput. Intell., Modelling Simulation*, Sep. 2010, pp. 172–177.

[8] H. Asadi, R. T. Moghaddam, N. S. Pourorcid, and E. Najafi, "A new nondominated sorting genetic algorithm based to the regression line for fuzzy traffic signal optimization problem," *Sci. Iranica*, vol. 25, no. 3, pp. 1712–1723, 2018.

[9] K.-H. Chao, R.-H. Lee, and M.-H. Wang, "An intelligent traffic light control based on extension neural network," in *Proc. Int. Conf. Knowl.-Based Intell. Inf. Eng. Syst.* Zagreb, Croatia: Springer, 2008, pp. 17–24.

[10] G. Shen and X. Kong, "Study on road network traffic coordination control technique with bus priority," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 39, no. 3, pp. 343–351, May 2009.

[11] H. Wan, "Research on traffic signal control algorithm based on self-organizing competitive neural network and optimization model," *Agro Food Ind. Hi-Tech*, vol. 28, no. 3, pp. 2911–2914, 2017.

[12] B. P. Gokulan and D. Srinivasan, "Distributed geometric fuzzy multiagent urban traffic signal control," *IEEE Trans. Intell. Transp. Syst.*, vol. 11, no. 3, pp. 714–727, Sep. 2010.

[13] W. Yang, L. Zhang, Z. He, and L. Zhuang, "Optimized two-stage fuzzy control for urban traffic signals at isolated intersection and paramics simulation," in *Proc. 15th Int. IEEE Conf. Intell. Transp. Syst.*, Sep. 2012, pp. 391–396.

[14] M. J. S. Shiri and H. R. Maleki, "Maximum green time settings for traffic-actuated signal control at isolated intersections using fuzzy logic," *Int. J. Fuzzy Syst.*, vol. 19, no. 1, pp. 247–256, Feb. 2017.

[15] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 2018.

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

[17] L. A. Prashanth and S. Bhatnagar, "Reinforcement learning with function approximation for traffic signal control," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 2, pp. 412–421, Jun. 2011.

[18] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 1861–1870.

[19] S. Araghi, A. Khosravi, M. Johnstone, and D. Creighton, "Intelligent traffic light control of isolated intersections using machine learning methods," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, Oct. 2013, pp. 3621–3626.

[20] J. Jin and X. Ma, "Adaptive group-based signal control by reinforcement learning," *Proc. Transp. Res.*, vol. 10, pp. 207–216, Jan. 2015.

[21] S. Touhbi *et al.*, "Adaptive traffic signal control: Exploring reward definition for reinforcement learning," *Proc. Comput. Sci.*, vol. 109, pp. 513–520, Jan. 2017.

[22] T. Chu, J. Wang, L. Codecà, and Z. Li, "Multi-agent deep reinforcement learning for large-scale traffic signal control," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 3, pp. 1086–1095, Mar. 2020.

[23] C.-H. Wan and M.-C. Hwang, "Value-based deep reinforcement learning for adaptive isolated intersection signal control," *IET Intell. Transp. Syst.*, vol. 12, no. 9, pp. 1005–1010, 2018.

[24] S. El-Tantawy and B. Abdulhai, "Comprehensive analysis of reinforcement learning methods and parameters for adaptive traffic signal control," Transp. Res. Board 90th Annu. Meeting, Washington, DC, USA, Tech. Rep. 11-1492, 2011.

[25] L. Li, Y. Lv, and F.-Y. Wang, "Traffic signal timing via deep reinforcement learning," *IEEE/CAA J. Autom. Sinica*, vol. 3, no. 3, pp. 247–254, Apr. 2016.

[26] J. Zeng, J. Hu, and Y. Zhang, "Adaptive traffic signal control with deep recurrent Q-learning," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2018, pp. 1215–1220.

[27] H. Wei, G. Zheng, H. Yao, and Z. Li, "IntelliLight: A reinforcement learning approach for intelligent traffic light control," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2018, pp. 2496–2505.

[28] M. Wiering, "Multi-agent reinforcement learning for traffic light control," in *Proc. 17th Int. Conf. Mach. Learn. (ICML)*, 2000, pp. 1151–1158.

[29] M. Abdoos, N. Mozayani, and A. L. C. Bazzan, "Traffic light control in non-stationary environments based on multi agent Q-learning," in *Proc. 14th Int. IEEE Conf. Intell. Transp. Syst. (ITSC)*, Oct. 2011, pp. 1580–1585.

[30] J. Gao, Y. Shen, J. Liu, M. Ito, and N. Shiratori, "Adaptive traffic signal control: Deep reinforcement learning algorithm with experience replay and target network," 2017, *arXiv:1705.02755*. [Online]. Available: http://arxiv.org/abs/1705.02755

[31] M. Schutera, N. Goby, S. Smolarek, and M. Reischl, "Distributed traffic light control at uncoupled intersections with real-world topology by deep reinforcement learning," 2018, *arXiv:1811.11233*. [Online]. Available: http://arxiv.org/abs/1811.11233

[32] S. Richter, D. Aberdeen, and J. Yu, "Natural actor-critic for road traffic optimisation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2007, pp. 1169–1176.

[33] A. Salkham, R. Cunningham, A. Garg, and V. Cahill, "A collaborative reinforcement learning approach to urban traffic control optimization," in *Proc. IEEE/WIC/ACM Int. Conf. Web Intell. Intell. Agent Technol.*, vol. 2, Dec. 2008, pp. 560–566.

[34] P. G. Balaji, X. German, and D. Srinivasan, "Urban traffic signal control using reinforcement learning agents," *IET Intell. Transp. Syst.*, vol. 4, no. 3, pp. 177–188, Sep. 2010.

[35] H. M. A. Aziz, F. Zhu, and S. V. Ukkusuri, "Learning-based traffic signal control algorithms with neighborhood information sharing: An application for sustainable mobility," *J. Intell. Transp. Syst.*, vol. 22, no. 1, pp. 40–52, 2018.

[36] H. Ge, Y. Song, C. Wu, J. Ren, and G. Tan, "Cooperative deep Q-learning with Q-value transfer for multi-intersection signal control," *IEEE Access*, vol. 7, pp. 40797–40809, 2019.

[37] D. Kim and O. Jeong, "Cooperative traffic signal control with traffic flow prediction in multi-intersection," *Sensors*, vol. 20, no. 1, p. 137, Dec. 2019.

[38] S. El-Tantawy, B. Abdulhai, and H. Abdelgawad, "Multiagent reinforcement learning for integrated network of adaptive traffic signal controllers (MARLIN-ATSC): Methodology and large-scale application on downtown Toronto," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 3, pp. 1140–1150, Sep. 2013.

[39] T. Chu and J. Wang, "Traffic signal control by distributed reinforcement learning with min-sum communication," in *Proc. IEEE Amer. Control Conf. (ACC)*, May 2017, pp. 5095–5100.

[40] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, pp. 529–533, Feb. 2015.

[41] H. van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double Q-learning," in *Proc. 30th Assoc. Adv. Artif. Intell. Conf. Artif. Intell. (AAAI)*, 2016, pp. 2094–2100.

[42] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized experience replay," 2015, *arXiv:1511.05952*. [Online]. Available: http://arxiv.org/abs/1511.05952

[43] R. S. Sutton, D. A. McAllester, S. P. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2000, pp. 1057–1063.

[44] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 1889–1897.

[45] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017, *arXiv:1707.06347*. [Online]. Available: http://arxiv.org/abs/1707.06347
[46] V. Mnih *et al.*, "Asynchronous methods for deep reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1928–1937.
[47] T. P. Lillicrap *et al.*, "Continuous control with deep reinforcement learning," in *Proc. Int. Conf. Learn. Represent.*, 2016, pp. 1–10.
[48] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 1587–1596.
[49] P. Sunehag *et al.*, "Value-decomposition networks for cooperative multi-agent learning based on team reward," in *Proc. AAMAS*, 2018, pp. 2085–2087.
[50] T. Rashid, M. Samvelyan, C. Schroeder, G. Farquhar, J. Foerster, and S. Whiteson, "QMIX: Monotonic value function factorisation for deep multi-agent reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 4295–4304.
[51] A. Singh, T. Jain, and S. Sukhbaatar, "Learning when to communicate at scale in multiagent cooperative and competitive tasks," in *Proc. Int. Conf. Learn. Represent.*, 2019, pp. 1–11.
[52] A. Das *et al.*, "TarMAC: Targeted multi-agent communication," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 1538–1546.
[53] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010.
[54] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.
[55] P. A. Lopez *et al.*, "Microscopic traffic simulation using sumo," in *Proc. IEEE Int. Conf. Intell. Transp. Syst.*, Nov. 2018, pp. 2575–2582.
[56] K. Chua, R. Calandra, R. McAllister, and S. Levine, "Deep reinforcement learning in a handful of trials using probabilistic dynamics models," in *Proc. 32nd Int. Conf. Neural Inf. Process. Syst.*, 2018, pp. 4759–4770.

Hongwei Ge received the B.S. and M.S. degrees in mathematics from Jilin University, China, and the Ph.D. degree in computer application technology from Jilin University in 2006. He is currently a Professor with the College of Computer Science and Technology, Dalian University of Technology, Dalian, China. He has published more than 100 articles in prestigious journals, such as IEEE TRANSACTIONS ON CYBERNETICS, IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS: SYSTEMS, and *Pattern Recognition*. His research interests include artificial intelligence, machine learning, deep learning, swarm intelligence, and intelligent systems.

Dongwan Gao received the B.S. and M.S. degrees from Dalian University of Technology, Dalian, China, in 2018 and 2021, respectively, where he is currently pursuing the Ph.D. degree with the College of Computer Science and Technology. His main research interests include machine learning and multi-agent systems.

Liang Sun received the double Ph.D. degrees from Kochi University of Technology and Jilin University in March 2012 and June 2012, respectively. He is currently an Associate Professor with the College of Computer Science and Technology, Dalian University of Technology, Dalian, China. His main research interests include machine learning, computational intelligence, deep learning, and computer vision.

Yaqing Hou (Member, IEEE) received the Ph.D. degree in artificial intelligence from Nanyang Technological University, Singapore, in 2017. He is currently a Lecturer with the College of Computer Science, Dalian University of Technology, Dalian, China. His publications have appeared in IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION and *IEEE Computational Intelligence Magazine*. His current research interests include multi-agent reinforcement learning and transfer learning.

Chao Yu received the Ph.D. degree in computer science from the University of Wollongong, Australia, in 2014. He is currently an Associate Professor with the School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou, China. He has published more than 70 articles in prestigious journals, such as IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, IEEE TRANSACTIONS ON CYBERNETICS, and IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY. His research interests include multi-agent systems and reinforcement learning.

Yuxin Wang received the M.S. degree in computer application and the Ph.D. degree in computer science from Dalian University of Technology, China, in 2000 and 2012, respectively. He is currently an Associate Professor with the College of Computer Science, Dalian University of Technology. His research interests include distributed computing, big data analysis, and deep learning.

Guozhen Tan received the M.S. degree from Harbin Institute of Technology, Harbin, China, and the Ph.D. degree from Dalian University of Technology, Dalian, China. He is currently a Professor with the College of Computer Science and Technology, Dalian University of Technology. His current research interests include intelligent transportation systems and machine learning.