MetaVIM: Meta Variationally Intrinsic Motivated Reinforcement Learning for Decentralized Traffic Signal Control

Liwen Zhu, Peixi Peng, Zongqing Lu, Yonghong Tian *Fellow, IEEE*

Abstract—Traffic signal control aims to coordinate traffic signals across intersections to improve the traffic efficiency of a district or a city. Deep reinforcement learning (RL) has been applied to traffic signal control recently and demonstrated promising performance where each traffic signal is regarded as an agent. However, there are still several challenges that may limit its large-scale application in the real world. On the one hand, the policy of the current traffic signal is often heavily influenced by its neighbor agents, and the coordination between the agent and its neighbors needs to be considered. Hence, the control of a road network composed of multiple traffic signals is naturally modeled as a multi-agent system, and all agents' policies need to be optimized simultaneously. On the other hand, once the policy function is conditioned on not only the current agent's observation but also the neighbors', the policy function would be closely related to the training scenario and cause poor generalizability because the agents in various scenarios often have heterogeneous neighbors. To make the policy learned from a training scenario generalizable to new unseen scenarios, a novel Meta Variationally Intrinsic Motivated (MetaVIM) RL method is proposed to learn the decentralized policy for each intersection that considers neighbor information in a latent way. Specifically, we formulate the policy learning as a meta-learning problem over a set of related tasks, where each task corresponds to traffic signal control at an intersection whose neighbors are regarded as the unobserved part of the state. Then, a learned latent variable is introduced to represent the task's specific information and is further brought into the policy for learning. In addition, to make the policy learning stable, a novel intrinsic reward is designed to encourage each agent's received rewards and observation transition to be predictable only conditioned on its own history. Extensive experiments conducted on CityFlow demonstrate that the proposed method substantially outperforms existing approaches and shows superior generalizability.

Index Terms—Traffic Signal Control, Reinforcement Learning, Meta Reinforcement Learning, Variational Autoencoder

1 Introduction

Traffic signals that direct traffic movements play an important role in efficient transportation. Most conventional methods aim to control traffic signals by fixed-time plans [1] or hand-crafted heuristics [2]. However, the predefined rules cannot adapt to dynamic and uncertain traffic conditions. Recently, deep reinforcement learning (RL) [3], [4], [5], [6], [7], [8], [9], [10], [11] has been applied in Intelligent Transportation Systems (ITS) and demonstrated promising performance. Specifically, recent works [12], [13], [14] use RL for traffic signal control, and provide a promising way to handle dynamic and uncertain traffic conditions, where a RL agent employs a deep neural network to control an intersection and the network is learned by directly interacting with the environment.

The most straightforward RL baseline considers each intersection independently and models the task as a single agent RL problem [12]. However, the observation, received reward and dynamics of each traffic signal are closely related to its neighbors, and the coordination between signals should be modeled. Hence, optimizing traffic signal control in a large-scale road network could be modeled as a multi-agent reinforcement learning (MARL) problem. Prior works [15], [16] in this domain resort to centralized training to ensure that agents learn to coordinate, they demonstrate that communication among agents could help coordination. However, as the joint action space grows exponentially with the number of agents, it is infeasible or costly in realistic deployment, and training emergent communication protocols also remains a challenging problem. In addition, once the policy function is conditioned on not only the current agent's observation but also the neighbors', the policy function would be closely related to the training scenario and cause poor generalizability because the agents in various scenarios often have heterogeneous neighbors. Therefore, learning decentralized policies opens up a window of new opportunities for advanced MARL research in this area.

To learn effective decentralized policies, there are two main challenges. Firstly, it is impractical to learn an individual policy for each intersection in a city or a district containing thousands of intersections. Parameter sharing may help. However, each intersection has a different traffic pattern, and a simple shared policy hardly learns and acts optimally at all intersections. To handle this challenge,

- *Corresponding authors: Peixi Peng and Yonghong Tian.*
- *Liwen Zhu, Peixi Peng, Zongqing Lu and Yonghong Tian are with National Engineering Research Center of Visual Technology, School of Computer Science, Peking University, China, and are also with Pengcheng Laboratory, Shenzhen, China. E-mail:{liwenzhu,pxpeng,zongqing.lu,yhtian}@pku.edu.cn.*
- *This work is partially supported by grants from the Key-Area Research and Development Program of Guangdong Province under contact No. 2020B0101380001 and grants from the National Natural Science Foundation of China under contract No. 62027804, No. 61825101 and No. 62088102. The computing resources of Pengcheng Cloudbrain are used in this research.*

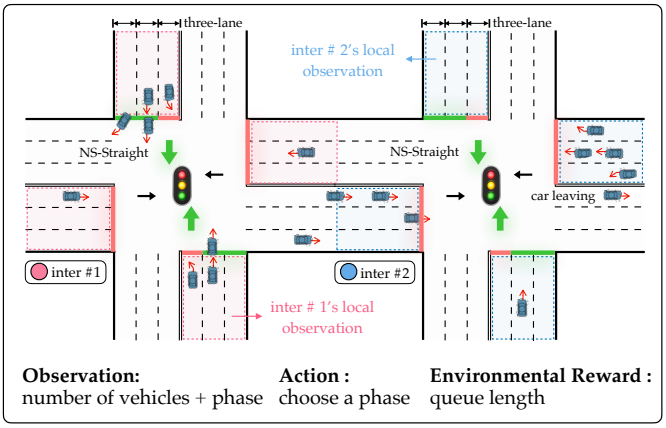


Fig. 1: Illustration of the multi-intersection control scenario where policies and rewards are affected by neighboring policies, i.e., the instability of the multi-agent scenario.

we formulate the policy learning in a road network as a meta-learning problem, where traffic signal control at each intersection corresponds to a task, and a policy is learned to adapt to various tasks. Reward function and state transition of these tasks vary but share similarities since they follow the same traffic rules and have similar optimization goals. Therefore, we represent each task as a learned and low-dimensional latent variable obtained by encoding the past trajectory in each task. The latent variable is a part of the input of the policy, which captures task-specific information and helps improve the policy adaption.

Secondly, even for a specific task, the received rewards and observations are uncertain to the agent, as illustrated in Fig. 1, which make the policy learning unstable and non-convergent. Even if the agent performs the same action on the same observation at different timesteps, the agent may receive different rewards and observation transitions because of neighbor agents' different actions. In this case, the received rewards and observation transitions of the current agent could not be well predicted only conditioned on its own or partial neighbors' observations and performed actions. To avoid this situation, four decoders are introduced to predict the next observations and rewards without neighbor agents' policies or with partially neighbor agents, respectively. In addition, an intrinsic reward is designed to reduce the bias among different predictions and enhance learning stability. In other words, the design of the decoders and intrinsic reward is similar to the law of contra-positive. The unstable learning will cause the predicted rewards and observation transitions unstable in a decentralized way, while our decoders and intrinsic reward encourage the prediction convergent. In addition, from the perspective of information theory, the intrinsic reward design makes the policy of each agent robust to neighbours' polices, which could make the learned policy easy to transfer.

Intrinsic motivation refers to reward functions that allow agents to learn useful behaviors across various tasks. Previous approaches to intrinsic motivation often focus on curiosity [17], imagination [18] or synergy [19], these approaches rely on hand-crafted rewards specific to the environment, or limit to the bimanual manipulation tasks. Such structural constraints make it impossible to achieve independent training of MARL agents across multiple environments. Here, we consider the problem of deriving intrinsic motivation as an exploration bias from other agents in MARL. Our key idea is that a good guiding principle for intrinsic motivation is to make approximations robust to a dynamic environment. Intrinsic motivation is assessed using comparison reasoning: at each timestep, an agent measures the effect of another agent's policy on its model's predictions. Neighbor's policies that lead to a relatively higher change in an agent's own model estimates are considered highly influential, and because the explicit effect of neighbor's influence in this task is to increase incoming traffic volumes, which exacerbates the agent's own traffic load. Therefore, we minimize this influence bias to make the model's estimations free from interference from others. We show that this inductive bias will drive agents to learn effective decentralized policies.

We conduct extensive experiments on CityFlow [20] in public datasets Hangzhou (China), Jinan (China), New York (USA), and our derived dataset Shenzhen (China) road networks under various traffic patterns, and empirically demonstrate that our proposed method can achieve state-of-the-art performances over the above scenarios. Furthermore, our method shows superior adaptivity in transfer experiments. In summary, the main contributions of this paper are concluded as three aspects:

1) The traffic signal control is modeled as a meta-learning problem over a set of related tasks, and a novel method MetaVIM is proposed to learn decentralized policy to handle large-scale traffic lights.

2) A learnable latent variable is introduced to represent task-specific information by performing approximate inference on the task, and make the policy function shareable across the tasks.

3) A novel intrinsic reward is designed to tackle the challenge of unstable policy learning in dynamically changing traffic environments. We show that this design will drive agents to learn effective decentralized policies.

The paper is structured as follows. We describe the related work in Section 2, and the MARL setting in Section 3. Section 4 introduces the details of the proposed method. Section 5 presents experimental results that empirically demonstrate the efficacy of MetaVIM. Finally, conclusions and future work are discussed in Section 6.

2 Related Work

2.1 Conventional and Adaptive Traffic Signal Control

Most conventional traffic signal control methods are designed based on fixed-time signal control [21], actuated control [22] or self-organizing traffic signal control [23]. These approaches rely on expert knowledge and often perform unsatisfactorily in complicated real-world situations. To solve this problem, several optimization-based methods have been proposed to optimize average travel time, throughput, *etc.*, which decide the traffic signal plans according to the dynamical observed data. Specifically, the method [2] calculates the difference between the number of upstream and downstream vehicles and then sorts the values to determine the phase order. Besides, several methods [24], [25], [26], [27] consider the scheduling of urban traffic light as the model-based optimization problem and employ search based methods

to solve it. Furthermore, the dynamic programming [28], mixed-integer linear programming [29] and non-linear programming models [30] are also used. Please see [31] for more details. Although effective in some situations, these approaches typically rely on prior assumptions which may fail in some cases, or require environment model which is unavailable and unreliable in complex scenarios.

2.2 RL-based Traffic Signal Control

RL-based traffic signal control methods aim to learn the policy from interactions with the environment. Earlier studies use tabular Q-learning [32], [33], [34], [35] where the states are required to be discretized and low-dimensional. To handle more complex continuous state, recent advances employ deep RL to map the high-dimensional state representations (such as images or feature vectors) into actions.

Efforts have been made to design strategies that formulate the task as a single agent [12], [14], [36], [37] or some isolated intersections [38], [39], [40], [41], [42], [43] , i.e., each agent makes decision for its own. This type of methods is usually easy to scale, but may have difficulty to achieve global optimal performance due to the lack of collaboration. To address the problem, another way is to jointly model the action among learning agents with centralized optimization [15], [16]. However, as the number of agents increases, joint optimization usually leads to dimensional explosion, which has inhibited the widespread adoption of such methods to a large-scale traffic signal control. To overcome the difficulty, another type of methods are implemented in a decentralized manner. For example, the methods proposed in [32], [44] directly add neighboring information into states, and the neighbors' hidden features are merged into states in [3], [45], [46], [47]. Compared with them, our method uses neighbor information to form intrinsic motivation rather than as additional input of the policy. It makes our method easy to transfer to a new scenario which may have different neighbour topology with the training scenario. Besides, the neighborhood travel time is optimized in [48] as an additional reward. However, simple concatenation of neighboring information is not reasonable enough because the influence of neighboring intersections is not balanced.

To make the policy transferable, traffic signal control is also modeled as a meta-learning problem in [14], [36], [49]. Specifically, the method in [14] performs meta-learning on multiple independent MDPs and ignores the influences of neighbor agents. A data augmentation method is proposed in [49] to generates diverse traffic flows to enhance meta-RL, and also regards agents as independent individuals, without explicitly considering neighbors. In addition, a model-based RL method is proposed in [36] for high data efficiency. However it may introduce cumulative errors due to error of the learned environment model and it is hard to achieve the asymptotic performance of model-free methods. Our method both belongs to meta-RL paradigms, the main advantages are two main aspects Firstly, we consider the neighbour information during the meta-learning, which is critical for the multi-agent coordination. Secondly, our method learns a latent variable to represent task-specific information, which can not only balance exploration and exploitation [50], but also help to learn the shared structures of reward and transition across tasks. As far as we know, our work is the first to propose an intrinsic motivation to enhance the robustness of the policy on traffic signal control. See Appendix F for a brief overview of the above methods.

2.3 Intrinsic Reward

Intrinsic motivation methods have been widely studied in the literature, such as handling the difficult-to-learn dilemma in sparse reward environments [51] or trading off the exploration and exploitation in non-sparse reward scenarios [50]. Most of the intrinsic reward approaches can be classified into two classes. The first class is counted-based paradigm, where agents are incentivized to reach infrequently visited states by maintaining state visitation counts [52], [53] or density estimators [54], [55]. However, this paradigm is challenging in continuous or high-dimensional state space. The second is curiosity-based paradigm, in which agents are rewarded for high prediction error in a learned reward [17], [56] or inverse dynamics model [55], [57]. The uncertainty of the agent's assessment of its behavior can be measured as a curiosity for environmental exploration.

Besides the above two classes, other intrinsic reward methods are mainly task-oriented and for a specific purpose. For example, the method in [19] uses the discrepancy between the marginal policy and the conditional policy as the intrinsic reward for encouraging agents to have a greater social impact on others. The errors between the joint cooperative behaviors and the individual actions are defined in [58] as an intrinsic reward, which is suitable for agent-pair tasks that rely heavily on collaboration, such as dual-arm robot tasks. Similar with them, the proposed intrinsic reward is specially designed for the traffic signal control task, and we adopt the discrepancy with/without neighbor agent's policies as a motivation for decentralized control in dynamically changing multi-agent scenarios, making the policy robust to the environment.

2.4 Latent Variable in RL

A number of prior works have explored how RL can be cast in the framework of variational inference. Latent variable could transform the dynamically updated task-related information such as trajectories into a continuous lower-dimensional space. For example, [59] shows that exploring in latent space can enhance the representation of the environment. In meta-RL, the agent is not given prepared task-specific data to adapt to, and it must fully explore the environment to collect useful information. Privileged information generated during the learning phase can be used during meta-training to guide exploration, such as expert trajectories [60], dense reward for meta-training but not testing [61], or ground-truth task IDs / descriptions [62], [63]. Similar work [64] introduces temporally-extended exploration with latent variable. A series of methods [50], [59], [61] use variational auto-encoders (VAE) [65] structure to help explore the environment. A branch of context-based methods automatically learns to trade-off exploration and exploitation by maximizing average adaptation performance [50], [66]. Differently, we learn a dynamical latent variable for each task to present task-specific information and indicate the correspond agent's belief.

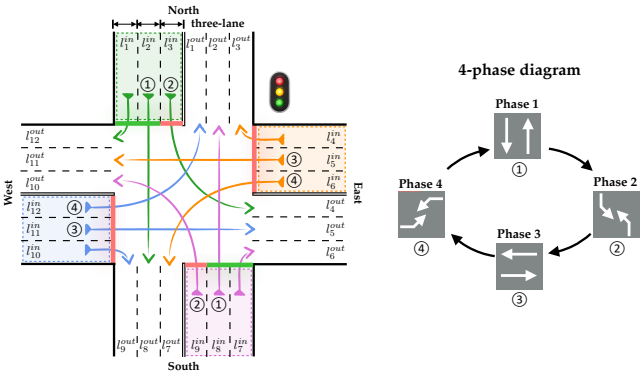


Fig. 2: Illustration of incoming, outgoing lanes and 4-phase diagram.

3 Problem Statement

3.1 Preliminary

In this paper, we investigate traffic signal control of multi-intersection as illustrated in Fig. 6. In order to explain the basic concepts, we use a 4-way intersection as an example, depicted in Fig. 2. Note that the concepts are easily generalized to different intersection structures (e.g., different numbers of entering approaches or phases).

Definition 1 (*Incoming/Outgoing Lanes*) For an intersection, the incoming lanes refer to the lanes where the vehicles are about to enter the intersection. In real world, most intersections are equipped with 4-way entering approaches, but some are 3-way or 5-way intersections. A standard 4-way intersection is shown in Fig. 2, which consists of four approaches, i.e., "east", "south", "west" and "north". Each approach consists of three types of lanes, representing "left-turn", "straight" and "right-turn" directions from inner to outer. The outgoing lanes refer to the lanes where the vehicles are about to leave the intersection. Notes that vehicles on the incoming lanes are affected directly by the traffic signal at the current intersection. Therefore, we adopt the traffic information on the incoming lanes as part of the observation, which is the same as most existing works [13], [14], [41], [46].

Definition 2 (*Phase*) Phase is a controller timing unit associated with the control of one or more movements, representing the permutation and combination of different traffic flows. At each phase, vehicles in the specific lanes can continue to drive. The 4-phase setting is the most common configuration in reality, but the number of phases can vary due to different intersection topologies (3-way, 5-way intersections, etc.). Fig. 2 illustrates a standard 4-phase setting: "north-south-straight", "north-south-left", "east-west-straight" and "east-west-left", "north-south-straight" means that the signal on the corresponding lanes are green. Note that the signal on the right-turn lanes is always green for consistency with real world.

Definition 3 (*Average Travel Time*) The travel time of a vehicle is the time discrepancy between entering and leaving a particular area. A vehicle from the origin to the destination (OD) is regarded as a travel. Average travel time of all vehicles in a road network is the most frequently used measure to evaluate the performance of traffic signal control [12], [13], [14], [36], [41], [43], [46], [48], [49].

3.2 Problem Definition

The traffic signal control in a road network is modeled as a multi-agent system, where each signal is controlled by an agent. Before formulating the problem, we firstly design the learning paradigm by analyzing the characteristics of the traffic signal control (TSC). Due to the coordination among different signals, the most direct paradigm may be centralized learning. However, the global state information in TSC is not only highly redundant and difficult to obtain in realistic deployment, but also likely suffers from dimensional explosion. Moreover, once the policy function relies on the global state information or neighbors on execution, it is hard to transfer the policy from the training scenario to other unseen scenarios containing different road networks. Hence, it is natural to resort to the decentralized policy, which controls each signal only conditioned on its own history. However, the fully decentralized learning ignores the coordination. If agents are behaved independently, agents maximize their own rewards and may sacrifice the interests of others, it is difficult for the entire system to reach the optimum. Therefore, we model the task as Decentralized Partially Observable Markov Decision Process (Dec-POMDP) [67]. The neighbors' information is considered, all agents' policies are optimized synchronously in training, while only the agent's observation history is used in the execution.

Specifically, a Dec-POMDP could be denoted by a tuple $\mathcal{G} =< \mathcal{S}, \mathcal{N}, \mathcal{A}, \mathcal{O}, \mathcal{Z}, \mathcal{P}, \mathcal{R}, \mathcal{H}, \gamma >$. Each intersection in the scenario is controlled by an agent. $\mathcal{S}$ is the state space, and $s_t \in \mathcal{S}$ denotes the state at time step t. Since the environment is partially observed, each agent i only has access to its local observation $o_{i,t} \in \mathcal{O}$ that is acquired through an observation function $\mathcal{Z}(s) : \mathcal{S} \to \mathcal{O}$. At current time, all agents perform joint action $\boldsymbol{a} = \{a_i, \ldots, a_N\}, a_i \in \mathcal{A}$ and cause the state transition dynamics $\mathcal{P}\left(s_{t+1}|s_t, \boldsymbol{a}\right) := \mathcal{S} \times \mathcal{A}^N \to \mathcal{S}$ where s_{t+1} is the next state. The observation-action history of agent i at time t is denoted as $\tau_{i,:t}$. $\mathcal{R} = \{\mathcal{R}_i\}_{i=1}^N$ is the reward for each agent. As stated in Sec. 3.3, the reward is calculated by the partial observation (queue length) and the observation transition may be unstable in a multi-agent system. That is, even if the agent performs the same action on the same observation at different timesteps, the agent may receive different observation transitions because neighbor agents may perform different actions. Hence, we define the reward function of each agent as $r_{i,t} = \mathcal{R}_i(o_{i,t}, a_i, o_{i,t+1})$. Our goal is to learn the decentralized policy $\pi_i\left(a_i \mid o_{i,t}\right)$ to maximize its cumulative rewards: $\sum_t \gamma^t r_{i,t}$ where γ is the discounted factor. In the traffic signal task, the next observation of each agent not only relies on current agent's observation and performed action, but also is associated with neighbors' actions. Therefore, there exists an observation transition function $\mathcal{T}_i\left(o_{i,t+1} \mid o_{i,t}, a_i, \mathbf{a}^{-i}\right)$ for each agent i where $\mathbf{a}^{-i}$ is the joint action of neighbors.

Based on the above formulation, there are two key issues that need to be considered:

- Since training policy for each realistic scenario is time-consuming even impossible, hence our goal is to learn the decentralized policy $\pi_i\left(a_i \mid o_{i,t}\right)$ from the given training scenario (i.e., a road network), which can be generalized to unseen scenarios. Thus the meta-learning framework is employed where the policy learning of

each agent corresponds to a task. The reward function R_i and observation transition $\mathcal{T}_i$ of these tasks vary but also share similarities since they follow the same traffic rules and have similar optimization goals. Therefore, we represent each task as a learned and low dimensional latent variable m_i. By incorporating m_i, we assume the reward, observation transition and policy functions could be shareable across tasks:

$$\begin{aligned} \mathcal{T}_i\big(o_{i,t+1} \mid o_{i,t}, a_i, \mathbf{a}^{-i}\big) &\approx \mathcal{T}\big(o_{i,t+1} \mid o_{i,t}, a_i, \mathbf{a}^{-i}, m_i\big), \\ \mathcal{R}_i\left(o_{i,t}, a_i, o_{i,t+1}\right) &\approx \mathcal{R}\left(o_{i,t}, a_i, o_{i,t+1}, m_i\right), \\ \pi_i\left(a_i \mid o_{i,t}\right) &\approx \pi\left(a_i \mid o_{i,t}, m_i\right), \end{aligned} \tag{1}$$

which make the meta-learning possible.

- Since $\mathcal{R}_i$ not only rely on $o_{i,t}$ and a_i but also $o_{i,t+1}$ which is generated by $\mathcal{T}_i$ and related with a^{-i}, hence learning π_i from $\mathcal{R}_i$ may cause learning non-stationary because the agent may receive different rewards and observation transitions for the same action at the same observation. In this case, the received rewards and observation transitions of the current agent could not be well predicted only conditioned on its own observations and performed actions. Conversely, to avoid suffering such non-stationary, we hope the learned decentralized policy could make the observation transition and reward predictable. That is, based on the learned $\pi(a_i|o_{i,t})$, there exists functions $\mathcal{T}_i'$ and $\mathcal{R}_i'$ satisfy that:

$$\begin{aligned} \mathcal{T}_i'(o_{i,t+1} \mid o_{i,t}, a_i) &\approx \mathcal{T}_i\big(o_{i,t+1} \mid o_{i,t}, a_i, \mathbf{a}^{-i}\big), \\ \mathcal{R}_i'(o_{i,t+1}, o_{i,t}, a_i) &\approx \mathcal{R}_i\big(o_{i,t+1}, o_{i,t}, a_i, \mathbf{a}^{-i}\big), \\ where\ a_i &\sim \pi(a_i|o_{i,t}). \end{aligned} \tag{2}$$

3.3 Agent Design

Each intersection in the system is controlled by an agent, and here we present the detailed definitions of the agent.

- **Observation.** Each agent has its own local observation, including the number of vehicles on each incoming lane and the current phase of the intersection, where phase is the part of the signal cycle allocated to any combination of traffic movements, as explained in Section 3.1. Observation of agent i is defined by

$$o_{i,t} = \{\mathcal{V}_1, \mathcal{V}_2, \ldots, \mathcal{V}_M, \mathtt{p}\}, \tag{3}$$

  where M is the total number of incoming lanes and $\mathcal{V}_M$ means the number of vehicles in the M^{th} incoming lanes, p is the current phase and represented as a one-hot vector.
- **Action.** At time t, each agent i chooses a phase p as its action a_i, indicating the traffic signal should be set to phase p. Note that the phases may organize in a sequential way in reality, while directly selecting a phase makes the traffic control plan more flexible. In a grid road network, each agent has four phases as described in Section 3.1. For a complex and heterogeneous road network, we unite the phases of all structures together as the full action space, and mask the unavailable phases of each traffic signal as unavailable actions of the agent.
- **Reward.** We define the reward for agent i as the negative of the queue length on incoming lanes. Note that optimizing queue length has been proved to be equivalent to optimizing average travel time in [38] under certain assumptions. Average travel time is a global criteria which cannot be optimized directly by an agent. Hence, queue length is widely used as reward in traffic signal control. Reward of agent i is defined by

$$r_i = -\sum_m^M q_m, \tag{4}$$

  where q_m is the queue length on the incoming lane m, and M is the total number of incoming lanes.

4 METHOD

In this section, we propose Meta Variationally Intrinsic Motivated (MetaVIM) method to achieve Eq. 1 and Eq. 2, as illustrated in Fig. 3. MetaVIM employs latent variable to represent each task to make the reward, observation transition and policy functions shareable. At the same time, MetaVIM makes the approximations and policy robust to neighbors by intrinsic reward design. From now on, we drop the sub- and superscript i to denote the current agent for ease of notation. We start by describing the overall architecture in Sec. 4.1, and then elaborate the policies with latent variable in Sec. 4.2 and intrinsic reward design in Sec. 4.3.

4.1 Model

MetaVIM consists of a multi-head VAE (mVAE) and a policy network, where the mVAE consists of an encoder and four decoders:

- **Encoder.** The encoder takes the past trajectories $\tau_{:t}$ up until t as input to calculate the latent variable m in Eq. 1, parameterised by ψ. The encoder could be:

$$\mathbf{e}^{\psi}\left(m \mid \tau_{:t}\right). \tag{5}$$

- **Decoder.** Four decoders are used to predict the environmental dynamics, and parameterised by $\phi_{\mathrm{r}}, \phi_{\tilde{\mathrm{r}}}, \phi_{\mathrm{o}}, \phi_{\tilde{\mathrm{o}}}$ respectively, where the former two are used to predict next reward with/without neighbors and the latter two are used to predict next observation with/without neighbors:

$$\mathbf{p}^{\phi_{\mathrm{r}}}\left(r_{t+1} \mid o_{t+1}, o_t, a, m\right), \tag{6}$$
$$\mathbf{p}^{\phi_{\mathrm{o}}}\left(o_{t+1} \mid o_t, a, m\right), \tag{7}$$
$$\mathbf{p}^{\phi_{\tilde{\mathrm{r}}}}\left(\tilde{r}_{t+1} \mid o_{t+1}, o_t, a, a_j, m\right), \tag{8}$$
$$\mathbf{p}^{\phi_{\tilde{\mathrm{o}}}}\left(\tilde{o}_{t+1} \mid o_t, a, a_j, m\right), \tag{9}$$

  where j is any neighbor agent of current intersection, a_j is the neighbor action, and m is the latent variable generated by the encoder.
- **Policy.** The policy takes local observations o_t and latent variable as input, parameterised by θ and dependent on ψ, which is defined by

$$\pi^{\theta}\left(a \mid o_t, m\right), \tag{10}$$

  where m is derived by encoder. The policy is conditioned on both observation and posterior over m, which is similar to the formulation of Bayesian RL [68], [69], [70].

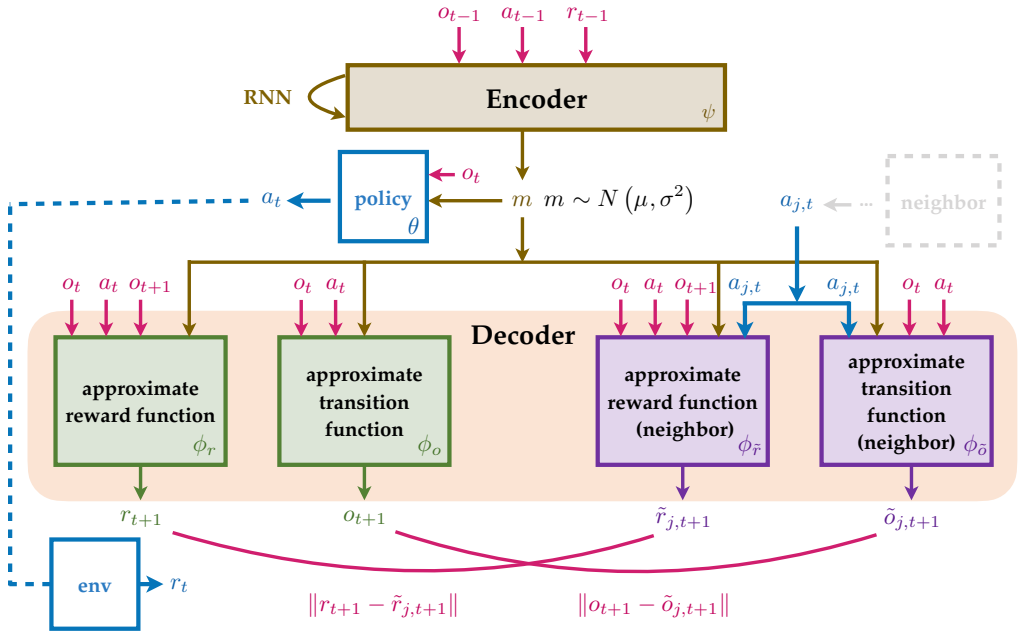


Fig. 3: MetaVIM consists of a mVAE and a policy network. RL agent is augmented with a latent variable m, obtained by encoding over past trajectories $\tau_{:t}$ to represent the task-specific information.

4.2 Latent Variable

4.2.1 Design

From the perspective of meta-RL, there exists a true variable to represent a certain task (i.e., an agent). However, we do not have access to this information. Alternatively, we aim to learn variable m from trajectories of the current task.

Considering the uncertainty of the task, we model the task as a Gaussian distribution $\mathcal{N}(\mu, \sigma^2)$, where the mean μ and the standard deviation σ are generated by the encoder. Specifically, a RNN is employed to track the characteristics of trajectory $\tau_{:t}$ and calculate μ and σ respectively. Then, m is randomly sampled from $\mathcal{N}(\mu, \sigma^2)$, that is, $m \sim \mathcal{N}(\mu, \sigma^2)$. Note $m \sim \mathcal{N}(\mu, \sigma^2)$ is underivable, we alternatively use the reparameterization technique [65] for backpropagation:

$$m = \mu + \sigma m_0, \tag{11}$$

where m_0 is randomly sampled from the standard normal distribution $m_0 \sim N(\mu, I)$, and I is the identity matrix.

4.2.2 Analysis

Besides making the meta-learning possible as shown in Eq. 1, the latent variable brings two additional advantages:

(1) Trade-off of Exploration and Exploitation: Latent variable is an additional input of the policy network except for the observation, which could be regarded as prior knowledge about the task. Previous research [64] has shown that random Gaussian distribution could be regarded as the noise disturbance and provide randomness of the policy. Specifically, the regular policy network outputs action probability based on the current observation only, and the distribution of sampled action is certain once the current observation is given. In contrast, our method takes the latent variable m as the additional input of the policy network besides the observation, where m is randomly sampled from the learnable Gaussian distribution $\mathcal{N}(\mu, \delta^2)$. Hence, m could bring randomness to the distribution of the sampled action in the learning procedure. Hence, it helps to explore the tasks.

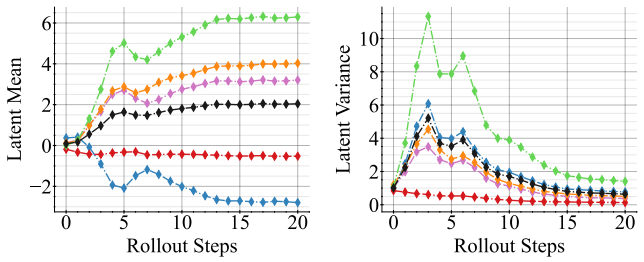
Since our Gaussian distribution changes dynamically in training rather than keeps as constant. It not only helps to explore but also helps to trade off the exploration and exploitation [50]. With the converging of the model, the randomness gradually decreases. Specifically, the mean μ tends to be stable and the variance δ^2 tends to zero. Fig. 4 illustrates the adaptation of latent variable. It has a dynamic guiding effect on policy: randomness is the largest at the beginning, the dispersive distribution of m necessitates the agent to perform diverse actions to explore, and then randomness decreases as the posterior distribution $\mathcal{N}(\mu, \sigma^2)$ gradually converges. In this phase, the policy network encourages the agent to exploit the environment. In addition, it also makes the final learned policy stable with very little randomness. This can be analogous to the impact of ϵ-greedy in DQN on trade-off of the exploration and exploitation. The intuitive idea is: random noise (random latent variable) is equivalent to the fixed ϵ in ϵ-greedy, and the learnable Gaussian distribution (learnable latent variable) is equivalent to the adjustable ϵ in ϵ-greedy, which has a dynamic guiding effect on the trade-off of the exploration and exploitation.

(2) Modelling Latent Coordination with Neighbors: In our method, the neighbor information is available only in training, and the decoders are abandoned in execution. Only using individual observation as the input of policy may ignore the latent neighbors information. As shown in Eq. 1, the observation transition is caused by not only the current agent but also its neighbors. In turn, the latent variable could reflect the latent neighbor's information.

4.3 Intrinsic Reward

4.3.1 Design

As stated in Eq. 2, the non-stationary learning often causes the observation transition and received rewards unpre-



(a) Latent Mean (b) Latent Variance

Fig. 4: Visualisation of the latent space. (a) and (b) denote the mean and variance of the latent variable respectively. The latent variable has 5 dimensions, each line is one latent dimension, the black line is the average.

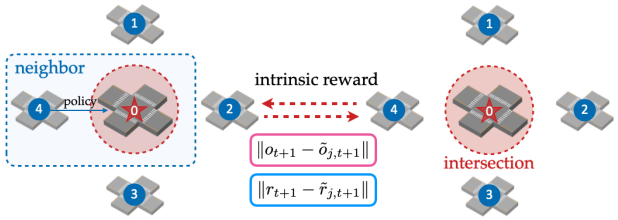


Fig. 5: Illustration of intrinsic reward design.

dictable only conditioned on individual observation and action. Conversely, we hope the learned policy makes them be predicted stably. To achieve this goal, we design a novel intrinsic reward based on VAE, as illustrated in Fig. 5, which is the negative of the prediction bias with/without neighbor agents' policies:

$$r_t^{int} = -\sum_j \left(\|r_{t+1} - \tilde{r}_{j,t+1}\| + \|o_{t+1} - \tilde{o}_{j,t+1}\| \right), \quad (12)$$

where r_{t+1} and $\tilde{r}_{j,t+1}$ are the predicted rewards with/without neighbor agents' policies respectively, o_{t+1} and $\tilde{o}_{j,t+1}$ are predicted next observations with/without neighbor agent's policies respectively. They are generated by the corresponding decoders in Eq. 6.

4.3.2 Analysis

Here, will give an interpretation from the perspective of information theory why the intrinsic reward design can lead to a stable policy robust to neighbors' policy, which could make the learned policy easy to transfer. From the perspective of the current agent, the r_{t+1} is defined as

$$r_{t+1} = \mathcal{R}\left(o_{t+1}, o_t, \mathbf{u}_t, a_t; m\right), \quad (13)$$

for any neighbor agent j, the unobservable part $\mathbf{u}_t$ consists of two parts, the part containing j and the part not containing j, that is $\mathbf{u}_t = \{o_{j,t}, \mathbf{u}_t^{-j}\}$, and $a_{j,t} \sim \pi_j$. Then add the neighbor agent's policy, we can obtain:

$$\tilde{r}_{j,t+1} = \mathcal{R}(o_{t+1}, o_t, \underbrace{o_{j,t}, \mathbf{u}_t^{-j}}_{\text{unobs.}}, a_t, a_{j,t}; m). \quad (14)$$

Similar, o_{t+1} is defined as

$$o_{t+1} = \mathcal{T}\left(o_{t+1} \mid o_t, a_t, \mathbf{u}_t; m\right), \quad (15)$$

we split the unobservable part $\mathbf{u}_{t+1}$ into the part containing j and the part not containing j, then add the neighbor agent's policy, we have

$$\tilde{o}_{j,t+1} = \mathcal{T}(o_{t+1} \mid o_t, a_t, \underbrace{o_{j,t}, \mathbf{u}_t^{-j}}_{\text{unobs.}}, a_{j,t}; m). \quad (16)$$

According to Eq. 12, our intrinsic reward is:

$$\begin{aligned} r_t^{int} &= -\sum_j \left(\|r_{t+1} - \tilde{r}_{j,t+1}\| + \|o_{t+1} - \tilde{o}_{j,t+1}\| \right) \quad (17) \\ &= -\sum_j \Big(\big\| \mathcal{R}\left(o_{t+1}, o_t, \mathbf{u}_t, a_t; m\right) - \\ &\qquad \mathcal{R}(o_{t+1}, o_t, o_{j,t}, \mathbf{u}_t^{-j}, a_t, a_{j,t}; m) \big\|_2^2 \\ &\quad + \big\| \mathcal{T}\left(o_{t+1} \mid o_t, a_t, \mathbf{u}_t; m\right) - \\ &\qquad \mathcal{T}(o_{t+1} \mid o_t, a_t, o_{j,t}, \mathbf{u}_t^{-j}, a_{j,t}; m) \big\|_2^2 \Big) \\ &= -\sum_j \Big(\big\| \mathcal{R}\left(r_{t+1} \mid z_t^2\right) - \mathcal{R}\left(r_{t+1} \mid a_{j,t}, z_t^2\right) \big\|_2^2 + \\ &\qquad \big\| \mathcal{T}\left(o_{t+1} \mid z_t^1\right) - \mathcal{T}\left(o_{t+1} \mid a_{j,t}, z_t^1\right) \big\|_2^2 \Big) \end{aligned}$$

where z_t^1 represents the conditioning variable 1 at timestep t, $z_t^1 = \langle o_t, a_t, \mathbf{u}_t, m \rangle$, and z_t^2 represents the conditioning variable 2 at timestep t, $z_t^2 = \langle o_{t+1}, o_t, \mathbf{u}_t, a_t, m \rangle$. The mutual information(MI) between the predicted state o_{t+1} and another agent's policy a_j is:

$$\begin{aligned} & I\left(o_{t+1}; a_{j,t} \mid z_t^1\right) \quad (18) \\ &= \sum_{o_{t+1}, a_{j,t}} p\left(o_{t+1}, a_{j,t} \mid z_t^1\right) \log \frac{p\left(o_{t+1}, a_{j,t} \mid z_t^1\right)}{p\left(o_{t+1} \mid z_t^1\right) p\left(a_{j,t} \mid z_t^1\right)} \\ &= \sum_{o_{t+1}, a_{j,t}} p\left(o_{t+1} \mid a_{j,t}, z_t^1\right) \cdot p\left(a_{j,t} \mid z_t^1\right) \cdot \\ &\qquad \log \frac{p\left(o_{t+1} \mid a_{j,t}, z_t^1\right) \cdot p\left(a_{j,t} \mid z_t^1\right)}{p\left(o_{t+1} \mid z_t^1\right) p\left(a_{j,t} \mid z_t^1\right)} \\ &= \sum_{a_{j,t}} p\left(a_{j,t} \mid z_t^1\right) D_{\mathrm{KL}}\left[p\left(o_{t+1} \mid a_{j,t}, z_t^1\right) \| p\left(o_{t+1} \mid z_t^1\right)\right] \\ &= \sum_{a_{j,t}} p\left(a_{j,t} \mid z_t^1\right) D_{\mathrm{KL}}\left[T\left(o_{t+1} \mid a_{j,t}, z_t^1\right) \| T\left(o_{t+1} \mid z_t^1\right)\right]. \end{aligned}$$

The mutual information(MI) between the predicted reward r_{t+1} and another agent's policy $a_{j,t}$ is as follows:

$$\begin{aligned} & I\left(r_{t+1}; a_{j,t} \mid z_t^2\right) \quad (19) \\ &= \sum_{r_{t+1}, a_{j,t}} p\left(r_{t+1}, a_{j,t} \mid z_t^2\right) \log \frac{p\left(r_{t+1}, a_{j,t} \mid z_t^2\right)}{p\left(r_{t+1} \mid z_t^2\right) p\left(a_{j,t} \mid z_t^2\right)} \\ &= \sum_{r_{t+1}, a_{j,t}} p\left(r_{t+1} \mid a_{j,t}, z_t^2\right) \cdot p\left(a_{j,t} \mid z_t^2\right) \\ &\qquad \log \frac{p\left(r_{t+1} \mid a_{j,t}, z_t^2\right) \cdot p\left(a_{j,t} \mid z_t^2\right)}{p\left(r_{t+1} \mid z_t^2\right) p\left(a_{j,t} \mid z_t^2\right)} \\ &= \sum_{a_{j,t}} p\left(a_{j,t} \mid z_t^2\right) D_{\mathrm{KL}}\left[p\left(r_{t+1} \mid a_{j,t}, z_t^2\right) \| p\left(r_{t+1} \mid z_t^2\right)\right] \\ &= \sum_{a_{j,t}} p\left(a_{j,t} \mid z_t^2\right) D_{\mathrm{KL}}\left[R\left(r_{t+1} \mid a_{j,t}, z_t^2\right) \| R\left(r_{t+1} \mid z_t^2\right)\right]. \end{aligned}$$

Combine the Eq. 18 and Eq. 19, we have

$$
\begin{aligned}
& I\left(o_{t+1}; a_{j,t} \mid z_t^1\right) + I\left(r_{t+1}; a_{j,t} \mid z_t^2\right) \qquad (20)\\
&= \sum_{a_{j,t}} p\left(a_{j,t} \mid z_t^1\right) D_{\mathrm{KL}}\left[T\left(o_{t+1} \mid a_{j,t}, z_t^1\right) \| T\left(o_{t+1} \mid z_t^1\right)\right] \\
&+ \sum_{a_{j,t}} p\left(a_{j,t} \mid z_t^2\right) D_{\mathrm{KL}}\left[R\left(r_{t+1} \mid a_{j,t}, z_t^2\right) \| R\left(r_{t+1} \mid z_t^2\right)\right],
\end{aligned}
$$

By sampling N independent trajectories τ_N from the environment, we perform a Monte-Carlo approximations of the MI:

$$
\begin{aligned}
& I\left(o_{t+1}; a_{j,t} \mid z_t^1\right) + I\left(r_{t+1}; a_{j,t} \mid z_t^2\right) \qquad (21)\\
&= \mathbb{E}_\tau\left[D_{\mathrm{KL}}\Big[T\left(o_{t+1} \mid a_{j,t}, z_t^1\right) \| T\left(o_{t+1} \mid z_t^1\right)\Big] \mid z_t^1\right] + \\
&\quad \mathbb{E}_\tau\left[D_{\mathrm{KL}}\Big[R\left(r_{t+1} \mid a_{j,t}, z_t^2\right) \| R\left(r_{t+1} \mid z_t^2\right)\Big] \mid z_t^2\right], \\
&\approx \frac{1}{N} \sum_N \Big(\left\|T\left(o_{t+1} \mid a_{j,t}, z_t^1\right) - T\left(o_{t+1} \mid z_t^1\right)\right\| + \\
&\quad \left\|R\left(r_{t+1} \mid a_{j,t}, z_t^2\right) - R\left(r_{t+1} \mid z_t^2\right)\right\|\Big).
\end{aligned}
$$

The last approximation is because the predictions in our implementation are deterministic rather than stochastic.

Thus, in expectation, the intrinsic reward is the negative of MI above. As each agent maximizes the long-term cumulative reward, which therefore minimizes MI. As a result, agents become independent. This can be an interpretation from the information-theoretical perspective. Note that the prediction results are only used to form intrinsic rewards, and our method tries to minimize them. That means our method mainly relies on the trend of change of predicted results, not the predicted value. Therefore, we expect our method is resilient to the decoders' modeling error accumulation.

4.4 Learning

The parameters of MetaVIM consist of a policy network $\pi^\theta\left(a_t \mid o_t, m\right)$ and a mVAE which includes a encoder $\mathbf{e}^\psi\left(m \mid \tau_{:t}\right)$ and four decoders $\mathbf{p}^{\phi_{\mathrm{r}}}\left(r_{t+1} \mid o_{t+1}, o_t, a_t, m\right)$, $\mathbf{p}^{\phi_o}\left(o_{t+1} \mid o_t, a_t, m\right)$, $\mathbf{p}^{\phi_{\tilde{\mathrm{r}}}}\left(\tilde{r}_{t+1} \mid o_{t+1}, o_t, a_t, a_{j,t}, m\right)$ and $\mathbf{p}^{\phi_{\tilde{o}}}\left(\tilde{o}_{t+1} \mid o_t, a_t, a_{j,t}, m\right)$, where $\theta, \psi, \phi_{\mathrm{r}}, \phi_{\mathrm{o}}, \phi_{\tilde{r}}$ and $\phi_{\tilde{o}}$ are the corresponding network weights. The learning mainly contains two parts:

(1) Maximizing the Cumulative Rewards: For the policy network θ, the learning objective is to maximize the cumulative reward as well as the intrinsic reward r_t^{int} in Eq. 12:

$$
\max_\theta J(\theta) = \mathbb{E}_{\substack{a_t \sim \pi^\theta(a_t|o_t,m) \\ m \sim \mathcal{N}(\mu,\sigma^2)}} \sum_{t=0}^{\mathcal{H}} (r_t + \alpha r_t^{int}), \qquad (22)
$$

where α is a positive weight. To achieve Eq. 22, the Proximal Policy Optimization (PPO) [71] is employed. PPO is an on-policy actor-critic method which adds a soft constraint that can be optimized by a first-order optimizer, effectively using existing data to take step that leads to the biggest possible improvement on a policy, without stepping too far to accidentally cause performance collapse.

(2) Training mVAE: For current agent, the past trajectory up to time step t is collected as

$$
\tau_{:t} = \left(o_0, a_0, r_1, o_1, a_1, r_2, \ldots, o_{t-1}, a_{t-1}, r_t, o_t\right), \qquad (23)
$$

Algorithm 1 MetaVIM: Meta-training Phase

Require: A set of meta-training tasks $\{\kappa_i\}_{i \in \mathcal{N}}$ drawn from intersections $\mathcal{N} = 1, 2, \ldots, N$ in a multi-agent scenario, number of training episodes E
Initialize policy replay buffers $\mathcal{B}^\pi$
Initialize mVAE replay buffers $\mathcal{B}^{\mathrm{mVAE}}$
Initialize encoder $\mathbf{e}^\psi$, decoders $\mathbf{p}^{\phi_{\mathrm{r}}}, \mathbf{p}^{\phi}, \mathbf{p}^{\phi_{\tilde{\mathrm{r}}}}, \mathbf{p}^{\phi_{\tilde{o}}}$
Initialize policy π^θ
while not done **do**
  for episodes$= 1, 2, \ldots, E$ **do** {Data collection}
    Compute latent variable m from the current rollouts
    Add latent variable m to $\mathcal{B}^\pi$
    for steps in training steps **do** {Training}
      Take action according to π^θ
      Get neighbor actions $\{a_j\}_{j=1,\ldots,J}$
      Act to the environment and get environmental reward r
      Compute intrinsic reward r^{int} using Eq. (12)
      Update latent variable m by $\mathbf{e}^\psi$
      Add trajectories τ and latent variable m to $\mathcal{B}^{\mathrm{mVAE}}$
      Add trajectories τ to $\mathcal{B}^\pi$
      Train encoder $\mathbf{e}^\psi$ and decoders $\mathbf{p}^{\phi_{\mathrm{r}}}, \mathbf{p}^{\phi}, \mathbf{p}^{\phi_{\tilde{\mathrm{r}}}}, \mathbf{p}^{\phi_{\tilde{o}}}$ by maximising ELBO in Eq. (26)
      Train policy π^θ by maximizing reward in Eq. (22)
      clean up $\mathcal{B}^\pi$
    end for
  end for
end while

Algorithm 2 MetaVIM: Meta-testing Phase

Require: A set of meta-testing tasks $\{\kappa_i\}_{i \in \mathcal{M}}$ drawn from intersections $\mathcal{M} = 1, 2, \ldots, M$ in a multi-agent scenario, number of testing episodes E
Load policy model π^θ
Load encoder model $\mathbf{e}^\psi$
for episodes=1,2, ...,E **do** {Rollouts}
  Compute latent variable m from the current rollouts
  for step in testing steps **do**
    Take action according to π^θ
    Act to the environment
    Compute latent variable m by $\mathbf{e}^\psi$
  end for
end for
for step in testing steps **do** {Testing}
  Take action according to $\pi^\theta(a \mid o, m)$
end for

where $a_t \sim \pi^\theta\left(a_t \mid o_t, \mathbf{e}^\psi\left(m \mid \tau_{:t}\right)\right)$. Based on $\tau_{:t}$, at any given time step t, our mVAE learning objective is thus to maximise

$$
\mathbb{E}_{\tau_{:t} \sim \rho}\left[\log \mathbf{p}^{\phi_r, \phi_{\tilde{r}}, \phi_o, \phi_{\tilde{o}}}(\tau_{:t})\right], \qquad (24)
$$

Since the original optimized objective of VAE can't be optimized directly, instead of optimising Eq. 24, we use a variational evidence lower bound (ELBO) [65] which is widely used to optimize the VAE-based network. Maximizing ELBO can approximately maximize the $\log \mathbf{p}^{\phi_r, \phi_{\tilde{r}}, \phi_o, \phi_{\tilde{o}}}(\tau_{:t})$,

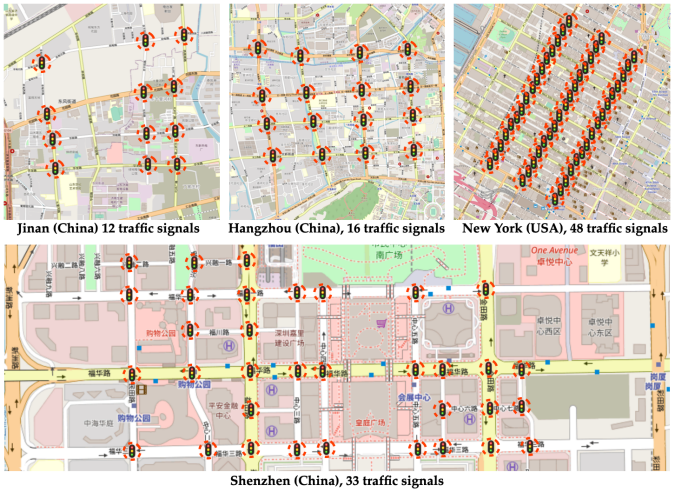


Fig. 6: The illustration of the road networks. The first row shows the road networks of Jinan (China), Hangzhou (China) and New York (USA), containing 12, 16 and 48 traffic signals respectively, and the second row shows the road network of Shenzhen containing 33 traffic signals.

the ELBO can be derived as follows:

$$
\begin{aligned}
&\mathbb{E}_{\tau_{:t}\sim\rho}\left[\log \mathbf{p}^{\phi_r,\phi_{\tilde{r}},\phi_o,\phi_{\tilde{o}}}(\tau_{:t})\right] \qquad (25)\\
&=\mathbb{E}_{\tau_{:t}\sim\rho}\Big[\log \mathbf{p}^{\phi_r}(r_{t+1})+\log \mathbf{p}^{\phi_{\tilde{r}}}(r_{t+1})\\
&\quad+\log \mathbf{p}^{\phi_o}(o_{t+1})+\log \mathbf{p}^{\phi_{\tilde{o}}}(o_{t+1})\Big]\\
&=\mathbb{E}_{\tau_{:t}\sim\rho}\left[\log \mathbb{E}_{\mathbf{e}_\psi(m|\tau_{:t})}[\frac{\mathbf{p}^{\phi_r}(r_{t+1},m)}{\mathbf{e}_\psi(m|\tau_{:t})}\right.\\
&\quad\left.+\frac{\mathbf{p}^{\phi_{\tilde{r}}}(r_{t+1},m)}{\mathbf{e}_\psi(m|\tau_{:t})}+\frac{\mathbf{p}^{\phi_o}(o_{t+1},m)}{\mathbf{e}_\psi(m|\tau_{:t})}+\frac{\mathbf{p}^{\phi_{\tilde{o}}}(o_{t+1},m)}{\mathbf{e}_\psi(m|\tau_{:t})}]\right]\\
&\geq\mathbb{E}_{\tau_{:t}\sim\rho,\ \mathbf{e}_\psi(m|\tau_{:t})}\left[\log\frac{\mathbf{p}^{\phi_r}(r_{t+1},m)}{\mathbf{e}_\psi(m|\tau_{:t})}+\log\frac{\mathbf{p}^{\phi_{\tilde{r}}}(r_{t+1},m)}{\mathbf{e}_\psi(m|\tau_{:t})}\right.\\
&\quad\left.+\log\frac{\mathbf{p}^{\phi_o}(o_{t+1},m)}{\mathbf{e}_\psi(m|\tau_{:t})}+\log\frac{\mathbf{p}^{\phi_{\tilde{o}}}(o_{t+1},m)}{\mathbf{e}_\psi(m|\tau_{:t})}\right]\\
&=\mathbb{E}_{\tau_{:t}\sim\rho,\ \mathbf{e}_\psi(m|\tau_{:t})}\Big[\log \mathbf{p}^{\phi_r}(r_{t+1}|m)+\log \mathbf{p}^{\phi_{\tilde{r}}}(r_{t+1}|m)\\
&\quad+\log \mathbf{p}^{\phi_o}(o_{t+1}|m)+\log \mathbf{p}^{\phi_{\tilde{o}}}(o_{t+1}|m)-\log \mathbf{e}_\psi(m|\tau_{:t})\Big]\\
&=\mathbb{E}_{\mathbf{e}_\psi(m|\tau_{:t})}\sum_{t'=0}^{t-1}(\log \mathbf{p}^{\phi_r}(r_{t+1})+\log \mathbf{p}^{\phi_{\tilde{r}}}(r_{t+1})\\
&+\log \mathbf{p}^{\phi_o}(o_{t+1})+\log \mathbf{p}^{\phi_{\tilde{o}}}(o_{t+1}))-KL(\mathbf{e}_\psi(m|\tau_{:t})||q(m)).\\
&=ELBO(\psi,\phi_r,\phi_{\tilde{r}},\phi_o,\phi_{\tilde{o}}|\tau_{:t}).
\end{aligned}
$$

Then the mVAE learning objective is thus to maximise:

$$
\begin{aligned}
&ELBO(\psi,\phi_r,\phi_{\tilde{r}},\phi_o,\phi_{\tilde{o}}|\tau_{:t})\\
&=\mathbb{E}_{\mathbf{e}^\psi(m|\tau_{:t})}\sum_{t=0}^{t-1}(\log \mathbf{p}^{\phi_r}(r^\tau_{t+1})+\log \mathbf{p}^{\phi_{\tilde{r}}}(r^\tau_{t+1})\\
&\quad+\log \mathbf{p}^{\phi_o}(o^\tau_{t+1})+\log \mathbf{p}^{\phi_{\tilde{o}}}(o^\tau_{t+1}))\\
&\quad-KL(\mathbf{e}^\psi(m|\tau_{:t})||q(m)). \qquad (26)
\end{aligned}
$$

The first 4 terms are often referred to as the reconstruction loss, and the term $KL(\mathbf{e}^\psi(m|\tau_{:t})||q(m))$ is the KL-divergence between the variational posterior and the prior distribution $q(m)$, which is set to standard normal distribution $\mathcal{N}(0,\mathcal{I})$ initially, where $\mathcal{I}$ means the identity matrix. Suppose ρ_i is the trajectory distribution induced by our policy and the transition function $\mathcal{T}(\cdot)$, then the learning objective of the mVAE is to maximize the ELBO over ρ:

$$
\max_{\psi,\phi_r,\phi_{\tilde{r}},\phi_o,\phi_{\tilde{o}}}\mathbb{E}_{\tau_{:t}\sim\rho}ELBO(\psi,\phi_r,\phi_{\tilde{r}},\phi_o,\phi_{\tilde{o}}|\tau_{:t}). \qquad (27)
$$

The overall training algorithm is provided in Alg. 1. In our experiments, we train the policy and the mVAE using different replay buffers: $\mathcal{B}^\pi$ only collects recent trajectories since we use on-policy RL algorithms, and for the $\mathcal{B}^{\text{mVAE}}$ we maintain a larger buffer of observed trajectories. At meta-test time, we roll out the policy to evaluate the performance, meta-testing procedure is provided in Alg. 2. The decoders are not used at test time, and no gradient adaptation is done: θ and ψ are shared across different tasks and the policy has learned to act approximately optimal during meta-training. See Appendix B for implementation details.

5 Experiments

We conduct the experiments on CityFlow [20], an city-level open-source simulation platform for traffic signal control. The simulator is used as the environment to provide state for traffic signal control, the agents execute actions by changing the phase of traffic lights, and the simulator returns feedback. Specifically, we conduct additional experiments on another platform SUMO[1] and under different lane configurations to demonstrate the robustness of the method in Appendix D and Appendix C respectively.

5.1 Datasets

5.1.1 Road Networks

The evaluation scenarios come from four real road network maps of different scales, including **Hangzhou** (China), **Jinan** (China), **New York** (USA) and **Shenzhen** (China), illustrated in Fig. 6. The road networks and data of Hangzhou, Jinan and New York are from the public datasets[2]. The road network map of Shenzhen is made by ourselves which is derived from OpenStreetMap[3]. The road networks of Jinan and Hangzhou contain 12 and 16 intersections in 4×3 and 4×4 grids, respectively. The road network of New York includes 48 intersections in 16×3 grid. The road network of Shenzhen contains 33 intersections, which is not grid compared to other three maps, illustrated in Fig. 7. In addition, an additional experiment are conducted on a much larger road network to validate the scalability, more details are listed in Appendix E.

5.1.2 Flow configurations

We run the experiments under three traffic flow configurations: real traffic flow, mixed low traffic flow and mixed high traffic flow. The real traffic flow is real-world hourly statistical data with slight variance in vehicle arrival rates, as shown in Tab. 1. Since the real-world strategies tend to break down during bottleneck period (peak hour), to better

1. http://sumo.dlr.de/index.html
2. https://traffic-signal-control.github.io/
3. https://github.com/zhuliwen/RoadnetSZ.

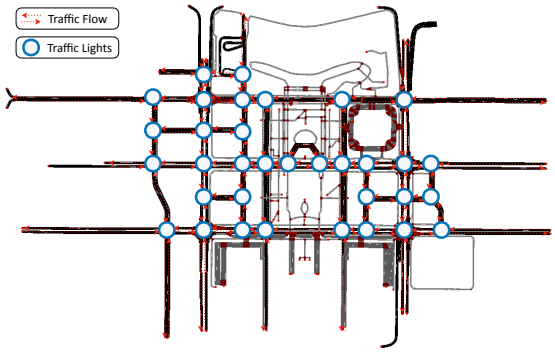


Fig. 7: Overview of the Shenzhen (China) road network. There are total 33 traffic signal intersections across the area.

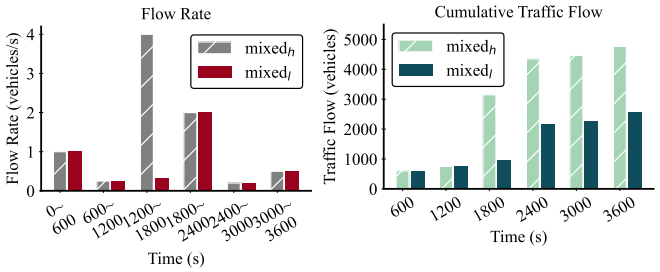


Fig. 8: Illustration of mixed flow. The left figure shows the arrival rate of vehicles, and the right figure shows the cumulative number of vehicles arriving in one hour.

evaluate the performances of traffic light control methods in the flat-peak-flat scenario, we use two synthetic datasets: mixed high and mixed low traffic flow, which have a more dramatic variance in vehicle arrival rates, as shown in Tab. 2. A detailed description of traffic flow configurations is:

- ***Real.*** The traffic flows of **Hangzhou** (China), **Jinan** (China) and **New York** (USA) are from the public datasets[4], which are processed from multiple sources. The traffic flow of **Shenzhen** (China) is made by ourselves generated based on the traffic trajectories collected from 80 red-light cameras and 16 monitoring cameras in a hour. The data statistics are listed in Tab. 1 and Tab. 3.
- **Mixed$_l$.** The mixed$_l$ is a mixed low traffic flow with a total flow of 2550 in one hour, to simulate a light peak. The arrival rate changes every 10 minutes, which is used to simulate the uneven traffic flow distribution in the real world, the details of the vehicle arrival rate and cumulative traffic flow are shown in Fig. 8.
- **Mixed$_h$.** The mixed$_h$ is a mixed high traffic flow with a total flow of 4770 in one hour, in order to simulate a heavy peak. The difference from the mixed$_l$ setting is that the arrival rate of vehicles during 1200-1800s increased from 0.33 vehicles/s to 4.0 vehicles/s. The data statistics are listed in Tab. 2.

5.2 Evaluation Criteria

Following existing studies [13], [14], [40], [41], [46], we use the **average travel time** to evaluate the performance of different methods for traffic signal control. The average travel time indicates the overall traffic situation in an area over a period of time. For a detailed definition of average travel time, see

4. https://traffic-signal-control.github.io/

TABLE 1: Arrival rate of real-world traffic dataset

Dataset	# Intersections	Arrival rate (vehicles/300s)			
		Mean	Std	Max	Min
Hangzhou	16 (4 × 4)	248.58	42.25	333	212
Jinan	12 (4×3)	524.58	102.91	672	256
NewYork	48 (16×3)	235.33	5.84	244	224
Shenzhen	33 (Non-grid)	147.92	79.35	255	22

TABLE 2: Data statistics of synthetic traffic dataset

Dataset	Total vehicles	Arrival rate (vehicles/300s)			
		Mean	Std	Max	Min
mixed$_l$	2570	214.17	198.41	600	60
mixed$_h$	4770	397.50	420.75	1200	60

Section 3.1. Since the number of vehicles and the origin-destination (OD) positions are fixed, better traffic signal control strategies result in less average travel time.

5.3 Testing Mode

The method is evaluated in two modes: **(1) Common Testing Mode**: the model trained on one scenario with one traffic flow configuration is tested on the same scenario with the same configuration. It is used to validate the ability of the RL algorithm to find the optimal policy. **(2) Meta-Test Mode**: we train the model in the Hangzhou road network and transfer the model to the other three networks directly. It is used to validate the generality of the model.

5.4 Baselines

Our method is compared with the following two categories of methods: conventional transportation methods and RL methods[5]. Note that for a fair comparison all the RL methods are learned without any pre-trained parameters and the methods are evaluated under the same settings. The results are obtained by running the source codes[6]. All the baselines are run with three random seeds, and the mean is taken as the final result. The action interval is five seconds for each method, and the horizon is 3600 seconds for each episode. Specifically, the compared methods contain:

5.4.1 *Conventional methods*

- **Random** selects available phase randomly.
- **MaxPressure [2]** is a leading conventional method, which greedily chooses the phase with the maximum pressure. The pressure is defined as the difference of vehicle density between the incoming lane and the outgoing lane.
- **Fixedtime [1]** with random offset [72] executes each phase in a phase loop with a pre-defined span of phase duration, which is widely used for steady traffic.
- **FixedtimeOffset [1]** where multiple intersections use the same synchronized fix-time plan. The offset is the time interval between the start time of the green light among intersections.

5. Some existing RL based methods, such as AttendLight [42] and SD-MaCAR [3], evaluate their methodS under different experimental settings (e.g., road network or traffic flow), and the source codes are not available yet. Therefore, they are not compared.

6. https://github.com/traffic-signal-control/RL_signals

TABLE 3: Route statistics of real-world traffic dataset

Dataset	# Intersections	Route (lanes/route)			
		Mean	Std	Max	Min
Hangzhou	16 (4 × 4)	4.65	2.15	16	2
Jinan	12 (4×3)	4.37	1.88	17	2
NewYork	48 (16×3)	10.00	4.63	21	3
Shenzhen	33 (Non-grid)	7.57	3.91	41	1

- **SlidingFormula [72]** is designed based on the expert experience, which dynamically divides the duration of each phase according to the number of vehicle arriving.
- **SOTL [23]** specifies a pre-defined threshold for the number of waiting vehicles on approaching lanes. Once the waiting vehicles exceeds the threshold, it will switch to the next phase.

5.4.2 RL-based methods

- **Individual RL [12]** where each intersection is controlled by one agent. The replay buffer and network parameters are not shared, and the model update is independent. There is no information transfer between agents.
- **MetaLight [14]** is a value-based meta RL method via parameter initialization based on MAML [73]. MetaLight is originally a single-agent approach for meta-learning on multiple separate tasks. Here we extend it to a multi-agent scenario without considering neighbor information.
- **PressLight [13]** combines the traditional traffic method MaxPressure [2] with RL together, and optimizes the pressure of each intersection.
- **CoLight [46]** uses graph convolution and attention mechanism to model the neighbor information, and then further uses this neighbor information to optimize the queue length.
- **GeneraLight [49]** is a meta RL method which uses generative adversarial network to generate diverse traffic flows and uses them to build training environments.

5.5 Performance Comparison

5.5.1 Evaluation on Common Testing

Tab. 4 lists the comparative results on the common testing mode, and it is evident that:

1) In general, RL methods perform better than conventional methods, and it indicates the advantage of the RL. The reason is that the conventional methods often rely on prior knowledge which may fails in some cases. A typical case is MaxPressure. It shows good performances on several cases including Hangzhou with the *real* configuration, Jinan with the *real*,$mixed_l$ configurations, NewYork with the *real*,$mixed_l$ configurations, and Shenzhen with the *real*,$mixed_l$ configurations. However, it dramatically drops in other scenarios. That is, once the scenarios meet the assumption well, the method performs well and vice versa. Moreover, MetaVIM is comprehensively superior to other methods clearly in all scenarios and configurations, which demonstrates the effectiveness of the method.

2) MetaVIM shows good generalization for different scenarios and configurations. MetaVIM performs the second best in Hangzhou with the $mixed_l$ configuration, Jinan with the *real* configuration and Shenzhen with the $mixed_l$ configuration, and performs best in other scenarios. Overall, MetaVIM has the best mean performance. Except MaxPressure analysed above, GeneraLight achieves the best in Hangzhou with the $mixed_l$ configuration, while performs poorly in other scenarios. The reason is that GeneraLight trains several models on diverse generated traffic flows, and select the model in testing by matching the flow. Hence, it limits the generalization once the testing flow differs largely from the training flows.

3) MetaVIM outperforms Individual RL, MetaLight and PrssLight with 827, 423 and 411, respectively. The main reason is that they learn the traffic signal's policy only using its own observation and ignore the influence of the neighbors, while MetaVIM considers the neighbors as the unobserved part of the current signal to help learning. In addition, Individual RL performs relatively worst in all scenarios because it employs independent replay buffer and neural network parameters among agents. In traffic signal control task, different signals vary but also share similarities since they follow the same traffic rules and have similar optimization goals. Hence, sharing the replay buffer and neural network is helpful.

4) The neighbors' information is modeled in CoLight and it performs well.It indicates modeling neighbors is critical for the coordination. The results of MetaVIM is superior to CoLight on each scenario and configuration, resulting mean 43 improvement. Compared to Colight, MetaVIM proposes an intrinsic reward to help the policies learning stable, and use latent variable to better trade off the exploration and exploitation. In addition, Colight needs the agents' communications in testing, which is unnecessary in MetaVIM. It makes MetaVIM easy to deploy.

5.5.2 Evaluation on Meta-Test

The comparative results evaluated on the meta-test mode are shown in Tab. 5. The "original" means the model is trained on the current testing scenario, and the "transfer" stands for the model is trained on the road map of Hangzhou. From the results, we can obtain follow findings:

1) Colight needs full state information in both training and testing, hence it cannot be used for a new scenario which contains different number intersections compared with the training scenario. That is, the heterogeneous scenarios will cause heterogeneous inputs of the policy network, which makes the network fail to work. Hence, its results are unavailable in this setting. In contrast, the input of decentralized policy is the current agent's observation, and it is easy to adapt to a new scenario. Therefore, it is necessary to develop the decentralized policy for cross-scenario generalization.

2) The performances of Individual RL and PressLight drop 38% and 41% when the model is transferred. It shows that the models learned by the regular RL algorithms indeed rely on the training scenario. MetaLight is more robust to various scenarios than Individual RL and PressLight, and it indicates the advantage of the meta-learning framework. The meta-learning framework could help to learn task-shared model. Overall, MetaVIM achieves the state-of-the-art performances and only drops 9% when transferring the model. The main reason is that: the task-specific information is modeled by the latent variable in our method, and the learned policy function could be adaptive to diverse latent

TABLE 4: Performance on the Common Testing Mode. The smaller average travel times (s/veh) mean the better performances.

Model	Hangzhou			Jinan			Newyork			Shenzhen			Mean
	real	mixed$_l$	mixed$_h$	real	mixed$_l$	mixed$_h$	real	mixed$_l$	mixed$_h$	real	mixed$_l$	mixed$_h$	
Random	727	1721	1794	836	1547	1733	1858	1865	2105	728	1775	1965	1554
MaxPressure	416	2449	2320	**355**	839	1218	380	488	1481	**389**	753	1387	1039
Fixedtime	718	1756	1787	814	1532	1739	1849	1865	2086	786	1705	1845	1540
FixedtimeOffset	736	1755	1725	854	1553	1720	1919	1901	2141	798	1886	2065	1588
SlidingFormula	441	1102	1241	576	759	1251	1096	986	1656	452	876	1347	982
SOTL	1209	2051	2062	1453	1779	1991	1890	1923	2140	1376	1902	2098	1823
Individual RL	743	1704	1819	843	1552	1745	1867	1869	2100	769	1753	1845	1551
MetaLight	480	1465	1576	784	984	1854	261	482	2145	694	954	2083	1147
PressLight	529	1538	1754	809	1173	1930	302	437	1846	639	834	1832	1135
CoLight	297	960	1077	511	733	1217	159	305	1457	438	657	1367	767
GeneraLight	335	**798**	1575	586	1257	1616	1209	1257	1686	792	916	1574	1133
Base	526	1501	1674	793	1093	1904	298	432	1793	593	813	1732	1096
Base+m	302	923	1643	593	837	1532	152	298	1834	426	676	1630	906
Base+m+tran_RS	348	1134	1289	694	863	1406	189	368	1783	472	694	1416	888
Base+m+rew_RS	338	1049	1274	683	849	1375	173	327	1738	453	683	1372	860
MetaVIM	**284**	893	**986**	492	**694**	**1189**	**149**	**288**	**1387**	408	**622**	**1272**	**724**

TABLE 5: Comparisons on the Meta-Test Mode. The *original* means the model is trained on the current testing scenario, and the *transfer* stands for the model is trained on the road map of Hangzhou.

Model	Jinan			Newyork			Shenzhen			Decline ratio
	real	mixed$_l$	mixed$_h$	real	mixed$_l$	mixed$_h$	real	mixed$_l$	mixed$_h$	
Individual RL (origin)	843	1552	1745	1867	1869	2100	769	1753	1845	-
Individual RL (transfer)	1198	2198	2493	2578	2330	2837	1046	2487	2513	38%
MetaLight (origin)	784	984	1854	261	482	2145	694	954	2083	-
MetaLight (transfer)	983	982	2287	316	593	2487	865	1139	2593	19%
PressLight (origin)	809	1173	1930	302	437	1846	639	834	1832	-
PressLight (transfer)	1119	1703	2693	429	569	2376	906	1287	2673	41%
CoLight (origin)	511	733	1217	159	305	1457	438	657	1367	-
CoLight (transfer)	-	-	-	-	-	-	-	-	-	-
GeneraLight (origin)	586	1257	1616	1209	1257	1686	792	916	1574	-
GeneraLight (transfer)	686	1571	2101	1330	1320	2057	935	1026	1747	17%
MetaVIM (origin)	**492**	**694**	**1189**	**149**	**288**	**1387**	**408**	**622**	**1272**	-
MetaVIM (transfer)	**513**	**729**	**1362**	**153**	**341**	**1477**	**443**	**682**	**1401**	**9%**

variables. That is, given a novel or unseen task, the task-specific information would be represented as latent variable rather than acting as distractors. Hence, the latent variable helps to learn the across-task shared policy function better.

5.5.3 Ablations

To better validate the contribution of each component, four variants of MetaVIM are evaluated on the common testing mode in the Shenzhen road network, including:

- **Base** only keeps the policy network and removes the mVAE and latent variable m.
- **Base+m** where the encoder and m are introduced. Base+m contains policy network and encoder, which keeps latent variable but discards intrinsic rewards.
- **Base+m+tran_RS** contains policy network, encoder and transition decoders but discards reward decoders. Transition decoders $\mathbf{p}^{\phi_0}(o_{t+1})$ and $\mathbf{p}^{\phi_{\tilde{o}}}(\tilde{o}_{t+1})$ are used and only $\|o_{t+1} - \tilde{o}_{j,t+1}\|$ is remained in Eq. 17 to form the intrinsic reward.
- **Base+m+rew_RS** contains policy network, encoder and reward decoders but discards transition decoders. Reward encoders $\mathbf{p}^{\phi_{\mathrm{r}}}(r_{t+1})$ and $\mathbf{p}^{\phi_{\tilde{\mathrm{r}}}}(\tilde{r}_{t+1})$ are used and only $\|r_{t+1} - \tilde{r}_{j,t+1}\|$ is remained in Eq. 17 to form the intrinsic reward.

Overall, MetaVIM contains the whole modules: policy network, encoder and decoders.

The qualitative evaluation results are listed in Tab. 4 and the learning curves are shown in Appendix A. We can obtain the following findings: 1) Among these 5 models, the performance of *Baseline* is the worst. The reason is that it is hard to learn the effective decentralized policy independently in the multi-agent traffic signal control task, where one agent's reward and transition are affected by its neighbors. 2) Compared with the baseline, the improvement of *Baseline* + m demonstrates the effectiveness of the latent variable m. The latent variable not only identifies the POMDP-specific information and helps to learn POMDP-shared policy network, but also trades off the exploration and exploitation during the RL procedure. 3) The *tran_RS* and *rew_RS* are both effective because each of them encourages the policy learning stable. Compared to them, the superiority of *MetaVIM* indicates *tran_RS* and *rew_RS* are complementary to each other. 4) Overall, all of the proposed components contribute positively to the final model.

6 Conclusions and Future Works

In this paper, we propose a novel Meta RL method MetaVIM for multi-intersection traffic signal control, which can make

the policy learned from a training scenario generalizable to new unseen scenarios. MetaVIM learns the decentralized policy for each intersection which considers neighbor information in a latent way. We conduct extensive experiments and demonstrate the superior performance of our method over the state-of-the-art. We have collected and released more complex scenarios containing different structures [7], and will improve the method based on these scenarios in the future. In addition, the utilization of latent variable in model-based RL for traffic signal control will also be explored to improve sample efficiency.

REFERENCES

[1] P. Koonce and L. Rodegerdts, "Traffic signal timing manual." United States. Federal Highway Administration, Tech. Rep. FHWA-HOP-08-024, 2008.

[2] P. Varaiya, "The max-pressure controller for arbitrary networks of signalized intersections," in *Advances in Dynamic Network Modeling in Complex Transportation Systems*. Springer, 2013.

[3] X. Guo, Z. Yu, P. Wang, Z. Jin, J. Huang, D. Cai, X. He, and X. Hua, "Urban traffic light control via active multi-agent communication and supply-demand modeling," *IEEE Transactions on Knowledge and Data Engineering*, pp. 1–11, 2021, early access.

[4] J. Ke, F. Xiao, H. Yang, and J. Ye, "Learning to delay in ride-sourcing systems: A multi-agent deep reinforcement learning framework," *IEEE Transactions on Knowledge and Data Engineering*, vol. 34, no. 5, pp. 2280–2292, 2022.

[5] Z. Pan, W. Zhang, Y. Liang, W. Zhang, Y. Yu, J. Zhang, and Y. Zheng, "Spatio-temporal meta learning for urban traffic prediction," *IEEE Transactions on Knowledge and Data Engineering*, vol. 34, no. 3, pp. 1462–1476, 2022.

[6] S. He and K. G. Shin, "Spatio-temporal capsule-based reinforcement learning for mobility-on-demand coordination," *IEEE Transactions on Knowledge and Data Engineering*, vol. 34, no. 3, pp. 1446–1461, 2022.

[7] Y. Tong, D. Shi, Y. Xu, W. Lv, Z. Qin, and X. Tang, "Combinatorial optimization meets reinforcement learning: Effective taxi order dispatching at large-scale," *IEEE Transactions on Knowledge and Data Engineering*, pp. 1–12, 2021, early access.

[8] S. Wang, J. Cao, and P. S. Yu, "Deep learning for spatio-temporal data mining: A survey," *IEEE Transactions on Knowledge and Data Engineering*, vol. 34, no. 8, pp. 3681–3700, 2022.

[9] J. Gu, Q. Zhou, J. Yang, Y. Liu, F. Zhuang, Y. Zhao, and H. Xiong, "Exploiting interpretable patterns for flow prediction in dockless bike sharing systems," *IEEE Transactions on Knowledge and Data Engineering*, vol. 34, no. 2, pp. 640–652, 2022.

[10] J. Liu, T. Li, S. Ji, P. Xie, S. Du, F. Teng, and J. Zhang, "Urban flow pattern mining based on multi-source heterogeneous data fusion and knowledge graph embedding," *IEEE Transactions on Knowledge and Data Engineering*, pp. 1–13, 2021, early access.

[11] F. Zhang, Y. Liu, N. Feng, C. Yang, J. Zhai, S. Zhang, B. He, J. Lin, X. Zhang, and X. Du, "Periodic weather-aware lstm with event mechanism for parking behavior prediction," *IEEE Transactions on Knowledge and Data Engineering*, vol. 34, no. 12, pp. 5896–5909, 2022.

[12] H. Wei, G. Zheng, H. Yao, and Z. Li, "Intellilight: A reinforcement learning approach for intelligent traffic light control," in *SIGKDD*, 2018.

[13] H. Wei, C. Chen, G. Zheng, K. Wu, V. Gayah, K. Xu, and Z. Li, "Presslight: Learning max pressure control to coordinate traffic signals in arterial network," in *SIGKDD*, 2019.

[14] X. Zang, H. Yao, G. Zheng, N. Xu, K. Xu, and Z. Li, "Metalight: Value-based meta-reinforcement learning for traffic signal control," in *AAAI*, 2020, pp. 1153–1160.

[15] L. Kuyer, S. Whiteson, B. Bakker, and N. Vlassis, "Multiagent reinforcement learning for urban traffic control using coordination graphs," in *ECML-PKDD*. Springer, 2008.

[16] E. Van der Pol and F. A. Oliehoek, "Coordinated deep reinforcement learners for traffic light control," in *NeurIPS 2016 Workshop on Learning, Inference and Control of Multi-Agent Systems*, 2016, pp. 1–8.

[17] D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell, "Curiosity-driven exploration by self-supervised prediction," in *CVPR Workshops*, 2017.

[18] M. Andrychowicz, F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, O. P. Abbeel, and W. Zaremba, "Hindsight experience replay," in *NeurIPS*, 2017, pp. 5048–5058.

[19] R. Chitnis, S. Tulsiani, S. Gupta, and A. Gupta, "Intrinsic motivation for encouraging synergistic behavior," in *ICLR*, 2019, pp. 1–15.

[20] H. Zhang, S. Feng, C. Liu, Y. Ding, Y. Zhu, Z. Zhou, W. Zhang, Y. Yu, H. Jin, and Z. Li, "Cityflow: A multi-agent reinforcement learning environment for large scale city traffic scenario," in *WWW*, 2019.

[21] F. Webster, "Traffic signal settings, road research technical paper no. 39," *Road Research Laboratory*, 1958.

[22] S. Chiu, "Adaptive traffic signal control using fuzzy logic," in *Proceedings of the Intelligent Vehicles Symposium*. IEEE, 1992.

[23] S.-B. Cools, C. Gershenson, and B. D'Hooghe, "Self-organizing traffic lights: A realistic simulation," in *Advances in applied self-organizing systems*. Springer, 2013.

[24] K. Gao, Y. Zhang, A. Sadollah, and R. Su, "Optimizing urban traffic light scheduling problem using harmony search with ensemble of local search," *Applied Soft Computing*, vol. 48, pp. 359–372, 2016.

[25] K. Gao, Y. Zhang, R. Su, F. Yang, P. N. Suganthan, and M. Zhou, "Solving traffic signal scheduling problems in heterogeneous traffic network by using meta-heuristics," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 9, pp. 3272–3282, 2018.

[26] S. A. Celtek, A. Durdu, and M. E. M. Alı, "Real-time traffic signal control with swarm optimization methods," *Measurement*, vol. 166, p. 108206, 2020.

[27] Y. Cheng, X. Hu, Q. Tang, H. Qi, and H. Yang, "Monte carlo tree search-based mixed traffic flow control algorithm for arterial intersections," *Transportation Research Record*, vol. 2674, no. 8, pp. 167–178, 2020.

[28] S. Chen and D. J. Sun, "An improved adaptive signal control method for isolated signalized intersection based on dynamic programming," *IEEE Intelligent Transportation Systems Magazine*, vol. 8, no. 4, pp. 4–14, 2016.

[29] Q. He, R. Kamineni, and Z. Zhang, "Traffic signal control with partial grade separation for oversaturated conditions," *Transportation research part C: emerging technologies*, vol. 71, pp. 267–283, 2016.

[30] R. Mohebifard and A. Hajbabaie, "Optimal network-level traffic signal control: A benders decomposition-based solution algorithm," *Transportation Research Part B: Methodological*, vol. 121, pp. 252–274, 2019.

[31] S. S. S. M. Qadri, M. A. Gökçe, and E. Öner, "State-of-art review of traffic signal control methods: challenges and opportunities," *European transport research review*, vol. 12, no. 1, pp. 1–23, 2020.

[32] 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 Transactions on Intelligent Transportation Systems*, vol. 14, no. 3, pp. 1140–1150, 2013.

[33] M. Abdoos, N. Mozayani, and A. L. Bazzan, "Holonic multi-agent system for traffic signals control," *Engineering Applications of Artificial Intelligence*, vol. 26, no. 5-6, pp. 1575–1587, 2013.

[34] I. Dusparic and V. Cahill, "Distributed w-learning: Multi-policy optimization in self-organizing systems," in *self-adaptive and self-organizing systems*. IEEE, 2009.

[35] M. Abdoos, N. Mozayani, and A. L. Bazzan, "Traffic light control in non-stationary environments based on multi agent q-learning," in *IEEE International Conference on Intelligent Transportation Systems*. IEEE, 2011.

[36] X. Huang, D. Wu, M. Jenkin, and B. Boulet, "Modellight: Model-based meta-reinforcement learning for traffic signal control," *arXiv preprint arXiv:2111.08067*, 2021.

[37] Q. Jiang, J. Li, W. S. SUN, and B. Zheng, "Dynamic lane traffic signal control with group attention and multi-timescale reinforcement learning." IJCAI, 2021.

[38] G. Zheng, X. Zang, N. Xu, H. Wei, Z. Yu, V. Gayah, K. Xu, and Z. Li, "Diagnosing reinforcement learning for traffic signal control," *arXiv*, 2019.

[39] G. Zheng, Y. Xiong, X. Zang, J. Feng, H. Wei, H. Zhang, Y. Li, K. Xu, and Z. Li, "Learning phase competition for traffic signal control," in *CIKM*, 2019.

[40] Y. Xiong, G. Zheng, K. Xu, and Z. Li, "Learning traffic signal control from demonstrations," in *CIKM*, 2019.

[41] C. Chen, H. Wei, N. Xu, G. Zheng, M. Yang, Y. Xiong, K. Xu, and Z. Li, "Toward a thousand lights: Decentralized deep reinforcement learning for large-scale traffic signal control." in *AAAI*, 2020, pp. 3414–3421.

7. https://github.com/zhuliwen/RoadnetSZ

[42] A. Oroojlooy, M. Nazari, D. Hajinezhad, and J. Silva, "Attendlight: Universal attention-based reinforcement learning model for traffic signal control," *arXiv preprint arXiv:2010.05772*, 2020.

[43] H. Zhang, M. Kafouros, and Y. Yu, "Planlight: Learning to optimize traffic signal control with planning and iterative policy improvement," *IEEE Access*, vol. 8, pp. 219 244–219 255, 2020.

[44] T. Chu, J. Wang, L. Codecà, and Z. Li, "Multi-agent deep reinforcement learning for large-scale traffic signal control," *ITS*, vol. 21, no. 3, pp. 1086–1095, 2019.

[45] T. Nishi, K. Otaki, K. Hayakawa, and T. Yoshimura, "Traffic signal control based on reinforcement learning with graph convolutional neural nets," in *IEEE International Conference on Intelligent Transportation Systems*. IEEE, 2018.

[46] H. Wei, N. Xu, H. Zhang, G. Zheng, X. Zang, C. Chen, W. Zhang, Y. Zhu, K. Xu, and Z. Li, "Colight: Learning network-level cooperation for traffic signal control," in *CIKM*, 2019.

[47] Z. Yu, S. Liang, L. Wei, Z. Jin, J. Huang, D. Cai, X. He, and X.-S. Hua, "Macar: Urban traffic light control via active multi-agent communication and action rectification." in *IJCAI*, 2020.

[48] B. Xu, Y. Wang, Z. Wang, H. Jia, and Z. Lu, "Hierarchically and cooperatively learning traffic signal control," in *AAAI*, 2021, pp. 669–677.

[49] H. Zhang, C. Liu, W. Zhang, G. Zheng, and Y. Yu, "Generalight: Improving environment generalization of traffic signal control via meta reinforcement learning," in *CIKM*, 2020.

[50] L. Zintgraf, K. Shiarlis, M. Igl, S. Schulze, Y. Gal, K. Hofmann, and S. Whiteson, "Varibad: A very good method for bayes-adaptive deep rl via meta-learning," in *ICLR*, 2019, pp. 1–20.

[51] H. Yang, D. Shi, C. Zhao, G. Xie, and S. Yang, "Ciexplore: Curiosity and influence-based exploration in multi-agent cooperative scenarios with sparse rewards," in *CIKM*, 2021.

[52] M. Bellemare, S. Srinivasan, G. Ostrovski, T. Schaul, D. Saxton, and R. Munos, "Unifying count-based exploration and intrinsic motivation," in *NeurIPS*, 2016, pp. 1471–1479.

[53] H. Tang, R. Houthooft, D. Foote, A. Stooke, X. Chen, Y. Duan, J. Schulman, F. De Turck, and P. Abbeel, "# exploration: A study of count-based exploration for deep reinforcement learning," in *NeurIPS*, 2017.

[54] G. Ostrovski, M. G. Bellemare, A. Oord, and R. Munos, "Count-based exploration with neural density models," in *ICML*, 2017, pp. 2721–2730.

[55] Y. Burda, H. Edwards, D. Pathak, A. Storkey, T. Darrell, and A. A. Efros, "Large-scale study of curiosity-driven learning," *arXiv preprint arXiv:1808.04355*, 2018.

[56] J. Achiam and S. Sastry, "Surprise-based intrinsic motivation for deep reinforcement learning," *arXiv preprint arXiv:1703.01732*, 2017.

[57] N. Haber, D. Mrowca, L. Fei-Fei, and D. L. Yamins, "Learning to play with intrinsically-motivated self-aware agents," *arXiv preprint arXiv:1802.07442*, 2018.

[58] N. Jaques, A. Lazaridou, E. Hughes, C. Gulcehre, P. Ortega, D. Strouse, J. Z. Leibo, and N. De Freitas, "Social influence as intrinsic motivation for multi-agent deep reinforcement learning," in *ICML*, 2019, pp. 3040–3049.

[59] L. M. Zintgraf, L. Feng, C. Lu, M. Igl, K. Hartikainen, K. Hofmann, and S. Whiteson, "Exploration in approximate hyper-state space for meta reinforcement learning," in *ICML*, 2021, pp. 12 991–13 001.

[60] R. Dorfman, I. Shenfeld, and A. Tamar, "Offline meta learning of exploration," *arXiv preprint arXiv:2008.02598*, 2020.

[61] K. Rakelly, A. Zhou, C. Finn, S. Levine, and D. Quillen, "Efficient off-policy meta-reinforcement learning via probabilistic context variables," in *ICML*, 2019, pp. 5331–5340.

[62] P.-A. Kamienny, M. Pirotta, A. Lazaric, T. Lavril, N. Usunier, and L. Denoyer, "Learning adaptive exploration strategies in dynamic environments through informed policy regularization," *arXiv preprint arXiv:2005.02934*, 2020.

[63] J. Zhang, J. Wang, H. Hu, T. Chen, Y. Chen, C. Fan, and C. Zhang, "Metacure: Meta reinforcement learning with empowerment-driven exploration," in *ICML*, 2021, pp. 12 600–12 610.

[64] A. Gupta, R. Mendonca, Y. Liu, P. Abbeel, and S. Levine, "Meta-reinforcement learning of structured exploration strategies," in *NeurIPS*, 2018, pp. 5302–5311.

[65] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv*, 2013.

[66] Y. Duan, J. Schulman, X. Chen, P. L. Bartlett, I. Sutskever, and P. Abbeel, "Rl ^{2}: Fast reinforcement learning via slow reinforcement learning," *arXiv preprint arXiv:1611.02779*, 2016.

[67] F. A. Oliehoek, C. Amato *et al.*, *A concise introduction to decentralized POMDPs*. Springer, 2016.

[68] X. Zhou, W. Wang, T. Wang, Y. Lei, and F. Zhong, "Bayesian reinforcement learning for multi-robot decentralized patrolling in uncertain environments," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 12, pp. 11 691–11 703, 2019.

[69] P. Poupart, N. Vlassis, J. Hoey, and K. Regan, "An analytic solution to discrete bayesian reinforcement learning," in *ICML*, 2006, pp. 697–704.

[70] M. Ghavamzadeh, S. Mannor, J. Pineau, A. Tamar *et al.*, "Bayesian reinforcement learning: A survey," *Foundations and Trends® in Machine Learning*, vol. 8, no. 5-6, pp. 359–483, 2015.

[71] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv*, 2017.

[72] R. P. Roess, E. S. Prassas, and W. R. McShane, *Traffic engineering*. Pearson/Prentice Hall, 2004.

[73] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *ICML*, 2017, pp. 1126–1135.

[74] C. Cai, C. K. Wong, and B. G. Heydecker, "Adaptive traffic signal control using approximate dynamic programming," *Transportation Research Part C: Emerging Technologies*, vol. 17, no. 5, pp. 456–474, 2009.

[75] P. Mannion, J. Duggan, and E. Howley, "An experimental review of reinforcement learning algorithms for adaptive traffic signal control," in *Autonomic road transport support systems*. Springer, 2016.

[76] S. G. Rizzo, G. Vantini, and S. Chawla, "Time critic policy gradient methods for traffic signal control in complex and congested scenarios," in *SIGKDD*, 2019.

[77] I. Arel, C. Liu, T. Urbanik, and A. G. Kohls, "Reinforcement learning-based multi-agent system for network traffic signal control," *IET Intelligent Transport Systems*, vol. 4, no. 2, pp. 128–135, 2010.

Liwen Zhu received the B.S. degree from Beijing University of Technology, China, in 2019. She is currently pursuing the M.S. degree with the School of Electronic and Computer Engineering, Peking University, China. Her current research interests include reinforcement learning and intelligent transportation system.

Peixi Peng received the PhD degree from Peking University, in 2017, Beijing, China. He is currently an associate researcher with the School of Computer Science, Peking University, Beijing, China, and is also the assistant researcher of Peng Cheng Laboratory, Shenzhen, China. He is the author or co-author of more than 30 technical articles in refereed journals and top conferences. His research interests include computer vision and reinforcement learning.

Zongqing Lu is a Boya Assistant Professor in the School of Computer Science, also affiliated with the Institute for Artificial Intelligence, Peking University. He received the B.S. and M.S. degrees from Southeast University, China, and the Ph.D. degree from Nanyang Technological University, Singapore, 2014. His resarch interests include (multi-agent) reinforcement learning and intelligent distributed systems.

Yonghong Tian is currently a Boya Distinguished Professor with the School of Computer Science, Peking University, China, and is also the deputy director of Artificial Intelligence Research Center, PengCheng Laboratory, Shenzhen, China. His research interests include neuromorphic vision, brain-inspired computation and multimedia big data. He is the author or coauthor of over 200 technical articles in refereed journals. He was the recipient of the Chinese National Science Foundation for Distinguished Young Scholars in 2018. He is a fellow of IEEE, a senior member of CIE and CCF, a member of ACM.

APPENDIX A
LEARNING CURVES

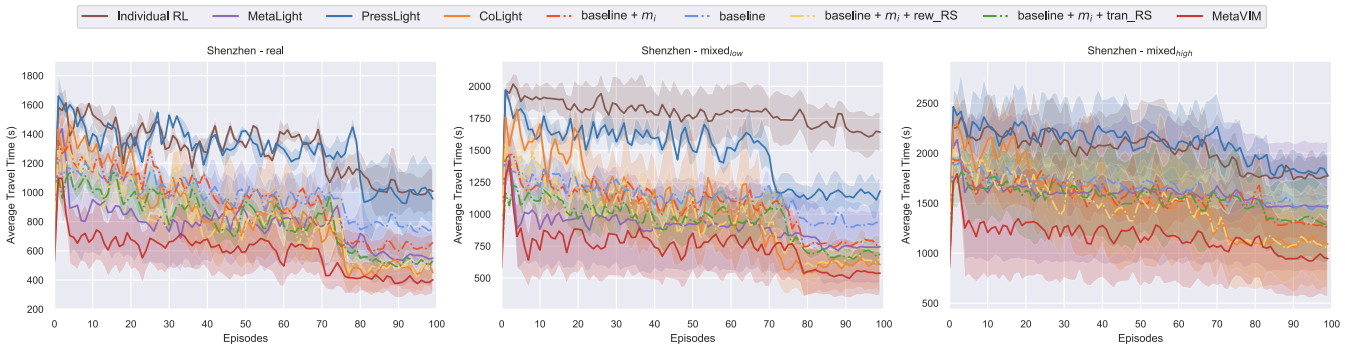


Fig. 9: Learning curves of MetaVIM and baselines in the Shenzhen road network map. The first, second and third columns correspond to the *real*, $mixed_l$ and $mixed_h$ configurations respectively. The final model MetaVIM outperforms all other models and shows great stability. In addition, the learning curves of ablations indicate that each component contributes positively.

APPENDIX B
IMPLEMENTATION DETAILS OF METAVIM

TABLE 6: Implementation details of MetaVIM

Items	Details
Number of policy steps	3600
Discount factor γ	0.95
Policy minibatch	16
mVAE minibatch	25
Value loss coefficient	0.5
Entropy coefficient	0.01
ELBO loss coefficient	1.0
Latent space dimensionality	5
Aggregator hidden size	64
Policy network architecture	2 hidden layers, 32 nodes each, Tanh activations
Policy network optimizer	Adam with learning rate 0.0007 and epsilon 1e-5
Encoder architecture	FC layer with 40 nodes, GRU with hidden size 64, output layer with 10 outputs (μ and σ), ReLU activations
Transition Decoder architecture	2 hidden layers, 32 nodes each, 25 outputs heads, ReLU activations
Reward Decoder architecture	2 hidden layers, 32 nodes each, 25 outputs heads, ReLU activations
mVAE optimizer	Adam with learning rate 0.001 and epsilon 1e-5

APPENDIX C
EXPERIMENTS ON DIFFERENT LANE CONFIGURATIONS

To validate that the method is robust to diverse lane configurations, we modify the 4x4 Hangzhou road network using different lane configurations: *left-2 / straight-4 /right-1* indicates the roads in the network is formed by 7 lanes, where the

TABLE 7: Experimental performance on intersections with different topologies

Model	**l-2 / s-4 / r-1**	**l-1 / s-2 / r-1**
Random	828.00	866.33
MaxPressure	829.06	867.16
Fixedtime	1591.83	1529.09
FixedtimeOffset	830.85	867.97
SOTL	1258.67	1285.66
Individual RL	835.64	860.37
MetaLight	775.65	790.00
PressLight	798.87	819.36
CoLight	501.37	516.55
MetaVIM	**485.36**	**490.12**

number of left-turn lanes, straight lanes and right-turn lanes are 1, 2, and 1 respectively. Similarly, *left-1 / straight-2 /right-1* indicates the roads in the network is formed by 4 lanes, where the number of left-turn lanes, straight lanes and right-turn lanes are 1, 1, and 1 respectively.

The results are shown in Tab. 7, and it is evident that: Among these baselines, the performance of Fixedtime is the worst because it can not adapt to the dynamics in the road network. RL-based methods show advantages than conventional traffic method. Among the RL-based method, MetaVIM outperforms Individual RL, MetaLight, PressLight and CoLight in both *left-2 / straight-4 /right-1* and *left-1 / straight-2 /right-1* scenarios. It indicates the method can handle the intersections with different lane configurations well.

APPENDIX D
EXPERIMENTS ON SUMO

TABLE 8: Verification on the SUMO simulation platform

Metrics	MA2C	IA2C	IQL-LR	IQL-DNN	MetaVIM
reward	-366.13	-1062.23	-1057.19	-2619.42	-298.58
avg. queue length [veh]	1.88	3.74	2.78	4.48	3.97
avg. intersection delay [s/veh]	11.98	52.52	74.56	166.03	10.56
avg. vehicle speed [m/s]	2.40	1.60	3.18	1.48	2.86
trip completion flow [veh/s]	0.63	0.38	0.73	0.23	0.79
trip delay [s]	322.81	560.61	186.67	477.43	157.23

To validate that the method is robust to different traffic simulation platforms, several experiments are conducted on SUMO[8]. For fair comparisons, 4 recent methods which provides the source code on SUMO are evaluated at the same setting, including:

• MA2C [44]: a fully scalable and decentralized MARL alogrithm for the deep RL agent: advantage actor critic (A2C), within the context of adaptive traffic signal control (ATSC).

• IA2C [44]: an independent advantage actor critic (A2C) method in multi-agent scenario.

• IQL-LR [74]: a control algorithm based on dynamic programming, using an approximation ot the value function of the dynamic programming and reinforcement learning to update the approximation.

• IQL-DNN [44]: the same as IQL-LR but uses DNN for fitting the Q-function.

We choose a 5x5 large traffic grid as the experimental scenario, which is the same as [44]. The scenario is formed by two-lane arterial streets with speed limit 20m/s and one-lane avenues with speed limit 11m/s. The evaluation metrics are provided by SUMO as follows:

• *Reward* The rewards received by all agents across the entire simulation time.

• *Avg. Queue Length* Average queue length is the average numbers of vehicles in the queue over all intersections. If a vehicle is waiting at an intersection and the speed less than 0.1m/s, we think the car is on the queue.

• *Avg. intersection delay* If a vehicle is waiting at an intersection and the speed less than 0.1m/s, the extra time the vehicle stays at the intersection is the delay time. We calculate the average time over all vehicles, then we get the avg. intersection delay.

8. http://sumo.dlr.de/index.html

• *Avg. vehicle speed* The average speed of all vehicles in the road network for the entire simulation time.

• *Trip completion flow* Trip completion flow is the number of vehicles that complete the trip. Trip completion flow is calculated by dividing the total number of vehicles that complete the trip during the entire simulation time by the horizon.

• *Trip delay* Trip delay is the extra time wasted for all vehicles that complete the trip in the road network.

Tab. 8 lists the comparative results on the common testing mode over SUMO, and it is evident that: In general, MetaVIM performs better than advantage actor critic and Q learning in both decentralized MARL and independent control, and it indicates the advantage of the MetaVIM in various evaluation criterias. MetaVIM achieves higher reward in evaluation. Moreover, MetaVIM is superior to other methods clearly in average queue length, average intersection delay, average vehicle speed, trip completion flow and trip delay, which demonstrates the effectiveness of the method in SUMO platform.

APPENDIX E
SCALABILITY VALIDATION

To validate scalability of the method, an additional experiment is conducted on a public dataset of 196 intersections in New York City [9], as illustrated in Figure 10. It is one of the largest public dataset in Cityflow to our knowledge. As shown in Tab. 9, MetaVIM achieves better results than compared methods, which demonstrates the superior scalability of MetaVIM. The reason is that MetaVIM is a decentralized method and doesn't need the joint action of all agents and full state. Hence, MetaVIM could avoid the dimensional explosion of large scale of agents. In addition, as the number of agents increases, more sample data will be collected. It leads to a significant increase in the number of training samples, rather than an increase in dimensionality. Therefore, MetaVIM could scale well.

Fig. 10: Simulation on a large-scale dataset with 196 intersections

TABLE 9: Experimental performance on 196 intersections in New York

Model	**real**	**mixed**$_l$	**mixed**$_h$	mean
Random	1869.00	1748.76	1892.73	1836.83
MaxPressure	845.72	589.24	920.31	785.09
Fixedtime	1831.65	1742.71	1205.97	1593.44
FixedtimeOffset	1702.82	1732.37	1869.66	1768.28
SlidingFormula	1185.64	921.28	1299.19	1135.37
SOTL	1862.34	1793.37	1939.53	1865.08
Individual RL	1877.46	1369.64	1906.43	1717.84
MetaLight	1285.74	1005.36	1368.96	1220.02
PressLight	1586.75	1547.74	1685.74	1606.74
CoLight	637.79	479.31	761.10	626.07
GeneraLight	710.23	485.84	862.47	686.18
MetaVIM	**586.36**	**396.57**	**748.65**	**577.19**

9. https://github.com/wingsweihua/colight/blob/master/data/NewYork/28_7/roadnet_28_7.json

APPENDIX F
RL-BASED TRAFFIC SIGNAL CONTROL SURVEY

A survey on RL-based traffic signal control methods is shown in Tab. 10.

TABLE 10: RL-based Methods Summary

Classes	Methods	Descriptions
tabular Q-learning	MARLIN-ATSC [32]	real-time adjustment of signal timing plans based on traffic fluctuations, with each agent coordinating with adjacent intersections to generate actions
	holonic Q-learning [33]	the traffic network is divided into multiple zones with two phases of control
	DWL [34]	collaboration-based agent self-optimizes for multiple strategies
	Q-learning based [35]	apply traditional Q-learning method to multi-agent systems
single agent	Intellilight [12]	use a variety of different combinations of traffic indicators as the states
	RLTSC [75]	use current phase index, phase duration, and queue length as states
	AttendLight [42]	a single general model is designed, and two attention models are used to deal with multiple topologies
	MT-GAD [37]	use a group attention structure to reduce the number of required parameters and to achieve a better generalizability
	PG Time EWMA [76]	a policy gradient method based on episodic conditions and a time-dependent baseline to learn an optimal policy for traffic signal control in congested conditions
isolated multi-agent	LIT [38]	analyze which traffic indicators are suitable as states or reward
	FRAP [39]	introduce phase competition to select a more suitable phase
	DemoLight [40]	demonstrations can be collected from classical methods to accelerate learning
	PressLight [13]	combine the traditional traffic method MaxPressure with RL technology together
	MPLight [41]	experiment on a large dataset using a shared strategy
	PlanLight [43]	learn from the demonstration of the rollouts
centralized	DQN-TP [16]	combination of DQN method and transfer algorithm to overcome instability problem
	max-plus [15]	coordination by max-plus algorithm to obtain optimal joint actions
decentralized	LQF [77]	a distributed static control approach that integrates adaptive RL systems and LQF
	NFQI with GCNN [45]	the traffic characteristics of long-distance roads are considered by using graph neural network
	CoLight [46]	use graph convolution and attention mechanism to model the neighbor information
	HiLight [48]	use hierarchical reinforcement learning to allow agents to optimize different short-term sub-goals
	MA2C [44]	use neighbor information to improve observability and reduce the learning difficulty of local agents
meta-learning	MetaLight [14]	a value-based meta reinforcement learning method via parameter initialization
	GeneraLight [49]	generate diversified traffic flows using GAN, and then improve generalization ability through clustering
	ModelLight [36]	a model-based meta-reinforcement learning framework, meta-learning method can improve data efficiency and reduce the number of interactions required with real-world environments