Stochastic Graph Neural Network-Based Value Decomposition for MARL in Internet of Vehicles

Baidi Xiao, *Graduate Student Member, IEEE*, Rongpeng Li, *Member, IEEE*, Fei Wang, Chenghui Peng, Jianjun Wu, Zhifeng Zhao, *Member, IEEE*, and Honggang Zhang, *Senior Member, IEEE*

***Abstract*—Autonomous driving has witnessed incredible advances in the past several decades, while Multi-Agent Reinforcement Learning (MARL) promises to satisfy the essential need of autonomous vehicle control in a wireless connected vehicle networks. In MARL, how to effectively decompose a global feedback into the relative contributions of individual agents belongs to one of the most fundamental problems. However, the environment volatility due to vehicle movement and wireless disturbance could significantly shape time-varying topological relationships among agents, thus making the Value Decomposition (VD) challenging. Therefore, in order to cope with this annoying volatility, it becomes imperative to design a dynamic VD framework. Hence, in this article, we propose a novel Stochastic Value Mixing (SVMIX) methodology by taking account of dynamic topological features during VD and incorporating the corresponding components into a multi-agent actor-critic architecture. In particular, Stochastic Graph Neural Network (SGNN) is leveraged to effectively capture underlying dynamics in topological features and improve the flexibility of VD against the environment volatility. Finally, the superiority of SVMIX is verified through extensive simulations.**

***Index Terms*—Autonomous vehicle control, multi-agent reinforcement learning, value decomposition, stochastic graph neural network.**

I. Introduction

In RECENT years, the unprecedented development of artificial intelligence (AI) brings self-driving vehicles [2], [3], [4] into the spotlight, and these intelligent vehicles form an Internet of Vehicles (IoV). Equipped with the capability for environmental perception [5], the vehicles try to respond to the observation of the surroundings and find an optimal trajectory between two sites [6], [7] with calibrated self-decision control and traffic congestion avoidance [8], [9]. Additionally, there emerges some research interest towards traffic signal control [10] or fleet control [11] in IoV as well. Nevertheless, these intricate cases (e.g., traffic control) require distributively deployed intelligent vehicles to collaborate on top of reliable communication, which further constitutes a Multi-Agent System (MAS) and catalyses the research progress in a myriad of related studies (e.g., traffic control and prediction) [12], [13]. In particular, Multi-Agent Reinforcement Learning (MARL), which incorporates Deep Reinforcement Learning (DRL) into MAS, emerges by astutely learning through the trial-and-error interaction between multiple agents (i.e., vehicles) and the complicated IoV environment, and promises to yield smart policies for the formulated Markov Decision Process (MDP).

Traditionally, in Independent Q-Learning (`IQL`) [14], one typical kind of MARL methods, each individual agent learns its policy by regarding other agents as parts of the environment, and often experiences a non-stationary environment, as the agents always update their policies independently during the learning. Consequently, the non-stationary issue impedes the optimization of agents' policies. Instead, mutual communication shall be considered for cooperative agents to reach holistic consistency. On the other hand, it is natural for MAS to only observe a global reward from the environment, from which it is usually infeasible to accurately infer individual rewards for agents due to their complicated relationship. Thus it becomes essential to tackle with the agent heterogeneity and credit assignment issue [15].

In recent years, algorithms with Centralized Training and Decentralized Execution (CTDE) have become the centerpiece of MARL as they can somewhat handle the aforementioned two problems (i.e., non-stationary learning, and credit assignment). As its name implies, CTDE can be divided into the training phase and execution phase. In particular, in the training phase, agents can implicitly observe the global information of the environment in a centralized manner, so as to guarantee the communication among agents and tackle the non-stationary problem. Subsequently, in the decentralized execution phase, each agent capably makes decisions based on its local observation only. Typical examples of CTDE include `MADDPG`, `COMA`, etc [15], [16], [17], [18]. Nevertheless, as the number of agents increases, the centralized action-value function in CTDE algorithms such as `COMA` suffers from an exponential increase of the action space as well as the awful growth of computational complexity. Therefore, the Value Decomposition

Manuscript received 21 March 2023; revised 14 August 2023; accepted 26 August 2023. Date of publication 6 September 2023; date of current version 13 February 2024. Part of the paper has been accepted by IEEE VTC 2023-Spring [DOI: 10.1109/VTC2023-Spring57618.2023.10199985]. This work was supported in part by the National Natural Science Foundation of China under Grant 62071425, in part by the Zhejiang Key Research and Development Plan under Grant 2022C01093, in part by Huawei Cooperation Project, and in part by the Zhejiang Provincial Natural Science Foundation of China under Grant LR23F010005. The review of this article was coordinated by Dr. Cailian Chen. *(Corresponding author: Rongpeng Li.)*

Baidi Xiao and Rongpeng Li are with the College of Information Science and Electronic Engineering, Zhejiang University, Hangzhou 310027, China (e-mail: xiaobaidi@zju.edu.cn; lirongpeng@zju.edu.cn).

Fei Wang, Chenghui Peng, and Jianjun Wu are with the Huawei Technologies, Shanghai 201206, China (e-mail: wangfei76@huawei.com; pengchenghui@huawei.com; wujianjun@huawei.com).

Zhifeng Zhao and Honggang Zhang are with the Zhejiang Lab, Zhejiang University, Hangzhou 310027, China (e-mail: zhaozf@zhejianglab.com; honggangzhang@zhejianglab.com).

Digital Object Identifier 10.1109/TVT.2023.3312574

0018-9545 © 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: Zhejiang University. Downloaded on February 14,2024 at 05:44:41 UTC from IEEE Xplore. Restrictions apply.

(VD)-based algorithms [17], [18], [19], [20] are proposed to decompose a centralized value function into individual ones, thus decreasing the computational complexity in CTDE. Regretfully, these methods mostly concentrate on learning from agents with the invariant communication topology [20], [21], but flounder under dynamic environments entailing uncertainty and perturbation due to fluctuating connections and dynamically changing topologies. Meanwhile, most of VD-based MARL methods focus on the decomposition of action-value function, and the policies of individual agents could easily get stuck at sub-optimal solutions owing to the environmental complexity and the awful number of state-action pairs.

On the other hand, the advent of Graph Neural Network (GNN) [22] makes it possible to capture the topological features of vehicle-formed graphs. For example, the spectral-based GNNs [23] can filter the noise and extract features of signals in the transformed frequency domain. However, most of GNNs concentrate on the deterministic graphs during the training phase and ignore the time-varying underlying topologies with internal perturbations in some scenarios (e.g., the possibly disconnected communication links due to the vehicle mobility and limited communication ranges in traffic control) [20]. Therefore, it's reasonable to re-design the VD-based method by taking account of the graph stochasticity during the training phase [24], so as to capture dynamic topological features and enhance the robustness of learned policies.

In this article, we focus on dealing with the self-decision control in IoV for intelligent traffic control. Inspired by the Value Mixing (VMIX) module [17], we propose the brand-new Stochastic VMIX (SVMIX) MARL algorithm. Compared to previous works in [17], [19] and [20], SVMIX adopts Stochastic Graph Neural Network (SGNN) [25] to capture the dynamic topological features of the time-varying graphs and further decompose the state-value function. Besides, in order to tackle with the continuous action control, SVMIX successfully incorporates the VMIX module into the Proximal Policy Optimization (PPO)-based CTDE architecture [26]. Notably, the performance after introducing SGNN to PPO for VD still heavily relies on the selection of a suitable decomposition target from several candidate options (i.e., action-value function, reward function, and state-value function). In most existing works [17], [18], [19], [20], it commonly relies on the decomposition of the action-value function (i.e., Q-value function) such as QMIX [17]. Nevertheless, given the awful size of action space, it might be difficult to accurately approximate Q-value function for all state-action pairs, thus incurring misleading Q values and leading to sub-optimal individual actions. On the other hand, directly decomposing a global reward into individual ones faces severe learning instabilities. Therefore, in contrast to these adopted approaches for reward decomposition or Q-value decomposition, we resort to a rather distinctive means by decomposing the state-value function, so as to mitigate the approximation difficulty and avoid a local optimum. The contribution of this article can be summarized as follows.

- We incorporate SGNN in the VD-based MARL method, and propose a novel MARL method (i.e., SVMIX), which first applies state-value VD on top of PPO [26] (a kind of RL algorithm within the actor-critic framework). By introducing SGNN, SVMIX can intentionally aggregate the information of agents through different randomly sampled graphs to imitate the practically dynamic connectivity of vehicles and therefore endow vehicles with the capability to resist the environmental disturbance.
- We theoretically analyze the role that SGNN plays in the MARL method and manifest the importance and effectiveness that captured dynamical topological features contribute to a feasible solution for estimating individual value functions of each agent.
- We verify the performance of SVMIX on extensive simulation settings. Compared with several benchmark MARL methods (i.e., QMIX [17], MGAN [20] and FIRL [27]), SVMIX demonstrates its superiority and stability in balancing the convergence rate and utility. More significantly, unlike most of the CTDE algorithms which employ the information of all agents during the training, the adoption of SGNN in SVMIX allows to gather and utilize the information from part of agents to maintain competitive performance. Hence, SVMIX contributes to reducing communication overheads as well.

The reminder of the article is organized as follows. Section II briefly introduces the related works. Section III presents the necessary background and formulates the system model. Section IV describes the implementation details of the proposed algorithm and presents theoretical analysis. In Section V, we introduce the experimental settings and discuss the related simulation results. Section VI concludes the article and discusses future research directions.

II. Related Works

The traditional physics-based models [28], [29] for autonomous vehicle control are data-efficient as there are only a few parameters to calibrate with mathematically tractability [30]. However, these models lie on strong assumptions on traffic scenarios and cannot quickly adapt to the protean environment in practice. Instead, AI has been widely adopted in the field of IoV, and DRL belongs to one of the most fundamental tools [31], taking the observation (e.g., the extracted features from the perception components and the information of nearby vehicles) as input and making self-decision control (e.g., on the acceleration and direction) the same as the classic controller. Besides, deep learning methods like Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) have also made great progress in vision-based detection and prediction-based control [32]. Meanwhile, through learning from the driving data of human beings, imitation learning is able to generate a policy similar to the style of human driving as well [33].

Furthermore, the MARL method stands out for MAS in IoV where the communication & collaboration of multiple agents is essential. Notably, a direct application such as IQL faces non-stationary IoV environment while a conventional MARL method encounters credit assignment issues [14], [15]. Instead, CTDE has become one of the most popular paradigms. For

Authorized licensed use limited to: Zhejiang University. Downloaded on February 14,2024 at 05:44:41 UTC from IEEE Xplore. Restrictions apply.

TABLE I
MAIN NOTATIONS USED IN THIS PAPER

Notation	Definition
M_t	Number of vehicles at time-step t
$N \leq M_t$	Number of DRL-driven vehicles
$s \in \mathcal{S}$	Global state
$a^{(i)} \in \mathcal{A}$	Individual action of agent i
$\boldsymbol{a} \in \boldsymbol{\mathcal{A}}$	Joint action of all agents
$o^{(i)} \in \Omega$	Individual observation of agent i
$\boldsymbol{o}$	Joint observation of all agents
$\mathcal{O}$	Observation function
$\mathcal{P}$	Transition function
r	Reward
$r^{(i)}$	Implicit individual reward approximated by agent i
$\hat{r}^{(i)}$	Implicit individual reward approximated through `SGNN`
U	Utility function
$\mathcal{R}$	Reward function
Q	Action-value function
V	State-value function
$V^{(i)}$	Individual state-value function of agent i
$\boldsymbol{V}(\boldsymbol{o})$	Concatenated individual state-values of all agents
$\boldsymbol{V}'_f(\boldsymbol{o})$	Filtered individual state-values from filter f
$\boldsymbol{V}_{\text{agg}}(\boldsymbol{o})$	Aggregated individual state-values through `SGNN`
$V_{\text{agg}}^{(i)}(\boldsymbol{o})$	Aggregated individual state-value of agent i in $\boldsymbol{V}_{\text{agg}}(\boldsymbol{o})$
V_{tot}	Total state-value function
$\pi^{(i)}$	Individual policy of agent i
π	Joint policy
J	Discounted accumulated return
γ	Discount factor
$\mathbb{E}$	Expectation according to RES given $\boldsymbol{V}(\boldsymbol{o}_t)$
$\mathbb{E}_{\mathcal{P}}$	Expectation according to the transition function
$\mathbb{E}_t$	Expectation in a trajectory
v and v_{d}	Velocity and desired velocity of a vehicle
D	Penalty term in the reward function
α	Weight of the penalty term
$\mathcal{G}_t$	Graph consisted of vehicles at t
$\mathcal{G}^+$	Specific graph that encompasses all possible $\mathcal{G}_t$
$\mathbf{S}$	Shift matrix
$\mathbf{S}_k$	Shift matrix of the sampled sub-graph at k-order
$h_{fk} \in \mathcal{H}$	Coefficient at k-order in the f-th filter
$\mathbf{H}$	Concatenated stochastic graph filter coefficients
$\mathbf{u}$	Output signals of a stochastic graph filter
F	Number of parallel stochastic graph filters
K	Order of a stochastic graph filter
p	Bernoulli sampling probability

example, within the actor-critic framework, [15] and [16] utilize a centralized critic for centralized training as well as multiple actors for decentralized execution. Besides, [17], [18], [19], [20] apply VD and decompose the joint value function into individual ones corresponding to each agent. These methods somewhat deal with the credit assignment with reduced computational costs. In addition, some researchers focus on the communication between agents. Foerster et al. [34] assume the availability of the communication of all the agents while Jiang and Lu [35] rely on the local communication for agents within a certain proximity. Mao et al. [21] further limit the communication to allow each agent to observe the information of its neighbours only. To tackle the exponentially growing action space, Yang et al. [36] apply the Mean Field Theory (MFT) to treat the other agents as an ensemble for every agent, thus significantly reducing the computation cost and complexity.

Towards MARL-enabled autonomous vehicle control, Yang et al. [13] assign an individual modified `PPO` agent as the DRL controller for each vehicle and traffic light, so as to allow traffic lights to capably collaborate with autonomous vehicles. Specially, agents in [13] observe the local information independently. Bhalla et al. [37] improve the algorithm in [34] through the more effective communication among agents with message approximation and verify the superiority of the improved algorithm in a highway autonomous driving scenario. Palanisamy [38] assumes that each agent can take the other agents' policies into account, and adopts a Markov game-based DRL. Benefiting from the information exchange among vehicles, Xu et al. [27] incorporate a consensus algorithm with federated learning within the CTDE structure. Similarly, Chen et al. [39] leverage GNN to aggregate the node feature of each vehicle based on the communication graph in IoV and learn a joint policy via a centralized DRL agent. In order to eliminate collisions and optimize overall traffic flow at intersections, Guillen-Perez and Cano [40] implement a kind of joint decision making model for vehicles, by incorporating RNN into DRL and allowing the communication of agents for decision making. Guo et al. [41] improve `QMIX` for automated vehicles control at intersections and obtain superior performance than `QMIX` in terms of the higher speed and less probability of collisions. Nevertheless, the aforementioned researches seldom shed light on the dynamic topological connection of the vehicles.

III. Preliminaries and System Model

In this section, we briefly introduce the related knowledge of MDP and VD-based methods, and present the formulated system model to apply MARL for the autonomous vehicle control.

A. Preliminaries

We have listed the important notations used in this article in Table I.

Generally, an MARL task is formulated as a Decentralized Partially Observable Markov Decision Process (Dec-POMDP) [42], which is defined by the tuple $\langle \mathcal{I}, \mathcal{S}, \mathcal{A}, \mathcal{P}, \Omega, \mathcal{O}, \mathcal{R}, \gamma \rangle$. Here we assume that all agents are homogeneous. $\mathcal{I}$ represents the set of N agents, $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space for a single agent and $\boldsymbol{\mathcal{A}} := \mathcal{A}^N$ is the joint action space. The joint action $\boldsymbol{a} = \{a^{(1)}, a^{(2)}, \ldots, a^{(N)}\}$ taken at the current state s results in the next state s' through a transition of environment according to the transition function $\mathcal{P}(s'|s, \boldsymbol{a}) : \mathcal{S} \times \boldsymbol{\mathcal{A}} \times \mathcal{S} \rightarrow [0, 1]$. Owing to the scant ability of perception against the colossal environment, agent i gets a local observation $o^{(i)} \in \Omega$ via the observation function $\mathcal{O}(o^{(i)}|s, i) : \mathcal{S} \times \mathcal{I} \times \Omega \rightarrow [0, 1]$ instead of s at each time-step. All agents share a global reward function $\mathcal{R}(s, \boldsymbol{a}) : \mathcal{S} \times \boldsymbol{\mathcal{A}} \rightarrow \mathbb{R}$ and γ denotes the discount factor.

Furthermore, in a Dec-POMDP, agent i adopts an action according to its policy $\pi^{(i)}(\cdot|o^{(i)}) : \Omega \times \mathcal{A} \rightarrow [0, 1]$, which denotes the probability distribution of taking action $a^{(i)} \in \mathcal{A}$ conditioned on $o^{(i)}$. Likewise, $\boldsymbol{a}$ is adopted according to the probability $\pi(\boldsymbol{a}|\boldsymbol{o}) = \prod_{i=1}^{N} \pi^{(i)}(a^{(i)}|o^{(i)})$ where π denotes the joint policy and $\boldsymbol{o} = \{o^{(1)}, \ldots, o^{(N)}\}$ implies the joint observation which can be generated from s_t. In order to coordinate all agents for maximizing the discounted accumulated return $J = \sum_{t=0}^{\infty} \gamma^t \mathcal{R}(s_t, \boldsymbol{a}_t)$, a joint action-value function is defined

Authorized licensed use limited to: Zhejiang University. Downloaded on February 14,2024 at 05:44:41 UTC from IEEE Xplore. Restrictions apply.

as the expected discounted accumulated return starting from s_t and $\boldsymbol{a}_t$ at time-step t. That is

$$Q^{\pi}(s_t, \boldsymbol{a}_t) = \mathbb{E}_{\mathcal{P}}\left[\sum_{k=t}^{\infty}\gamma^{k-t}\mathcal{R}\left(s_k, \boldsymbol{a}_k\right)|s_t, \boldsymbol{a}_t, \pi\right], \quad (1)$$

where $\mathbb{E}_{\mathcal{P}}(\cdot)$ means the expectation of all possible discounted accumulated returns according to $\mathcal{P}$ given the initial s_t and $\boldsymbol{a}_t$. It can also be observed that the function has an exponential computational complexity due to the joint action $\boldsymbol{a}_t$.

To deal with this scalability issue while pursuing the optimal joint policy $\pi^* = \arg\max_\pi Q^\pi(s_t, \boldsymbol{a}_t)$, instead of adopting a joint action-value function as in (1), VD-based methods [17], [18], [19], [20] adhibit an individual action-value function $Q^{\pi^{(i)}}(o_t^{(i)}, a_t^{(i)}) = \mathbb{E}_{\mathcal{P}}\big[\sum_{k=t}^{\infty}\gamma^{k-t}r_k^{(i)}|o_t^{(i)}, a_t^{(i)}, \pi^{(i)}\big]$ for agent i, where $r_k^{(i)}$ is the implicitly approximated individual reward of agent i with respect to $\mathcal{R}(s_k, \boldsymbol{a}_k)$. Moreover, the individual action-value function reflects different contributions among agents. As discussed earlier, the decomposition of Q-value function for all state-action pairs possibly incurs misleading Q values and leads to sub-optimal individual actions.

Recalling that the expected J from s_t can also be represented by the joint state-value function

$$V^{\pi}(s_t) = \mathbb{E}_{\mathcal{P}, \boldsymbol{a}_t\sim\pi(\cdot|\boldsymbol{o}_t)}\left[\sum_{k=t}^{\infty}\gamma^{k-t}\mathcal{R}\left(s_k, \boldsymbol{a}_k\right)|s_t, \pi\right], \quad (2)$$

the joint state-value function $V^\pi(s_t)$ is exactly a comprehensive evaluation of J from s_t, and is commonly used for calculating the advantage value of different actions at s_t as a baseline [43]. Using VD, we can also represent the individual state-value function of agent i as

$$V^{\pi^{(i)}}\left(o_t^{(i)}\right) = \mathbb{E}_{\mathcal{P}, a_t\sim\pi^{(i)}(\cdot|o_t^{(i)})}\left[\sum_{k=t}^{\infty}\gamma^{k-t}r_k^{(i)}|o_t^{(i)}, \pi^{(i)}\right], \quad (3)$$

which can be approximated by the individual agent and learned from the joint function in (2).

B. System Model

We primarily consider a mixed autonomy traffic system model that permits vehicles to flow in and out of road, leading to the changes in the number of vehicles. At time-step t, there are totally M_t vehicles (including N DRL-driven vehicles and $M_t \geq N$) as well as intersections or ramps as depicted in Fig. 1. For vehicle i, it has access to the velocity $v_t^{(i)} \in \mathbb{R}$ and the position $z_t^{(i)} = (x_t^{(i)}, y_t^{(i)}) \in \mathbb{R}^2$ of itself at t. Meanwhile, $v_t^{(i)}$ is controlled via the acceleration $u_t^{(i)} \in \mathbb{R}$ decided by vehicle i itself. It's also necessary for vehicles to obey some traffic rules. For example, at an intersection, the vehicle has to consider an appropriate passing order as well as control its direction and velocity so as to avoid collisions. Based on the above definition, we have the elements of the corresponding Dec-POMDP as follows.

1) State and Observation: In our scenario, each vehicle can receive the information of other accessible vehicles through

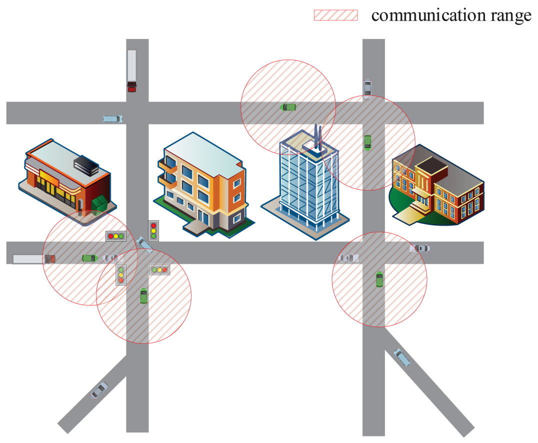


Fig. 1. Traffic control scenario with ramps, intersections and vehicles with a constrained communication range.

Vehicle-to-Vehicle (V2V) connections. Without loss of generality, assuming that vehicle i is only able to communicate with vehicles j and k at t, the observation of vehicle i can be represented as $o_t^{(i)} = \{v_t^{(i)}, z_t^{(i)}, v_t^{(j)}, z_t^{(j)}, v_t^{(k)}, z_t^{(k)}\}$. The state consists of the velocities and positions of all vehicles and can be represented by $s_t = \{v_t^{(1)}, z_t^{(1)}, v_t^{(2)}, z_t^{(2)}, \ldots, v_t^{(M_t)}, z_t^{(M_t)}\}$. Intuitively, the observation $o_t^{(i)}$ is a part of s_t, which can be mathematically determined by $\mathcal{O}(o_t^{(i)}|s_t, i)$.

2) Action: At time-step t, vehicle i adopts an action represented as $a_t^{(i)} = \{u_t^{(i)}, q_t^{(i)}\}$, where $q_t^{(i)}$ is an extra action that indicates whether the vehicle changes to another lane or direction. In particular, the N DRL-driven vehicles select their actions according to the learned policies instead of the others with fixed policies.

3) Reward: The goal of MARL for each vehicle is to maintain a velocity as close to the pre-set desired velocity v_{d} as possible on the basis of no undesirable collisions. Accordingly, the reward $r_t := \mathcal{R}(s_t, \boldsymbol{a}_t)$ is set as

$$r_t = \begin{cases} \frac{\sqrt{M_t}v_{\mathrm{d}} - \sqrt{\sum_{i=1}^{M_t}(v_{\mathrm{d}} - v_t^{(i)})^2}}{\sqrt{M_t}v_{\mathrm{d}} + C_1} & \\ -\alpha D(z^{(1)}, \ldots, z^{(M_t)}), & \text{if no collision;} \\ 0, & \text{otherwise,} \end{cases} \quad (4)$$

where C_1 is a very small constant. $D(z^{(1)}, \ldots, z^{(M_t)})$ is the penalty term with a safety coefficient α that hopes the vehicle to hold a safe distance with its preceding vehicle. (4) implies that MARL encourages the vehicle to choose a satisfactory velocity which deviates trivially from v_{d} while spares no efforts to avoid collisions at t. Notably, all the M_t vehicles participate in the calculation of the global reward.

On the other hand, the wireless-connected intelligent vehicles in IoV constitute a time-varying undirected graph $\mathcal{G}_t = \{\mathcal{V}, \mathcal{E}_t\}$, where each agent is denoted as a vertex and a communication link is regarded as an edge. In other words, $\mathcal{V} = \{1, 2, \ldots, N\}$ while an undirected edge in $\mathcal{E}_t$ corresponds to a V2V connection between two vehicles. It is worth recalling that the mobility of

Authorized licensed use limited to: Zhejiang University. Downloaded on February 14,2024 at 05:44:41 UTC from IEEE Xplore. Restrictions apply.

vehicles will frequently change the connections among vehicles due to the limited communication range and thus induce topological changes. For example, if a vehicle meets another vehicle at an intersection, it's obvious that a strong connection between them could be set up in order to avoid collisions. Additionally, the time-varying number of vehicles also results in the fluctuation of the number of vertices in $\mathcal{G}_t$. For simplicity and without losing the rationality, we will treat the vertices of $\mathcal{G}_t$ as fixed.

C. Problem Formulation

Following the system model, we expect the DRL-driven vehicle to maintain a regular speed that is close to its desired velocity on the premise of avoiding collisions. Therefore we have proposed a utility function U as the objective of policy optimization, which can be mathematically formulated as

$$\max_{\pi} \ U = \mathbb{E}_t \left[r_t | \pi \right]. \tag{5}$$

Here $\mathbb{E}_t(\cdot)$ means the expectation in a trajectory. In other words, the agents aim to learn a joint policy π under the guidance of maximizing the utility (i.e., the expected reward). Notably, for CTDE, the optimization of π is equivalent to learn the optimal individual policies of agents [17]. Therefore, (5) implies to devise an appropriate VD method for CTDE in volatile environment, so as to derive the optimal individual policy of each agent.

IV. Architecture and Designation of SVMIX

In this section, we describe the architecture and components of the proposed SVMIX algorithm as depicted in Fig. 2. First, we use PPO as the basis of each individual DRL agent. Afterwards, in order to attain proper individual state-value functions of the N agents from the total (or joint) state-value function through the VMIX network, we highlight how to leverage SGNN [25] to effectively capture the topological features from the dynamic graph $\mathcal{G}_t$.

Next we will introduce these essential components (i.e., PPO, SGNN and VMIX) in SVMIX as well as the training procedure. As SGNN is the centerpiece of our algorithm, we also mathematically explain why SGNN can help PPO agents to learn the appropriate solutions.

A. PPO-Based RL Agents

Belonging to one of the most famous Policy Gradient (PG) algorithms extensively used for continuous action control, PPO is composed of an actor network and a critic network, where the former is responsible for taking actions in accordance with its learned policy while the latter produces an approximated individual state-value function. Notably, in this article, we consider a DRL-driven vehicle and a PPO-based agent are equivalent.

As illustrated in Fig. 3, for agent i, the actor outputs the mean $\mu_i(o_t^{(i)})$ and the standard deviation $\sigma_i(o_t^{(i)})$ of a normal distribution $\mathcal{N}(\mu_i(o_t^{(i)}), \sigma_i^2(o_t^{(i)}))$ (i.e. the policy $\pi^{(i)}(\cdot|o_t^{(i)})$) through a Multi-Layer Perception (MLP) taking $o_t^{(i)}$ as input with nonlinear activation functions (omitted from Fig. 3 for

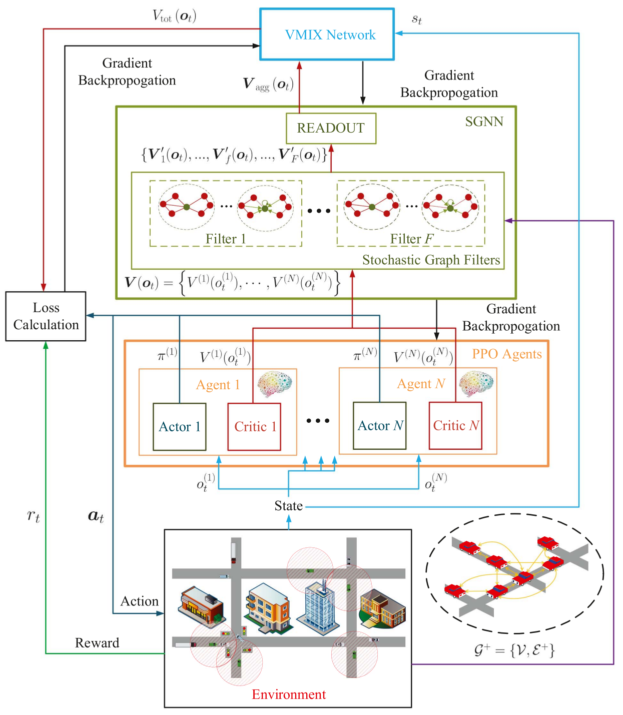
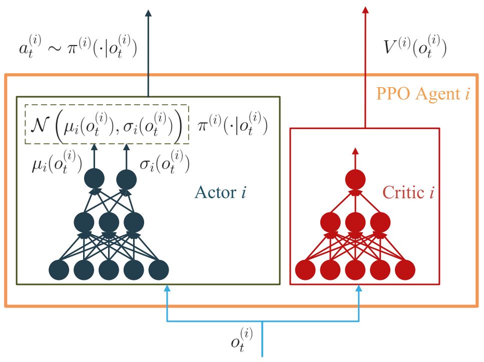


Fig. 2. Illustration of the SVMIX algorithm for autonomous vehicle control. PPO agents take $\boldsymbol{a}_t$ and approximate the state-values $\boldsymbol{V}(\boldsymbol{o}_t)$ according to local observations $\boldsymbol{o}_t$. Both SGNN and VMIX play the role of value decomposition, by sequentially processing the raw individual state-values.



$a_t^{(i)} \sim \pi^{(i)}(\cdot|o_t^{(i)})$
$V^{(i)}(o_t^{(i)})$
PPO Agent i
$\mathcal{N}\left(\mu_i(o_t^{(i)}), \sigma_i(o_t^{(i)})\right)$
$\pi^{(i)}(\cdot|o_t^{(i)})$
$\mu_i(o_t^{(i)})$
$\sigma_i(o_t^{(i)})$
Actor i
Critic i
$o_t^{(i)}$

Fig. 3. PPO agent within the actor-critic architecture.

simplicity). Sequentially, following $\pi^{(i)}(\cdot|o_t^{(i)})$, the PPO agent samples an action $a_t^{(i)}$ from $\mathcal{N}(\mu_i(o_t^{(i)}), \sigma_i^2(o_t^{(i)}))$ at t for balancing the exploration and the exploitation. Besides, the critic approximates the state-value function and produces a state-value $V^{(i)}(o_t^{(i)})$ through another MLP.

After that, the joint action $\boldsymbol{a}_t$ is taken and the environment correspondingly enters the next state s_{t+1}, thus shifting the next joint observation to $\boldsymbol{o}_{t+1}$ and yielding a global reward r_t. Notably, during this procedure, to better represent the total value function related to the dynamic graph, individual state-values are further processed through SGNN-based feature extraction and aggregation.

Authorized licensed use limited to: Zhejiang University. Downloaded on February 14,2024 at 05:44:41 UTC from IEEE Xplore. Restrictions apply.

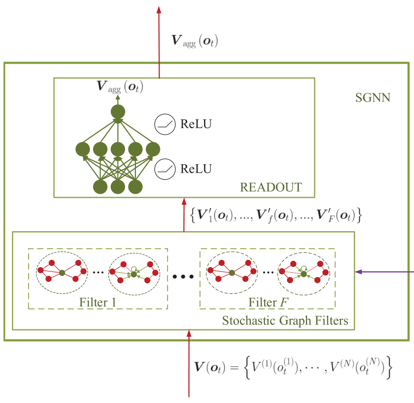

$\boldsymbol{V}_{\text{agg}}(\boldsymbol{o}_t)$
SGNN
ReLU
ReLU
READOUT
$\{\boldsymbol{V}'_1(\boldsymbol{o}_t),\ldots,\boldsymbol{V}'_f(\boldsymbol{o}_t),\ldots,\boldsymbol{V}'_F(\boldsymbol{o}_t)\}$
Filter 1
Filter F
Stochastic Graph Filters
$\mathcal{G}^+=\{\mathcal{V},\mathcal{E}^+\}$
$\boldsymbol{V}(\boldsymbol{o}_t)=\{V^{(1)}(o_t^{(1)}),\cdots,V^{(N)}(o_t^{(N)})\}$

Fig. 4. Structure of SGNN, which consists of several parallel stochastic graph filters and an MLP for signal aggregation.

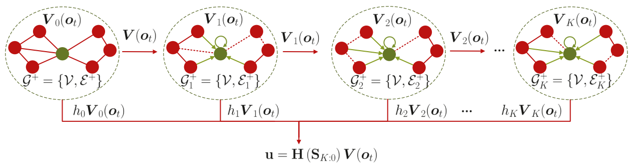

$\boldsymbol{V}_0(\boldsymbol{o}_t)$
$\boldsymbol{V}(\boldsymbol{o}_t)$
$\boldsymbol{V}_1(\boldsymbol{o}_t)$
$\boldsymbol{V}_1(\boldsymbol{o}_t)$
$\boldsymbol{V}_2(\boldsymbol{o}_t)$
$\boldsymbol{V}_2(\boldsymbol{o}_t)$
$\boldsymbol{V}_K(\boldsymbol{o}_t)$
$\mathcal{G}^+=\{\mathcal{V},\mathcal{E}^+\}$
$\mathcal{G}_1^+=\{\mathcal{V},\mathcal{E}_1^+\}$
$\mathcal{G}_2^+=\{\mathcal{V},\mathcal{E}_2^+\}$
$\mathcal{G}_K^+=\{\mathcal{V},\mathcal{E}_K^+\}$
$h_0\boldsymbol{V}_0(\boldsymbol{o}_t)$
$h_1\boldsymbol{V}_1(\boldsymbol{o}_t)$
$h_2\boldsymbol{V}_2(\boldsymbol{o}_t)$
$h_K\boldsymbol{V}_K(\boldsymbol{o}_t)$
$\mathbf{u}=\mathbf{H}(\mathbf{S}_{K:0})\boldsymbol{V}(\boldsymbol{o}_t)$

Fig. 5. Stochastic graph filter of K orders, where $\mathcal{G}_1^+,\ldots,\mathcal{G}_K^+$ are the sampled sub-graphs based on $\mathcal{G}^+$ through RES. Here we only show how the information of the green vertex is updated via green edges.

B. Topological Feature Extraction and Filtering by SGNN

As mentioned before, the vehicles can form an undirected graph $\mathcal{G}_t=\{\mathcal{V},\mathcal{E}_t\}$. Thus motivated by the extraordinary achievements of GNN to tackle the topology issues, we propose to apply graph signal processing techniques for feature extraction, by treating each individual state-value $V^{(i)}(o_t^{(i)})$ as the signal of one corresponding vertex in $\mathcal{G}_t$. Furthermore, we argue that SGNN makes a good complement to deal with the dynamic graph in traffic control.

Fig. 4 demonstrates that SGNN consists of stochastic graph filters and the READOUT mechanism for output integration. SGNN mainly adopts stochastic graph filters to fit the environment volatility based on the Random Edge Sampling (RES) model [24] with the underlying graph $\mathcal{G}^+=\{\mathcal{V},\mathcal{E}^+\}$, wherein $\mathcal{E}^+$ encompasses all sets of edges $\mathcal{E}_t$ for all t (i.e., $\mathcal{E}_t\subseteq\mathcal{E}^+,\forall t$). Besides, we define the adjacent matrix $\mathbf{A}\in\mathbb{R}^{N\times N}$ where an entry $\mathbf{A}_{m,n}$ is nonzero if and only if $(m,n)\in\mathcal{E}^+$, as well as the Laplacian matrix $\mathbf{L}=\operatorname{diag}(\mathbf{A}\mathbf{1})-\mathbf{A}$ for $\mathcal{G}^+$. In particular, a random sub-graph $\mathcal{G}_k^+=\{\mathcal{V},\mathcal{E}_k^+\}$ will be constructed by holding the original vertices of $\mathcal{G}^+$ but sampling the edges from $\mathcal{E}^+$ following a Bernoulli distribution with a success probability p. That is,

$$\Pr\left[(m,n)\in\mathcal{E}_k^+\right]=p,\quad\text{for all }(m,n)\in\mathcal{E}^+. \tag{6}$$

Through RES, we can emulate dynamic graphs where the V2V connections (edges) vary frequently. Therefore, to exert the advantage of centralized training, we actually let $\mathcal{G}^+$ be a generalized fixed graph encompassing all potential connections between vertices, so as to ensure enough exploration.

In detail, Fig. 5 demonstrates the corresponding structure of a K-order stochastic graph filter. Based on the input $\boldsymbol{V}(\boldsymbol{o}_t)=\{V^{(1)}(o_t^{(1)}),\ldots,V^{(N)}(o_t^{(N)})\}\in\mathbb{R}^N$ from PPO agents as well as the stochastic sub-graphs $\mathcal{G}_k^+$ corresponding to the adjacent matrix $\mathbf{A}_k$, the information among vertices recursively diffuses as $\boldsymbol{V}_k(\boldsymbol{o}_t)=\mathbf{S}_k\boldsymbol{V}_{k-1}(\boldsymbol{o}_t),k\in\{0,\ldots,K\}$ while $\mathbf{S}_0=\mathbf{I}$ (where $\mathbf{I}$ denotes the identity matrix) and $\boldsymbol{V}_0(\boldsymbol{o}_t)=\mathbf{S}_0\boldsymbol{V}(\boldsymbol{o}_t)=\boldsymbol{V}(\boldsymbol{o}_t)$ for $k=0$. Here $\mathbf{S}$ denotes the shift matrix and we can either have $\mathbf{S}_k=\mathbf{A}_k+\mathbf{I}$ or $\mathbf{S}_k=\mathbf{L}_k$. Therefore, unrolling the recursion, the intermediate signal can be represented as

$$\begin{aligned}\boldsymbol{V}_k(\boldsymbol{o}_t)=\mathbf{S}_k\boldsymbol{V}_{k-1}(\boldsymbol{o}_t)&=(\mathbf{S}_k\mathbf{S}_{k-1}\cdots\mathbf{S}_0)\boldsymbol{V}(\boldsymbol{o}_t)\\&:=\mathbf{S}_{k:0}\boldsymbol{V}(\boldsymbol{o}_t),\end{aligned} \tag{7}$$

where $\mathbf{S}_{k:0}$ is defined as $\mathbf{S}_{k:0}:=\mathbf{S}_k\mathbf{S}_{k-1}\cdots\mathbf{S}_0$. Hence, $\boldsymbol{V}_k(\boldsymbol{o}_t)$ aggregates the information of $\boldsymbol{V}(\boldsymbol{o}_t)$ through the random sequence of sub-graphs $\mathcal{G}_1^+,\ldots,\mathcal{G}_k^+$. Ultimately, the output $\mathbf{u}$ of a K-order stochastic graph filter with filter coefficients $\{h_k\}_{k=0}^K$ can be formulated as

$$\begin{aligned}\mathbf{u}=\sum_{k=0}^{K}h_k\boldsymbol{V}_k(\boldsymbol{o}_t)&=\sum_{k=0}^{K}h_k\mathbf{S}_{k:0}\boldsymbol{V}(\boldsymbol{o}_t)\\&:=\mathbf{H}\left(\mathbf{S}_{K:0}\right)\boldsymbol{V}(\boldsymbol{o}_t).\end{aligned} \tag{8}$$

Specifically, $\mathbf{H}(\mathbf{S}_{K:0})$ denotes the stochastic graph filter with learnable parameters $h_0,\ldots,h_K$.

Furthermore, we can use F parallel stochastic graph filters $\{\mathbf{H}(\mathbf{S}_{K:0})\}_{f=0}^F$ with coefficients $\mathcal{H}=\{h_{fk}\}_{f=1,k=0}^{F,K}$ to process $\boldsymbol{V}(\boldsymbol{o}_t)$. In particular, for the f-th stochastic graph filter, the output can be given on top of (8) as

$$\begin{aligned}\boldsymbol{V}'_f(\boldsymbol{o}_t)=\sigma\left[\mathbf{u}_f\right]&=\sigma\left[\mathbf{H}_f\left(\mathbf{S}_{K:0}\right)\boldsymbol{V}(\boldsymbol{o}_t)\right]\\&=\sigma\left[\sum_{k=0}^{K}h_{fk}\mathbf{S}_{f,k:0}\boldsymbol{V}(\boldsymbol{o}_t)\right],\end{aligned} \tag{9}$$

where $\sigma(\cdot)$ is the nonlinear activation function ReLU.

Subsequently, the mechanism $\texttt{READOUT}(\cdot):\mathbb{R}^{N\times F}\to\mathbb{R}^N$ is implemented to integrate the outputs of the stochastic graph filters through a two-layer MLP as

$$\boldsymbol{V}_{\text{agg}}(\boldsymbol{o}_t)=\texttt{READOUT}(\{\boldsymbol{V}'_1(\boldsymbol{o}_t),\ldots,\boldsymbol{V}'_F(\boldsymbol{o}_t)\}). \tag{10}$$

Finally, SGNN captures dynamic topological features through stochastic graph filters with learnable filter coefficients $\mathcal{H}$ and random sub-graphs generated by RES. That is, the filtered state-values $\boldsymbol{V}_{\text{agg}}(\boldsymbol{o}_t)$ involve dynamic topological features. Notably, consistent with methodology of CTDE, SGNN requires the information in every vertex as input in the centralized training phase, while it can be neglected at the decentralized execution phase.

C. The Role of SGNN

Intuitively, RES in SGNN brings substantial uncertainty to capture underlying topological features. Therefore, it's necessary to clarify the role of SGNN in SVMIX with theoretical analysis. Concentrating on the stochastic graph filter defined

Authorized licensed use limited to: Zhejiang University. Downloaded on February 14,2024 at 05:44:41 UTC from IEEE Xplore. Restrictions apply.

in (8) on the basis of RES, for a given graph $\mathcal{G}^+$ and $\boldsymbol{V}(\boldsymbol{o}_t)$ as input, the expected output can be given by

$$\mathbb{E}[\mathbf{u}] = \mathbb{E}[\mathbf{H}(\mathbf{S}_{K:0})\boldsymbol{V}(\boldsymbol{o}_t)] = \mathbb{E}\left[\sum_{k=0}^{K} h_k \mathbf{S}_{k:0}\right]\boldsymbol{V}(\boldsymbol{o}_t). \quad (11)$$

Here $\mathbb{E}$ specially denotes the expectation according to RES for a certain $\boldsymbol{V}(\boldsymbol{o}_t)$. Let $\mathbf{S}_k = \overline{\mathbf{S}} + \Delta_k$ where $\overline{\mathbf{S}} = \mathbb{E}[\mathbf{S}_k]$ and Δ represents the error matrix, we further get $\mathbb{E}[\Delta_k] = 0$ and

$$\begin{aligned}\mathbb{E}&\left[\sum_{k=0}^{K} h_k \mathbf{S}_{k:0}\right]\\ &= \sum_{k=0}^{K} h_k \mathbb{E}[\mathbf{S}_k\mathbf{S}_{k-1}\cdots\mathbf{S}_0]\\ &= \sum_{k=1}^{K} h_k \mathbb{E}\left[(\overline{\mathbf{S}}+\Delta_k)\cdots(\overline{\mathbf{S}}+\Delta_1)\right] + h_0\mathbf{I}\\ &= \sum_{k=1}^{K} h_k \overline{\mathbf{S}}^k + h_0\mathbf{I} \quad (12)\end{aligned}$$

as $\mathbb{E}[\Delta_m\Delta_n] = \mathbb{E}[\Delta_m]\mathbb{E}[\Delta_n] = 0$ if $m \neq n$ given the mutual independence between the samplings.

1) If $\mathbf{S}_k = \mathbf{L}_k$, for an entry $\overline{\mathbf{S}}_{m,n}$ in $\overline{\mathbf{S}}$, we have

$$\overline{\mathbf{S}}_{m,n} = \begin{cases} -p\mathbf{A}_{m,n}, & m \neq n;\\ pd_m, & \text{else}, \end{cases} \quad (13)$$

where d_m represents the degree of vertex m. Therefore, $\overline{\mathbf{S}} = p\mathbf{L}$, and finally we get $\mathbb{E}[\mathbf{u}] = (\sum_{k=1}^{K} h_k p^k \mathbf{L}^k + h_0\mathbf{I})\boldsymbol{V}(\boldsymbol{o}_t)$.

As $\mathbf{L}$ is a real symmetry matrix, it can be transformed into $\mathbf{L} = \mathbf{P}\Sigma\mathbf{P}^T$ through eigendecomposition where $\Sigma = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_N)$ with $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_N$. Notably, $\lambda_N \equiv 0$ as $\mathbf{L}$ always has an eigenvector $\mathbf{1}$ corresponding to the eigenvalue 0, and the algebraic multiplicity of λ_N equals the number of connected components in $\mathcal{G}^+$ [44]. Considering the influence of filtering coefficients $\{h_k\}_{k=0}^K$ and probability p, $\sum_{k=1}^K h_k p^k \mathbf{L}^k + h_0\mathbf{I}$ will be a full-rank matrix (e.g., if $h_k > 0$ for $k = 0, 1, \ldots, K$, the eigenvalues of $\sum_{k=1}^K h_k \overline{\mathbf{S}}^k + h_0\mathbf{I}$ can be $\tilde{\lambda}_1 \geq \tilde{\lambda}_2 \geq \cdots \geq \tilde{\lambda}_N = h_0 > 0$).

2) If $\mathbf{S}_k = \mathbf{A}_k + \mathbf{I}$, we have $\overline{\mathbf{S}}_{m,n}$ as

$$\overline{\mathbf{S}}_{m,n} = \mathbb{E}[\mathbf{A}_k + \mathbf{I}]_{m,n} = \begin{cases} p\mathbf{A}_{m,n}, & m \neq n;\\ 1, & \text{else}. \end{cases} \quad (14)$$

We can also express the real symmetry matrix as $\overline{\mathbf{S}} = \mathbf{P}\Sigma\mathbf{P}^T$ via eigendecomposition where $\Sigma = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_N)$. Assuming that $\mathcal{G}^+$ is a fully connected graph, there will be $\lambda_1 = 1 + (N-1)p$ and $\lambda_2 = \lambda_3 = \cdots = \lambda_N = 1 - p$. Thus, choosing a feasible p and taking $\{h_k\}_{k=0}^K$ into account, $\sum_{k=1}^K h_k \overline{\mathbf{S}}^k + h_0\mathbf{I}$ can be a full-rank matrix as well.

Based on the above analysis, we have proved that $\mathbb{E}[\mathbf{H}(\mathbf{S}_{K:0})]$ can be full-rank with appropriate $\{h_k\}_{k=0}^K$ and p. Hence, for a given expected output $\mathbb{E}[\mathbf{u}]$, there must be a nontrivial solution $\boldsymbol{V}(\boldsymbol{o}_t)$ that satisfies (11).

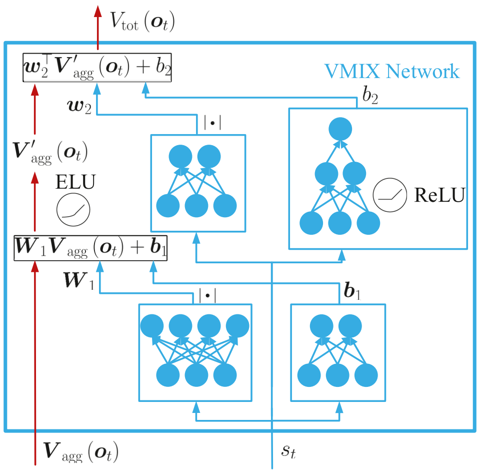

$V_{\text{tot}}(\boldsymbol{o}_t)$
$\boldsymbol{w}_2^\top \boldsymbol{V}'_{\text{agg}}(\boldsymbol{o}_t) + b_2$
VMIX Network
b_2
$\boldsymbol{w}_2$
$|\cdot|$
$\boldsymbol{V}'_{\text{agg}}(\boldsymbol{o}_t)$
ELU
ReLU
$\boldsymbol{W}_1\boldsymbol{V}_{\text{agg}}(\boldsymbol{o}_t) + \boldsymbol{b}_1$
$\boldsymbol{W}_1$
$|\cdot|$
$\boldsymbol{b}_1$
$\boldsymbol{V}_{\text{agg}}(\boldsymbol{o}_t)$
s_t

Fig. 6. Structure of the VMIX module.

Together with (9) and (10) which show the relationship between $\mathbf{u}$ and $\boldsymbol{V}_{\text{agg}}(\boldsymbol{o}_t)$, it becomes reasonable to use SGNN for feature extraction because $\boldsymbol{V}(\boldsymbol{o}_t)$ can finally be mapped to $V_{\text{agg}}^{(i)}(\boldsymbol{o}_t)$ where

$$\begin{aligned}V_{\text{agg}}^{(i)}(\boldsymbol{o}_t) &= [\boldsymbol{V}_{\text{agg}}(\boldsymbol{o}_t)]_i\\ &= \mathbb{E}_{\mathcal{P},\boldsymbol{a}_t\sim\pi(\cdot|\boldsymbol{o}_t)}\left[\sum_{k=t}^{\infty}\gamma^{k-t}\hat{r}_k^{(i)}|\boldsymbol{o}_t,\pi\right]. \quad (15)\end{aligned}$$

In other words, the individual state-value functions further take the observations and policies of other agents as well as the dynamics of topology into account. Consequently, $\hat{r}_k^{(i)}$ can be regarded as a more accurate approximation of the individual reward compared to $r_k^{(i)}$ in (2) after sufficient training.

By the way, as $\mathcal{G}^+$ is generalized for all possible $\mathcal{G}_t$ through RES during training, SGNN is able to explore the topologies that may appear, thus learning stable graph signal filters. Therefore, SVMIX ultimately obtain the capability of anti-disturbance. During this procedure, SGNN will be the important intermediate structure in SVMIX, which makes the individual state-values approximated by PPO agents more precise and decreases the difficulty of the approximation of the total value function for VMIX.

D. Value Decomposition With VMIX

Similar to the mixing network in QMIX [17], the VMIX network shown in Fig. 6 leverages the filtered signals from SGNN and further evaluates the contribution of each agent to generate the total state-value $V_{\text{tot}}(\boldsymbol{o}_t)$ on the basis of the global state s_t. VMIX consists of multiple MLPs which take s_t as input and output the weights and biases for linear transformations. Finally, the aggregated state-value can be formulated as

$$V_{\text{tot}}(\boldsymbol{o}_t) = \boldsymbol{w}_2^\top \boldsymbol{V}'_{\text{agg}}(\boldsymbol{o}_t) + b_2, \quad (16)$$

Authorized licensed use limited to: Zhejiang University. Downloaded on February 14,2024 at 05:44:41 UTC from IEEE Xplore. Restrictions apply.

where $\boldsymbol{V}'_{\text{agg}}(\boldsymbol{o}_t) = f(\boldsymbol{W}_1 \boldsymbol{V}_{\text{agg}}(\boldsymbol{o}_t) + \boldsymbol{b}_1)$. Besides, $\boldsymbol{W}_1 \in \mathbb{R}^{C\times N}$, $\boldsymbol{b}_1 \in \mathbb{R}^C$, $\boldsymbol{w}_2 \in \mathbb{R}^C$, $b_2 \in \mathbb{R}$ are the generated hyperparameters indicating non-negative weights and biases from MLPs taking s_t as input while the superscript $\top$ means the transpose. Besides, $f(\cdot)$ denotes the nonlinear ELU function and enables VMIX to produce a nonlinear total value function $V_{\text{tot}}(\cdot)$.

Thus via VMIX, the aggregated individual state-values $\boldsymbol{V}_{\text{agg}}(\boldsymbol{o}_t)$ with captured topological features from SGNN are further integrated into the total state-value $V_{\text{tot}}(\boldsymbol{o}_t)$ via nonlinear transformation, where the weights and biases learn the contribution of each agent under the guidance of the global state s_t. Conversely, $V^{(i)}(o_t^{(i)})$ is decomposed from $V_{\text{tot}}(\boldsymbol{o}_t)$ via $\boldsymbol{V}_{\text{agg}}(\boldsymbol{o}_t)$. Consequently, through gradient descent and back propagation, the parameters of each critic will be updated and thus an appropriate individual state-value functions $V^{(i)}(\cdot)$ will be learned.

E. The Training of SVMIX

The SVMIX network aims to learn a joint policy $\pi(\cdot|\boldsymbol{o}_t;\theta) = \prod_{i=1}^N \pi^{(i)}(\cdot|o_t^{(i)};\theta^{(i)})$ (i.e., the normal distribution $\mathcal{N}(\mu_i(o_t^{(i)}), \sigma_i^2(o_t^{(i)}))$ for $i = 1,\ldots,N$) and a total state-value function $V_{\text{tot}}(\cdot;\phi,\eta,\omega)$ based on decentralized critics, where $\theta = \langle \theta^{(1)},\ldots,\theta^{(N)}\rangle$ and $\phi = \langle \phi^{(1)},\ldots,\phi^{(N)}\rangle$ are the learnable parameters of actors and critics respectively. Besides, η and ω indicate the parameters of SGNN and VMIX, respectively. According to $\pi(\cdot|\boldsymbol{o}_t;\theta)$, the joint action $\boldsymbol{a}_t$ is adopted. In particular, $\boldsymbol{a}_t$ indicates the taken actions of vehicles at t while $\pi(\boldsymbol{a}_t|\boldsymbol{o}_t;\theta)$ and $V_{\text{tot}}(\boldsymbol{o}_t;\phi,\eta,\omega)$ participates in the calculation of loss functions along with the global reward r_t. Inspired by [26], the loss functions are defined as

$$\mathcal{L}_{\text{clip}}(\theta) = -\mathbb{E}_t[\min(\rho_{t;\theta} A_{t;\phi,\eta,\omega}, \text{clip}(\rho_{t;\theta},\epsilon) A_{t;\phi,\eta,\omega})], \quad (17)$$

$$\mathcal{L}_{\text{vf}}(\phi,\eta,\omega) = \frac{1}{2} A^2_{t;\phi,\eta,\omega}, \quad (18)$$

where

$$\rho_{t;\theta} = \frac{\pi(\boldsymbol{a}_t|\boldsymbol{o}_t;\theta)}{\pi(\boldsymbol{a}_t|\boldsymbol{o}_t;\theta_{\text{old}})} \quad (19)$$

and

$$A_{t;\phi,\eta,\omega} = \sum_{k=t}^{T} \gamma^{k-t} r_k - V_{\text{tot}}(\boldsymbol{o}_t;\phi,\eta,\omega). \quad (20)$$

Here T is the length of the episode, the clip function $\text{clip}(\cdot)$ removes the incentive for moving the ratio ρ_t outside of the interval $[1-\epsilon, 1+\epsilon]$ and A_t is the advantage value that evaluates the current policy in the form of Monte-Carlo error. Notably, for training with batches that include time-related samples, (20) will further be slightly modified as

$$A_{j;\phi,\eta,\omega} = \sum_{k=j}^{|\mathcal{B}|} \gamma^{k-j} r_k + \gamma^{|\mathcal{B}|+1-j} V_{\text{tot}}(\boldsymbol{o}_{|\mathcal{B}|+1};\phi,\eta,\omega) - V_{\text{tot}}(\boldsymbol{o}_j;\phi,\eta,\omega), \quad (21)$$

where $|\mathcal{B}|$ is the batch size and $j \in \{1,\ldots,|\mathcal{B}|\}$ is the index of samples. If the episode ends, (21) will degenerate into the form of (20) as the approximation of subsequent rewards is no longer needed. Finally, θ and ϕ,η,ω will be updated through gradient descent to minimize (17) and (18) separately.

Algorithm 1: The Training of SVMIX Algorithm.

```
1:  Initialize the underlying graph $\mathcal{G}^+ = \{\mathcal{V}, \mathcal{E}^+\}$, the
    number of agents $N$, the batch size $|\mathcal{B}|$, the length of
    an episode $T$, the number of PPO epochs $N_{\text{epoch}}$,
    discount factor $\gamma$ and constant $\epsilon$;
2:  Initialize the actor and critic networks in PPO agents,
    the SGNN network and the VMIX network with
    random parameters $\theta$, $\phi$, $\eta$ and $\omega$ respectively;
3:  Initialize the batch $\mathcal{B} \leftarrow \varnothing$ and the batch sample
    counter as counter $\leftarrow 0$;
4:  for every episode do
5:    Initialize the ending flag of the episode as
      done $\leftarrow 0$;
6:    for $t \leftarrow 1$ to $T$ do
7:      Obtain the global state $s_t$ and the joint observation
        $\boldsymbol{o}_t = \{o_t^{(1)},\ldots,o_t^{(N)}\}$ from the environment;
8:      Each agent generates the mean $\mu_i(o_t^{(i)};\theta^{(i)})$ and
        the standard deviation $\sigma_i(o_t^{(i)};\theta^{(i)})$ of the normal
        distribution $\mathcal{N}(\mu_i(o_t^{(i)};\theta^{(i)}), \sigma_i^2(o_t^{(i)};\theta^{(i)}))$;
9:      Each agent chooses the sampled action
        $a_t^{(i)} \sim \mathcal{N}(\mu_i(o_t^{(i)};\theta^{(i)}), \sigma_i^2(o_t^{(i)};\theta^{(i)}))$;
10:     Obtain the reward $r_t$, $s_{t+1}$ and $\boldsymbol{o}_{t+1}$;
11:     Store the tuple $\langle s_t, \boldsymbol{o}_t, \boldsymbol{a}_t, r_t, s_{t+1}, \boldsymbol{o}_{t+1}\rangle$ in $\mathcal{B}$ in
        order of time and set counter $\leftarrow$ counter $+ 1$;
12:     if $t = T$ then
13:       done $\leftarrow 1$;
14:     end if
15:     if counter $= |\mathcal{B}|$ or done $= 1$ then
16:       Clone $\theta_{\text{old}} \leftarrow \theta$;
17:       for $n_{\text{epoch}} \leftarrow 1$ to $N_{\text{epoch}}$ do
18:         for samples in $\mathcal{B}$ do
19:           Obtain $\pi(\boldsymbol{a}_j|\boldsymbol{o}_j;\theta)$ and $\pi(\boldsymbol{a}_j|\boldsymbol{o}_j;\theta_{\text{old}})$
              corresponding to the distributions generated
              from actors and calculate $\rho_{j;\theta}$ by (19);
20:           Obtain the approximated total state-value
              $V_{\text{tot}}(\boldsymbol{o}_j;\phi,\eta,\omega)$ through critics, SGNN and
              VMIX sequentially in (9)–(16);
21:           Calculate $A_{j;\phi,\eta,\omega} \leftarrow \sum_{k=j}^{|\mathcal{B}|} \gamma^{k-j} r_k + (1 -$
              done$) * \gamma^{|\mathcal{B}|+1-j} V_{\text{tot}}(\boldsymbol{o}_{|\mathcal{B}|+1};\phi,\eta,\omega) -$
              $V_{\text{tot}}(\boldsymbol{o}_j;\phi,\eta,\omega)$;
22:         end for
23:         Update $\theta$ and $\phi,\eta,\omega$ according to (17) and
            (18) separately via batch gradient descent;
24:       end for
25:       Initialize $\mathcal{B} \leftarrow \varnothing$ and counter $\leftarrow 0$;
26:     end if
27:   end for
28: end for
```

Substantially, SVMIX decomposes $V^{(i)}(o_t^{(i)})$ from $V_{\text{tot}}(\boldsymbol{o}_t)$ through gradient descent on the minimization of (18) via SGNN and VMIX. In particular, with the aid of SGNN, a feasible

Authorized licensed use limited to: Zhejiang University. Downloaded on February 14,2024 at 05:44:41 UTC from IEEE Xplore. Restrictions apply.

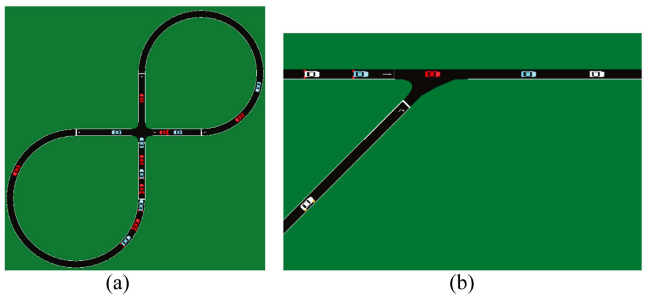


Fig. 7. Two scenarios for simulations. Here the red vehicles are the DRL-driven vehicles, the blue vehicles are the manned vehicles observed by the DRL-driven vehicles while the white vehicles are the manned vehicles which are not observed in the state space. (a) "Figure Eight". (b) "Merge".

mapping from $\boldsymbol{V}(\boldsymbol{o}_t)$ to $V_{\text{agg}}^{(i)}(\boldsymbol{o}_t)$, which takes the dynamics of topology and the influence from other agents into account, is learned to deal with the credit assignment issue first. Afterwards, a satisfied joint policy will be learned according to (17). Additionally, the RES model in SGNN enhances the anti-disturbance ability of SVMIX and the exploration ability by capturing the dynamic topological features of $\mathcal{G}_t$ given the learned $\mathcal{H}$ and certain p. In other words, for all possible inputs $\boldsymbol{V}(\boldsymbol{o}_t)$ related to a specific topological connection, SGNN helps to finally get the corresponding state-value $V^{(i)}(o_t^{(i)})$ for each agent and handles topological dynamics based on RES. To sum up, we describe the training procedure of SVMIX algorithm in Algorithm 1.

V. Experimental Settings and Numerical Results

In this section, we try to verify the performance of SVMIX in autonomous vehicle control and demonstrate the advantage of our proposed algorithm over other MARL methods. We implement two simulation scenarios on Flow [45], [46], which is a traffic control benchmarking framework for mixed autonomy traffic. As illustrated in Fig. 7, the "Figure Eight" scenario and the "Merge" scenario are chosen as two typical cases to evaluate the performance of our method.

A. The Settings for "figure Eight" and Simulation Results

The "Figure Eight" scenario in Fig. 7(a) consists of fixed $M = 14$ (i.e., $M_t = 14$ for any t) vehicles running circularly along a one-way lane with an intersection. So the vehicle must strive for a velocity close to its desired velocity, and timely adjust the velocity when passing through the intersection so as to avoid collisions. What's more, we deploy 7 manned vehicles and $N = 7$ DRL-driven vehicles alternately, where the former vehicles are controlled by Intelligent Driver Model (IDM) defined in [8] while the latter ones are controlled by MARL methods. Notably, all the vehicles will perform emergency braking if they are about to crash, and once a collision occurs the episode will be terminated immediately. The corresponding MDP for DRL agents in "Figure Eight" is defined as below.

- *State and Observation:* Here the state contains the information of all the vehicles. Besides, each DRL-driven vehicle can only observe the information of the vehicles ahead and behind. Therefore we have $s_t = \{v_t^{(1)}, z_t^{(1)}, v_t^{(2)}, z_t^{(2)}, \ldots, v_t^{(M)}, z_t^{(M)}\}$ and $o_t^{(i)} = \{v_t^{(i_{\text{ahead},t})}, z_t^{(i_{\text{ahead},t})}, v_t^{(i)}, z_t^{(i)}, v_t^{(i_{\text{behind},t})}, z_t^{(i_{\text{behind},t})}\}$ as the observation of DRL-driven vehicle i, where $i_{\text{ahead},t}$ and $i_{\text{behind},t}$ are the preceding and following vehicles of vehicle i at time-step t, respectively.
- *Action:* As each DRL-driven vehicle only needs to consider the acceleration (i.e. $a^{(i)} = u^{(i)}$), $\boldsymbol{a}_t = \{u^{(1)}, u^{(2)}, \ldots, u^{(N)}\}$.
- *Reward:* The reward function is the same as (4) where $\alpha = 0$ so the penalty term is ignored.

In our setting, the number of episodes N_{episode} is 300, while each episode has a maximum of $L = 1500$ iterations in the case of no collision. Also, we design the utility $U = \frac{1}{L}\sum_{t=1}^{T} r_t$ as a concrete form of (5) as the evaluation metrics. For SVMIX, we use the same architecture as depicted in Fig. 2 with $F = 32$, $K = 3$ and $p = 0.7$ for SGNN. Besides, we compare the performance of SVMIX with other MARL methods, including Federated Independent Reinforcement Learning (FIRL) [27], QMIX [17] and Multi-Graph Attention Network (MGAN) [20], where FIRL combines federated learning with a consensus algorithm based on multiple PPO agents, and MGAN adhibits Graph AttenTion network (GAT) into VD-based MARL architecture. Both MGAN and SVMIX are contingent on a complete graph for inter-agent communication, while FIRL concentrates on a specified graph for consensus. Especially, FIRL uploads the gradient of each agent to a centralized virtual agent every τ updates. Besides, an optimal baseline is rendered by Flow where all the 14 vehicles are controlled by IDM, as it is a typical car-following model that contains lots of prior knowledge. It's worth noting that all the methods deal with a global reward, while each agent in FIRL uses the local reward identical to the global reward. Typical parameter settings are summarized in Table II.

TABLE II
Key Parameter Settings for "Figure Eight" and the Algorithms

Parameters Definition		Settings
Number of DRL Agents		$N = 7$
Range of Acceleration per Vehicle		$[-2\text{m/s}^2, 2\text{m/s}^2]$
Desired Speed per Vehicle		$v_{\text{d}} = 10\text{m/s}$
Speed Limit per Vehicle		Up to 30m/s
Number of Episodes		$N_{\text{episode}} = 300$
Length per Time-step		0.1s
Maximal Length per Episode		$L = 1,500$
Discount Factor		$\gamma = 0.99$
Constant in Clip Function		$\epsilon = 0.2$
Safety Coefficient in (4)		$\alpha = 0$
FIRL	Upload Period	$\tau = 10$
SVMIX	Number of Filters	$F = 32$
	Order per Filter	$K = 3$
	Success Probability for RES	$p = 0.7$
	Number of Parameters per Agent	$N_{\text{para}} = 11,653$

Authorized licensed use limited to: Zhejiang University. Downloaded on February 14,2024 at 05:44:41 UTC from IEEE Xplore. Restrictions apply.

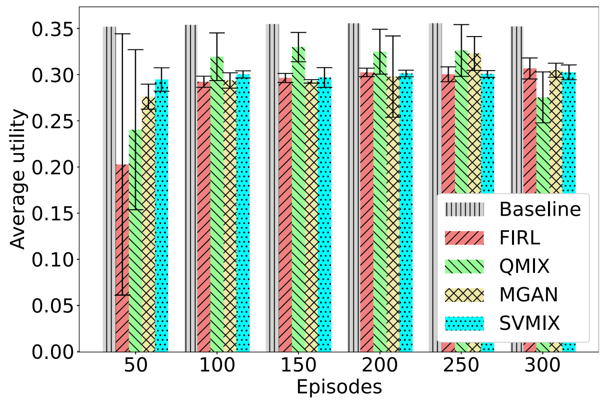


Fig. 8. Average utility of different methods under the scenario "Figure Eight" under a desired velocity $v_{\mathrm{d}} = 10\ \mathrm{m/s}$. Obviously, all methods converge within about 100 episodes, and SVMIX converges the fastest. For VD-based algorithms, the curves of QMIX and MGAN fluctuate violently and demonstrate least instability. For example, the higher average utility of MGAN comes at the cost of stability. However, SVMIX converges faster than the aforementioned methods as well as maintains a stable and satisfactory utility. FIRL learns a stable policy similar to SVMIX but converges more slowly. On the whole, SVMIX outstrips the other methods in terms of the convergence rate and stability with a higher utility.

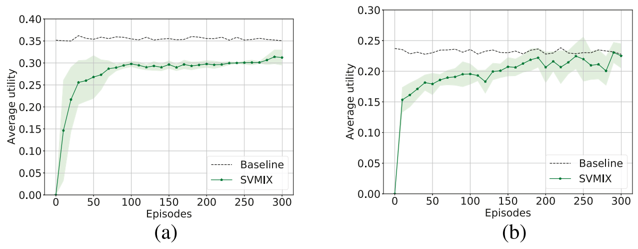


Fig. 9. Average utility curve of SVMIX under the scenario "Figure Eight" (a) under a desired velocity $v_{\mathrm{d}} = 10\ \mathrm{m/s}$; (b) under a desired velocity $v_{\mathrm{d}} = 20\ \mathrm{m/s}$. (a) $v_{\mathrm{d}} = 10\mathrm{m/s}$. (b) $v_{\mathrm{d}} = 20\mathrm{m/s}$.

Fig. 8 compares the average utility of different MARL methods and the optimal baseline in 3 independent simulations, while Fig. 9(a) further presents the learning curve of SVMIX in detail. In particular, the utility is computed by averaging the results of another five testing episodes after every 10 training episodes in a simulation, when each agent directly takes the mean $\mu_i(o_t^{(i)})$ as a deterministic action. It can be observed from Fig. 8 that both QMIX and MGAN converge with about 100 episodes of training since VD learns different contributions of agents with precise total value function and individual value functions. Compared to QMIX, MGAN converges faster and has a slightly higher average utility thanks to the captured topological features by GAT. But unfortunately, this also makes MGAN susceptible to the dynamics of graphs and more unstable. It can be observed from Table III that the fluctuation of utility in MGAN is mostly caused by the occurrence of collisions due to over-accelerated vehicles. In other words, QMIX and MGAN learn over-aggressive policies. Comparatively, benefiting from the stochastic graph filters with enhanced capability to capture dynamic topological features, SVMIX outperforms the two VD-based algorithms as it converges faster while learns a safer joint policy with significantly less fluctuations, as well as maintains a high utility.

TABLE III
PROBABILITY OF COLLISIONS FOR MARL METHODS AFTER CONVERGENCE UNDER THE SCENARIO "FIGURE EIGHT"

Desired Velocity	Methods			
	FIRL	QMIX	MGAN	SVMIX
10m/s	0.00%	4.57%	2.88%	0.00%
20m/s	1.30%	11.80%	9.27%	7.53%

TABLE IV
AVERAGE COMMUNICATION OVERHEADS OF FIRL AND SVMIX

Methods	Communication Overheads
FIRL	$\frac{1}{\tau}(N * N_{\mathrm{para}}) = 8,157.1$
SVMIX	$N * N_{\mathrm{epoch}} * \lvert\mathcal{B}\rvert = 7,168$

On the other hand, benefitting from the aggregation of gradient through federated learning and communication among agents, FIRL learns a stable policy similar to SVMIX but converges more slowly.

Moreover, as depicted in Table IV, which compares SVMIX and FIRL in terms of the uplink communication overhead[1] (i.e., the amount of data) required for training with $|\mathcal{B}|$ samples in the batch, SVMIX reduces 12.13% of the communication overheads compared to FIRL as the communication overheads of FIRL are proportional to the number of parameters (i.e., N_{para}) while those of SVMIX are proportional to the size of a batch (i.e., $|\mathcal{B}|$) and the number of PPO epochs (i.e., N_{epoch}). In other words, SVMIX only needs to transmit a batch of state-values rather than the gradients of all parameters. Hence, compared to FIRL, SVMIX better balances the learning performance and communication overheads.

Furthermore, we additionally test the MARL algorithms in the "Figure Eight" scenario with modified configurations, wherein the range of acceleration for each vehicle is expanded to $[-3\ \mathrm{m/s}^2, 3\ \mathrm{m/s}^2]$ and the desired speed is raised to $20\ \mathrm{m/s}$ (a larger yet more dangerous velocity), while the other settings remain the same as those in Table II. Intuitively, this modified scenario with a lager desired velocity implies an increased probability of collisions, consistent with the observations in Table III. Fig. 10 compares the average utilities of different MARL methods with the optimal baseline in 6 independent simulations respectively, while the learning curve of SVMIX is depicted in Fig. 9(b). It can be observed from Fig. 10 that FIRL yields less utility with the relatively stable performance and lower collision probability, while QMIX achieves a higher utility than FIRL but fluctuates drastically due to the frequent occurrence of collisions. MGAN further improves the convergence rate but still suffers from fluctuations. Instead, SVMIX still maintains good performance with a fast convergence rate and a small training variance, as well as the lower collision probability compared with QMIX and MGAN.

[1]Notably, the VD-based methods have almost the same communication overheads.

Authorized licensed use limited to: Zhejiang University. Downloaded on February 14,2024 at 05:44:41 UTC from IEEE Xplore. Restrictions apply.

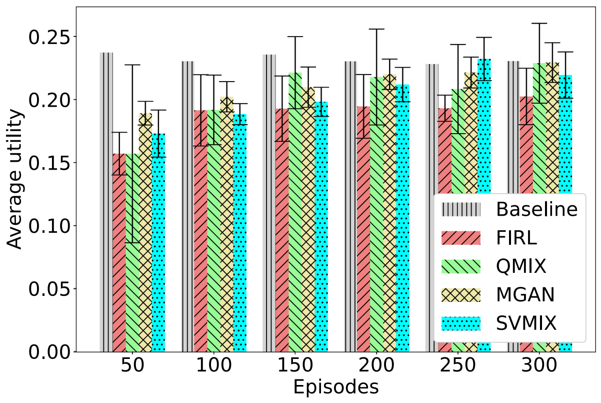


Fig. 10. Average utility of different methods under the scenario “Figure Eight” under a desired velocity $v_{\rm d} = 20$ m/s. Obviously all the methods converge within about 150 episodes, where FIRL learns a stable policy at the cost of the convergence rate and utility. QMIX achieves a higher utility than FIRL thanks to VD and MGAN further improves the convergence rate and reduces the fluctuation due to the incorporation of GAT. SVMIX has the similar performance to MGAN, but gives lower collision probability as in Table III.

B. The Settings for “merge” and Simulation Results

As depicted in Fig. 7(b), the “Merge” scenario simulates the on-ramp merging on the highway. Similar to “Figure Eight”, it’s essential for vehicles to consider how to avoid collisions and congestion due to the meeting at the merging point. As “Merge” is an unclosed network, there assumes to exist a traffic flow where the vehicles frequently flow in and out, leading to the variations in the number of vehicles. Following the definition of the model in Section III, the scenario is set to allow at most 2,100 vehicles to flow in per hour, including 2,000 vehicles at most on the trunk road and 100 vehicles at most on the ramp per hour. In particular, among the 2,000 vehicles, 25% of them will be assigned as DRL-driven vehicles. Consistent with the MDP defined in Section III, we consider an MDP as follows.

- *State and Observation:* Here the state contains the information of the observable vehicles (just like the red and blue vehicles in Fig. 7(b)) instead of all the vehicles. Also, each DRL-driven vehicle can only observe the information of the preceding and following vehicles. So s_t only consists of the positions and velocities of DRL-driven vehicles as well as the vehicles ahead of and behind them. Furthermore, we fix the number of algorithmically involved DRL-driven vehicles as $N = 13$ and if the practical number of vehicles $N_t > N$, the other $N_t - N$ vehicles will be treated as manned vehicles; otherwise, the state will be padded with zeros, as if there are N DRL-driven vehicles. The dimension of the state space consequently remain unchanged. Therefore the state is represented by $s_t = \{o_t^{(1)}, o_t^{(2)}, \ldots, o_t^{(N)}\}$ where $o_t^{(i)} = \{v_t^{(i_{\rm ahead},t)}, z_t^{(i_{\rm ahead},t)}, v_t^{(i)}, z_t^{(i)}, v_t^{(i_{\rm behind},t)}, z_t^{(i_{\rm behind},t)}\}$ expresses the observation of DRL-driven vehicle i where $i = 1, 2, \ldots, N$.
- *Action:* Same as the above, the $N = 13$ DRL-driven vehicles will choose their actions according to their policies so the dimension of the joint action space is also immutable and $\boldsymbol{a}_t = \{u^{(1)}, u^{(2)}, \ldots, u^{(N)}\}$. When $N_t > N$, the first N vehicles will be treated as DRL-driven vehicles and the others are regarded as manned vehicles. In case $N_t < N$, the extra actions will be ignored during the interaction with the environment. It’s worth noting that the underlying graph $\mathcal{G}^+$ is still a complete graph with fixed N vertices.
- *Reward:* The reward function is the same as (4) where $\alpha = 0.1$ and the penalty term is defined as

$$D(z^{(1)}, \ldots, z^{(M_t)}) = \sum_{j=1}^{M_t} \max(C_2 - \|z^{(i_{\rm ahead},t)} - z^{(i)}\|_2, 0), \tag{22}$$

where C_2 is a constant which represents the desired following distance of each vehicle.

TABLE V
KEY PARAMETER SETTINGS FOR “MERGE” AND THE ALGORITHMS

Parameters Definition		Settings
Number of DRL Agents		$N = 13$
Maximum of Vehicles per Hour		2100
Proportion of DRL-driven Vehicles		25%
Range of Acceleration per Vehicle		$[-1.5\text{m/s}^2, 1.5\text{m/s}^2]$
Desired Speed per Vehicle		$v_{\rm d} = 20\text{m/s}$
Speed Limit per Vehicle		Up to 30m/s
Number of Episodes		$N_{\rm episode} = 300$
Length of per Time-step		1s
Maximal Length per Episode		$L = 750$
Discount Factor		$\gamma = 0.99$
Constant in Clip Function		$\epsilon = 0.2$
Safety Coefficient in (4)		$\alpha = 0.1$
FIRL	Upload Period	$\tau = 10$
SVMIX	Number of Filters	$F = 32$
	Order per Filter	$K = 3$
	Success Probability for RES	$p = 0.7$

In our setting, the number of episodes $N_{\rm episode}$ is 300, while each episode has a maximum of $L = 750$ iterations in the case of no collision. For SVMIX, we use the same architecture as in Fig. 2 with $F = 32$, $K = 3$ and $p = 0.7$ for SGNN. Besides, we compare the performance of SVMIX with FIRL, QMIX and MGAN with the same detailed settings described earlier in Section V-A. Typical parameter settings are summarized in Table V.

Fig. 11 compares the average utility of different MARL methods and the optimal baseline in 3 independent simulations respectively, while the learning curve of SVMIX is depicted in Fig. 12. Consistently, the utility is computed by averaging the results of another five testing episodes after every 10 training episodes in a simulation, when each agent directly takes the mean $\mu_i(o_t^{(i)})$ as a deterministic action. It can be observed that the performance of SVMIX obviously outperforms the other algorithms in “Merge”, which verifies the aforementioned idea that it’s necessary to deal with the dynamic topology for better

Authorized licensed use limited to: Zhejiang University. Downloaded on February 14,2024 at 05:44:41 UTC from IEEE Xplore. Restrictions apply.

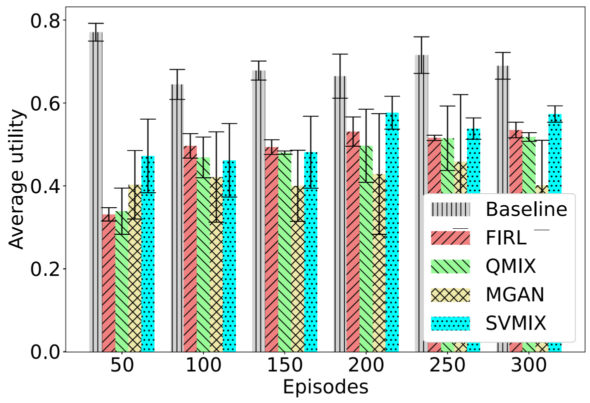


Fig. 11. Average utility of different methods under the scenario "Merge". All the methods converge within about 100 episodes, and SVMIX converges the fastest. FIRL still learns a stable policy and reaches a not bad average utility. MGAN converges quickly but seems unstable in terms of the undulant and low average utility. QMIX and SVMIX both hold a more stable performance, where the average utility of SVMIX is slightly better than QMIX.

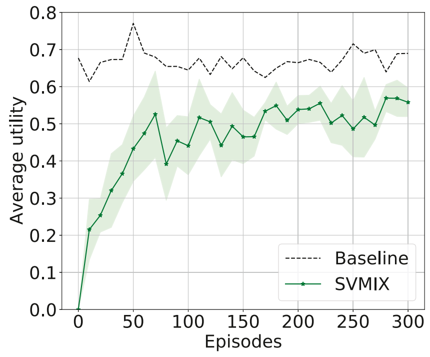


Fig. 12. Average utility curve of SVMIX under the scenario "Merge".

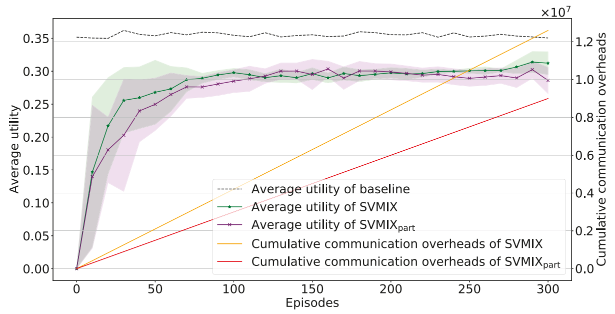


Fig. 13. Average utility as well as cumulative communication overheads of SVMIX and $\text{SVMIX}_{\text{part}}$ under the scenario "Figure Eight" under a desired velocity $v_{\text{d}} = 10$ m/s.

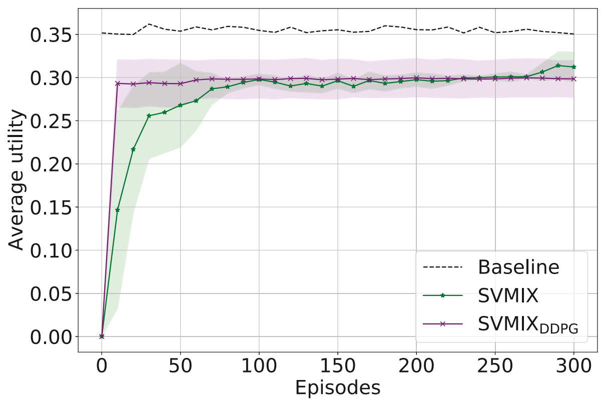


Fig. 14. Average utility of SVMIX and $\text{SVMIX}_{\text{DDPG}}$ under the scenario "Figure Eight" under a desired velocity $v_{\text{d}} = 10$ m/s.

performance of VD-based algorithms. In particular, compared with FIRL, SVMIX has comparable average utilities and still outperforms in terms of communication overheads. Moreover, MGAN achieves an unstable and the worst performance as shown in Fig. 11, and even worse than QMIX as well as FIRL. Especially at the last 150 episodes, MGAN fluctuates violently. This is possibly because the DRL-driven vehicles are irregularly allocated due to the traffic flow and it becomes a degenerative feedback to the learning of attention coefficients.

C. *Superiority of* SVMIX

We also find some interesting properties of SVMIX which have shown the superiority of SGNN and generality of SVMIX based on simulations in "Figure Eight" under the desired velocity of 10 m/s.

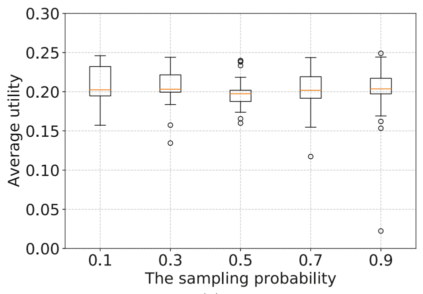
First, since SGNN simulates the dynamic graphs through RES, we speculate that SVMIX can further reduce the uplink communication overheads by uploading less data. Specifically, at each time-step t, we can sample part of the agents to upload the information (i.e., the state-value $V^{(i)}(o_t^{(i)})$) for VD, and denote this procedure as $\text{SVMIX}_{\text{part}}$. In "Figure Eight", we randomly choose 5 agents to upload their state-values for VD while leave the remaining 2 agents in silence. Fig. 13 provides the corresponding performance comparison between SVMIX and $\text{SVMIX}_{\text{part}}$ in 5 independent simulations. It can be observed that $\text{SVMIX}_{\text{part}}$ maintains almost the same performance as SVMIX. More significantly, the communication overheads of $\text{SVMIX}_{\text{part}}$ account for $\frac{5}{7}$ of those of SVMIX, which are equivalent to 62.76% of FIRL only.

We further evaluate the generality of SVMIX on top of other RL algorithms, by replacing PPO with Deep Deterministic Policy Gradient (DDPG) [16] (denoted as $\text{SVMIX}_{\text{DDPG}}$). Fig. 14 compares the average utility curves of SVMIX and $\text{SVMIX}_{\text{DDPG}}$ in 5 independent simulations. It can be observed that the extension of SGNN to DDPG is also applicable. However, compared to SVMIX with PPO, $\text{SVMIX}_{\text{DDPG}}$ yields good performance with a faster convergence rate but higher degree of instability (i.e., larger variance).

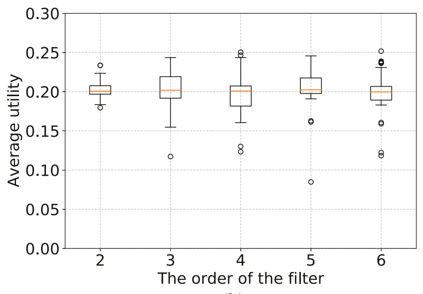
D. *Hyperparameters Adjusting for* SGNN *in* SVMIX

To clarify the influence of the hyperparameters in SGNN, we further carry out supplementary experiments in "Figure Eight" under a desired velocity of 20 m/s by changing one hyperparameter and keeping the others the identical. Fig. 15 provides the corresponding results where we get the results from the testing episodes after $100, 110, \ldots, 200$ training episodes (i.e., when SVMIX converges slowly) respectively through 5 independent simulations.

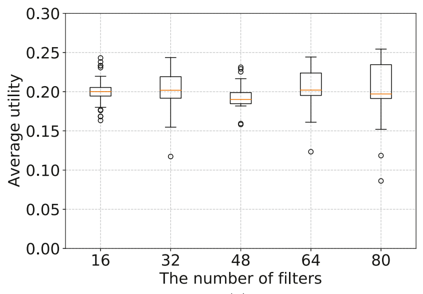
Authorized licensed use limited to: Zhejiang University. Downloaded on February 14,2024 at 05:44:41 UTC from IEEE Xplore. Restrictions apply.



(a)



(b)



(c)

Fig. 15. Boxplots of the average utility of `SVMIX` with different structure of `SGNN` under the corresponding hyperparameters by changing (a) the sampling probability p, (b) the order of each filter K and (c) the number of filters F from the testing episodes after $100, 110, \ldots, 200$ training episodes through 5 independent simulations. (a) Probability p. (b) Order of the filter K. (c) Number of filters F.

1) Probability p: As shown in Fig. 15(a), keeping $K = 3$ and $F = 32$, we evaluate the performance of SVMIX for $p \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$. For a Bernoulli-sampled RES, the variance of the outputs from SGNN is proportional to $p(1-p)$ and will be maximized when $p = 0.5$. Obviously, the result of $p = 0.5$ is the most unstable as there are many outliers. The settings of $p = 0.3$ and $p = 0.7$ yield the similar stable performance, while the performance of $p = 0.1$ and $p = 0.9$ seem more unstable. This is because when $p = 0.1$ and $p = 0.9$, SGNN is unable to make sufficient exploration due to the too small or too large probability p based on RES. Thus a probability like $p = 0.7$ sounds more appropriate for SGNN.

2) Order of the filter K: Fig. 15(b) demonstrates the influence of the filter order K by retaining $p = 0.7$ and $F = 32$ with K spanning from 2 to 6. Consistent with the analysis in [25], which states that the variance of the outputs from SGNN is proportional to K, the variance of the utility also becomes greater as K grows. Thus an order like $K = 2$ or $K = 3$ will be suitable for “Figure Eight”.

3) Number of filters F: Fig. 15(c) demonstrates the influence of the number of filters F by retaining $p = 0.7$ and $K = 3$ with $F \in \{16, 32, 48, 64, 80\}$. For $F = 16$, the capability of SGNN for feature extraction decreases and thus leads to an unstable performance. As F increases, SVMIX generally gives better performance, though it slightly suffers from the deficiency of training samples to train a larger neural network.

VI. Conclusion and Discussion

This article aims to address credit assignment problem in a dynamic environment by value decomposition and tentatively puts forward an SGNN based multi-agent actor-critic architecture called SVMIX with PPO agent as the individual agent. In particular, SGNN, which consists of parallel stochastic graph filters, has been leveraged to enhance the resilience to the environment volatility and thus capably capture dynamic underlying features in a more effective manner. To demonstrate the feasibility of SVMIX, we further explain why SGNN works by clarifying the influence of SGNN on exploring the comprehensive optimal mapping from individual state-values to the total state-value through theoretical analysis. Moreover, through extensive simulations in two different scenarios, SVMIX manifests itself with a superior capability in terms of the more balanced performance in terms of the convergence rate, the mean & the variance of the average utility and communication overheads.

However, the assumption that each agent can only make decisions based on the observation of itself might be over-emphasized as the communication between neighboring agents is allowed in practical scenarios in the decentralized execution. Therefore, it is worthwhile to incorporate the inter-agent communication more comprehensively. Meanwhile, a careful calibrated reward reshaping method, which takes account of desired velocity and collusion penalty, is meaningful as well.

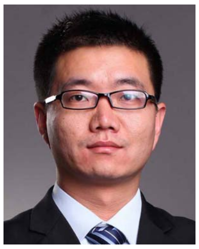
References

[1] B. Xiao et al., “Stochastic graph neural network-based value decomposition for multi-agent reinforcement learning in urban traffic control,” in *Proc. IEEE 97th Veh. Technol. Conf.*, 2023, pp. 1–7.
[2] D. González, J. Pérez, V. Milanés, and F. Nashashibi, “A review of motion planning techniques for automated vehicles,” *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 4, pp. 1135–1145, Apr. 2016.
[3] R. O’Malley et al., “Vision-based detection and tracking of vehicles to the rear with perspective correction in low-light conditions,” *IET Intell. Transport Syst.*, vol. 5, pp. 1–10, Mar. 2011.
[4] D. J. Yeong et al., “Sensor and sensor fusion technology in autonomous vehicles: A review,” *Sensors*, vol. 21, no. 6, 2021, Art. no. 2140.
[5] B. Paden, M. Čáp, S. Z. Yong, D. Yershov, and E. Frazzoli, “A survey of motion planning and control techniques for self-driving urban vehicles,” *IEEE Trans. Intell. Veh*, vol. 1, no. 1, pp. 33–55, Mar. 2016.
[6] M. Likhachev and D. Ferguson, “Planning long dynamically feasible maneuvers for autonomous vehicles,” *Int. J. Robot. Res.*, vol. 28, no. 8, pp. 933–945, 2009.
[7] Y. Kuwata, J. Teo, G. Fiore, S. Karaman, E. Frazzoli, and J. P. How, “Real-time motion planning with applications to autonomous urban driving,” *IEEE Trans. Control Syst. Technol.*, vol. 17, no. 5, pp. 1105–1118, Sep. 2009.
[8] M. Treiber et al., “Congested traffic states in empirical observations and microscopic simulations,” *Phys. Rev. E*, vol. 62, no. 2, 2000, Art. no. 1805.
[9] J. Liu, Z. Wang, and L. Zhang, “Integrated vehicle-following control for four-wheel-independent-drive electric vehicles against non-ideal V2X communication,” *IEEE Trans. Veh. Technol.*, vol. 71, no. 4, pp. 3648–3659, Apr. 2022.
[10] L. Li, Y. Lv, and F.-Y. Wang, “Traffic signal timing via deep reinforcement learning,” *IEEE/CAA J. Automatica Sinica*, vol. 3, no. 3, pp. 247–254, Jul. 2016.
[11] A. Soni and H. Hu, “Formation control for a fleet of autonomous ground vehicles: A survey,” *Robotics*, vol. 7, no. 4, 2018, Art. no. 67.
[12] K. K. Nguyen, T. Q. Duong, N. A. Vien, N.-A. Le-Khac, and M.-N. Nguyen, “Non-cooperative energy efficient power allocation game in D2D communication: A multi-agent deep reinforcement learning approach,” *IEEE Access*, vol. 7, pp. 100480–100490, 2019.

Authorized licensed use limited to: Zhejiang University. Downloaded on February 14,2024 at 05:44:41 UTC from IEEE Xplore. Restrictions apply.

[13] J. Yang, J. Zhang, and H. Wang, "Urban traffic control in software defined Internet of things via a multi-agent deep reinforcement learning approach," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 6, pp. 3742–3754, Jun. 2021.
[14] M. Tan, "Multi-agent reinforcement learning: Independent versus cooperative agents," in *Proc. 10th Int. Conf. Mach. Learn.*, 1993, pp. 330–337.
[15] J. N. Foerster et al., "Counterfactual multi-agent policy gradients," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 2974–2982.
[16] R. Lowe et al., "Multi-agent actor-critic for mixed cooperative-competitive environments," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 6382–6393.
[17] T. Rashid et al., "QMIX: Monotonic value function factorisation for deep multi-agent reinforcement learning," in *Proc. 35th Int. Conf. Mach. Learn.*, 2018, pp. 6846–6859.
[18] Y. Wang et al., "DOP: Off-policy multi-agent decomposed policy gradients," in *Proc. Int. Conf. Learn. Representations*, 2021, pp. 1–24.
[19] J. Su et al., "Value-decomposition multi-agent actor-critics," in *Proc. 35th AAAI Conf. Artif. Intell.*, 2021, pp. 11352–11360.
[20] Z. Xu et al., "Learning to coordinate via multiple graph neural networks," in *Proc. 28th Int. Conf. Neural Inf. Process.*, 2021, pp. 52–63.
[21] H. Mao et al., "Neighborhood cognition consistent multi-agent reinforcement learning," in *Proc. 34th AAAI Conf. Artif. Intell.*, 2020, pp. 7219–7226.
[22] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, "A comprehensive survey on graph neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 1, pp. 4–24, Jan. 2021.
[23] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. Int. Conf. Learn. Representations*, 2017, pp. 1–14.
[24] E. Isufi, A. Loukas, A. Simonetto, and G. Leus, "Filtering random graph processes over random time-varying graphs," *IEEE Trans. Signal Process.*, vol. 65, no. 16, pp. 4406–4421, Aug. 2017.
[25] Z. Gao, E. Isufi, and A. Ribeiro, "Stochastic graph neural networks," *IEEE Trans. Signal Process.*, vol. 69, pp. 4428–4443, 2021.
[26] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017, *arXiv:1707.06347*.
[27] X. Xu, R. Li, Z. Zhao, and H. Zhang, "Communication-efficient consensus mechanism for federated reinforcement learning," in *Proc. IEEE Int. Conf. Commun.*, 2022, pp. 80–85.
[28] W. J. Schakel, B. van Arem, and B. D. Netten, "Effects of cooperative adaptive cruise control on traffic flow stability," in *Proc. IEEE 13th Int. Conf Conf. Intell. Transp. Syst.*2010, pp. 759–764.
[29] Y. Zhou et al., "Stabilizing mixed vehicular platoons with connected automated vehicles: An h-infinity approach," *Transp. Res. Part B Methodol.*, vol. 132, pp. 152–170, 2020.
[30] X. Di and R. Shi, "A survey on autonomous vehicle control in the era of mixed-autonomy: From physics-based to AI-guided driving policy learning," *Transp. Res. Part C. Emerg. Technol.*, vol. 125, 2021, Art. no. 103008.
[31] V. Mnih et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.
[32] C.-W. Huang, C.-T. Chiang, and Q. Li, "A study of deep learning networks on mobile traffic forecasting," in *Proc. IEEE 28th Annu. Int. Symp. Pers., Indoor, Mobile Radio Commun.*, 2017, pp. 1–6.
[33] S. Yu, X. Gong, Q. Shi, X. Wang, and X. Chen, "EC-SAGINs: Edge-computing-enhanced space–air–ground-integrated networks for internet of vehicles," *IEEE Internet Things J.*, vol. 9, no. 8, pp. 5742–5754, Apr. 2022.
[34] J. Foerster et al., "Learning to communicate with deep multi-agent reinforcement learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 2145–2153.
[35] J. Jiang and Z. Lu, "Learning attentional communication for multi-agent cooperation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 7265–7275.
[36] Y. Yang et al., "Mean field multi-agent reinforcement learning," in *Proc. 35th Int. Conf. Mach. Learn.*, 2018, pp. 5571–5580.
[37] S. Bhalla et al., "Deep multi agent reinforcement learning for autonomous driving," in *Proc. Can. Conf. Artif. Intell.*, 2020, pp. 67–78.
[38] P. Palanisamy, "Multi-agent connected autonomous driving using deep reinforcement learning," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, 2020, pp. 1–7.
[39] S. Chen et al., "Graph neural network and reinforcement learning for multi-agent cooperative control of connected autonomous vehicles," *Comput.-Aided Civ. Infrastruct. Eng.*, vol. 36, no. 7, pp. 838–857, 2021.
[40] G.-P. Antonio and C. Maria-Dolores, "Multi-agent deep reinforcement learning to manage connected autonomous vehicles at tomorrow's intersections," *IEEE Trans. Veh. Technol.*, vol. 71, no. 7, pp. 7033–7043, Jul. 2022.
[41] Z. Guo, Y. Wu, L. Wang, and J. Zhang, "Coordination for connected and automated vehicles at non-signalized intersections: A value decomposition-based multiagent deep reinforcement learning approach," *IEEE Trans. Veh. Technol.*, vol. 72, no. 3, pp. 3025–3034, Mar. 2023.
[42] F. A. Oliehoek and C. Amato, *A Concise Introduction to Decentralized POMDPs*, vol. 1. Berlin, Germany: Springer, 2016.
[43] Z. Wang et al., "Dueling network architectures for deep reinforcement learning," in *Proc. 33rd Int. Conf. Mach. Learn.*, 2016, pp. 1995–2003.
[44] R. Merris, "Laplacianmatrices of graphs: A survey," *Linear Algebra Appl.*, vol. 197/198, pp. 143–176, 1994.
[45] C. Wu, A. R. Kreidieh, K. Parvate, E. Vinitsky, and A. M. Bayen, "Flow: A modular learning framework for mixed autonomy traffic," *IEEE Trans. Rob.*, vol. 38, no. 2, pp. 1270–1286, Apr. 2022.
[46] E. Vinitsky et al., "Benchmarks for reinforcement learning in mixed-autonomy traffic," in *Proc. 2nd Conf. Robot Learn.*, 2018, pp. 399–409.

Baidi Xiao (Graduate Student Member, IEEE) received the B.E. degree in information engineering from Zhejiang University, Hangzhou, China, in 2021. He is currently working toward the M.S. degree with the College of Information Science and Electronic Engineering, Zhejiang University, Hangzhou, China. His research interests include deep learning, multi-agent reinforcement learning, and Internet of vehicles.



Rongpeng Li (Member, IEEE) is currently an Associate Professor with the College of Information Science and Electronic Engineering, Zhejiang University, Hangzhou, China. From August 2015 to September 2016, he was a Research Engineer with the Wireless Communication Laboratory, Huawei Technologies Company, Ltd., Shanghai, China. From February 2020 to August 2020, he was a Visiting Scholar with the Department of Computer Science and Technology, University of Cambridge, Cambridge, U.K. His research interest focuses on networked intelligence for communications evolving. He was the recipient of the Wu Wenjun Artificial Intelligence Excellent Youth Award in 2021 and sponsorship, The National Postdoctoral Program for Innovative Talents in 2016. He is the Editor of *China Communications*.

Fei Wang received the doctor's degree from the University of science and Technology of China, Hefei, China. He is currently the Chief Researcher of Wireless Technology Lab, Huawei Technologies Co., Ltd., Shanghai, Chian. His research interests include 6G wireless network architecture, wireless distributed learning paradigm, federated learning, and large model.

Chenghui Peng is currently an Expert Researcher of Wireless Technology Lab, Huawei Technologies Co., Ltd., Shanghai, China. His research interests include 6G wireless network architecture, task-oriented native intelligent architecture, mobile computing network, and wireless distributed learning paradigm. He has applied for more than 100 patents in the LTE, 5G, and 6G field.

Authorized licensed use limited to: Zhejiang University. Downloaded on February 14,2024 at 05:44:41 UTC from IEEE Xplore. Restrictions apply.

Jianjun Wu is currently the Chief Researcher and Director of Future Network Laboratory of Huawei Technologies Co., Ltd., Shanghai, China. His research interests include future wireless network architecture include 6G network architecture definition, 5G E2E slicing solution research, standards, and industry development. He was the Director of the European Research Center Branch of Huawei 2012 Laboratories and led the local team to fully participate in the definition and research of 5G origins such as 5GIA and 5GPPP. He initiated and successfully established the 5GAA and 5GACIA industry alliances.

Zhifeng Zhao (Member, IEEE) received the B.E. degree in computer science, the M.E. degree in communication and information systems, and the Ph.D. degree in communication and information systems from the PLA University of Science and Technology, Nanjing, China, in 1996, 1999, and 2002, respectively. From 2002 to 2004, he was a Postdoctoral Researcher with Zhejiang University, Hangzhou, China, where his researches were focused on multimedia next-generation networks and softswitch technology for energy efficiency. From 2005 to 2006, he was a Senior Researcher with the PLA University of Science and Technology, where he performed research and development on advanced energy-efficient wireless router, *ad-hoc* network simulator, and cognitive mesh networking test-bed. From 2006 to 2019, he was an Associate Professor with the College of Information Science and Electronic Engineering, Zhejiang University, Hangzhou, China. He is currently the Chief Engineering Officer with the Zhejiang Lab, Hangzhou, China. His research interests include software defined networks, wireless network in 6G, computing networks, and collective intelligence. He is the Symposium Co-Chair of ChinaCom 2009 and 2010. He is the Technical Program Committee Co-Chair of the 10th IEEE International Symposium on Communication and Information Technology, 2010.

Honggang Zhang (Senior Member, IEEE) was an Honorary Visiting Professor with the University of York, York, U.K., and an International Chair Professor of excellence with the Université Européenne de Bretagne and Supélec, Rennes, France. He is currently the Chief Managing Editor of *Intelligent Computing, a Science Partner Journal*, and a Professor with the College of Information Science and Electronic Engineering, Zhejiang University, Hangzhou, China. He has coauthored and edited two books, *Cognitive Communications: Distributed Artificial Intelligence (DAI), Regulatory Policy & Economics, Implementation* (John Wiley & Sons) and *Green Communications: Theoretical Fundamentals, Algorithms and Applications* (CRC Press). His research interests include cognitive radio and networks, green communications, mobile computing, machine learning, artificial intelligence, and Internet of Intelligence. He was the co-recipient of the 2021 IEEE Communications Society Outstanding Paper Award and 2021 IEEE INTERNET OF THINGS JOURNAL Best Paper Award. He was the leading Guest Editor of the Special Issues on Green Communications of the *IEEE Communications Magazine*. He was the Series Editor of *IEEE Communications Magazine* (Green Communications and Computing Networks Series) from 2015 to 2018 and the Chair of Technical Committee on Cognitive Networks of IEEE Communications Society from 2011 to 2012. He is the Associate Editor-in-Chief of *China Communications*.

Authorized licensed use limited to: Zhejiang University. Downloaded on February 14,2024 at 05:44:41 UTC from IEEE Xplore. Restrictions apply.