Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 9329186, 10 pages
https://doi.org/10.1155/2022/9329186

Research Article

A Multiagent Cooperative Decision-Making Method for Adaptive Intersection Complexity Based on Hierarchical RL

Xiaojuan Wei,[1,2] Meng Jia,[3] and Mengke Geng[4]

[1] *Postdoctoral Station of Software Engineering, Zhengzhou University, Zhengzhou 450001, China*
[2] *Henan Province Fault-Tolerant Server Engineering Technology Research Center, Zhengzhou 450018, China*
[3] *Henan Polytechnic, Zhengzhou 450046, China*
[4] *Zhengzhou University of Aeronautics, Zhengzhou 450015, China*

Correspondence should be addressed to Xiaojuan Wei; 37093@hnzj.edu.cn

Received 11 August 2022; Revised 19 September 2022; Accepted 29 September 2022; Published 19 October 2022

Academic Editor: Muhammad Shafiq

In this paper, we propose a multiagent collaboration decision-making method for adaptive intersection complexity based on hierarchical reinforcement learning—H-CommNet, which uses a two-level structure for collaboration: the upper-level policy network fuses information from all agents and learns how to set a subtask for each agent, and the lower-level policy network relies on the local observation of the agent to control the action targets of the agents from each subtask in the upper layer. H-CommNet allows multiagents to complete collaboration on different time scales, and the scale is controllable. It also uses the computational intelligence of invehicle intelligence and edge nodes to achieve joint optimization of computing resources and communication resources. Through the simulation experiments in the intersection environment without traffic lights, the experimental results show that H-CommNet can achieve better results than baseline in different complexity scenarios when using as few resources as possible, and the scalability, flexibility, and control effects have been improved.

1. Introduction

The scheduling of traffic intersections is a crucial bottleneck to improve the efficiency of the whole road network, and the use of AI methods to study the scheduling of intersections is also a research focus in the field of vehicle networking. In current intersection control methods, some of them try to control vehicle traffic by learning traffic light strategies [1, 2], but such control methods are coarse-grained control with relatively limited strategy representation and no individualized traffic targets for each vehicle, so the traffic efficiency of the intersection cannot be maximized. In the vehicle self-organization control method that does not rely on traffic signals, each vehicle needs to be modeled as an agent and form a digital twin of the vehicle in the agent transportation system, which uses multiagent reinforcement learning methods to learn and make collaborative decisions of vehicles in the information space to obtain the optimal collaborative actions, and then, the actual actions are performed by the vehicle entities in the physical space [3, 4]. However, the current multiagent reinforcement learning synergy methods only focus on the action-level synergy, ignoring the continuity of the intent of the agent and the structured information of the synergistic task, which makes the synergy happen only at the finest granularity, and such a synergy approach lacks flexibility, and still maintains the synergy at the action level when facing a simple structured task, resulting in an enormous waste of computational resources. In addition, deploying the digitally twinned agent agents on different devices also yields two scheduling approaches: centralized scheduling and distributed scheduling. Among them, the former has a great delay, while the latter has a high demand for communication bandwidth. Therefore, these scheduling methods can not well meet the requirements of vehicle decision-making and ensure safe, efficient, and reliable driving.

In order to combine the advantages of distributed and centralized scheduling while addressing the lack of flexibility in collaborative granularity, we propose a multiagent collaboration decision-making method for adaptive intersection complexity based on hierarchical RL (reinforcement learning (RL))—H-CommNet. H-CommNet divides the collaborative decision-making task into two levels for implementation. The upper level of the policy network is required to achieve multiagent collaboration, but instead of directly generating actions of the agent, it generates subtasks that the agent needs to complete in the following period. We can achieve a different granularity of collaboration by controlling the time step of subtasks for the underlying policy to be guided. The lower layer policy network makes decisions based on the agent's own observations and is responsible for controlling the underlying actions of the agent to achieve the subtasks set by the upper layer. The task of the upper layer strategy is a collaborative task of multiple agents, and its goal is to maximize the cumulative reward of all the agents, so the strategy is updated using the cumulative reward of all the agents as the reward of the upper layer strategy. The lower layer strategy, on the other hand, is entirely a single-agent reinforcement learning task whose goal is to achieve the subtasks set by the upper layer and therefore uses an internal reward for subtask completion to update the lower layer strategy.

In order to realize the load balancing of computation, we place the upper policy network on the edge computing nodes for implementation and place the bottom policy network on each vehicle for implementation by onboard intelligence, so as to build a semicentralized and semidistributed vehicle-road cooperative scheduling system. In this system, the edge devices collect information from all vehicles in the region and make collaborative decisions before making decisions. After that, the decision results, i.e., the passage subtasks for each vehicle, are distributed to the respective vehicles. After receiving the set subtasks, the vehicles make decisions based on the single-vehicle agent for the next period of time until the subtasks are achieved. In this collaborative decision-making framework, both the decision-maker and the executor of vehicle actions are the vehicles themselves, and the vehicles can make decisions alone most of the time, thus significantly alleviating the hazards caused by decision latency. And each vehicle only needs to communicate with the edge device once in a period of time, which greatly reduces the requirement of communication bandwidth. Through a layered approach, we decompose the collaborative decision-making tasks and implement them on the edge computing nodes and invehicle agents, respectively, which also achieves the full utilization of computing resources.

In summary, a hierarchical collaborative approach is proposed in this paper, named as H-CommNet, and its main contributions can be summarized:

(I) A hierarchical multiagent reinforcement learning synergy method is proposed, which enables the synergy to occur at different granularity, thus improving the flexibility of the synergy and adaptability to different complexity scenarios

(II) A semicentralized and semidecentralized vehicle-road cooperative scheduling system is designed to achieve load balance between vehicle-mounted intelligence and edge intelligence in vehicle cooperative planning calculation, which reduces the delay of decision-making and improves the reliability of decision-making

(III) An asynchronous collaborative method based on request response is designed to solve the problem of inconsistent decision-making between upper and lower levels in the hierarchical collaborative decision-making framework of multiple vehicles. The hierarchical framework can work properly with the least communication and computing resources

The remainder of this paper is organized as follows: We give a review of related works in Section 2. In Section 3, we introduce the proposed method, H-CommNet. Section 4 describes our experimental setup, performance metrics, and experimental results. Finally, the conclusions are laid out in Section 5.

2. Related Works

With the development and extension of the Internet applications, Internet of Things (IoT) [5] gets great development; especially with the help of a new generation of computing models represented by fog computing [6], a large number of new models and new businesses have sprung up represented by smart transportation [7–9], smart medical care [10–14], smart education [15], smart agriculture [16], and industrial Internet [17–19]. 'Smart +' opens the door to us, and our lives have changed.

With the development of connected vehicles, it has become more important to use artificial intelligence techniques to study agent transportation. Among these works, the study of efficient scheduling of multiple vehicles at intersections is the focus of research. Some studies on intersection scheduling, such as [1, 2], improve the intersection efficiency by optimizing the traffic light intersection strategy. However, the traffic light-dependent scheduling strategy cannot achieve the characteristic scheduling of each vehicle and thus cannot maximize traffic efficiency. With the development of deep learning, especially, deep reinforcement learning has achieved excellent results in some scenarios that require multiagent collaboration, such as StarcraftII [20]. MARL method is also a suitable method for agent transportation systems, which can achieve self-organized collaboration of multiple vehicles by viewing each vehicle as an agent, thus improving the traffic system's efficiency. Among the current MARL methods, there are methods based on architectures with centralized training and distributed execution, such as QMIX [21], QTRAN [22], MADDPG [23], and COMA [24]. Such methods use global observations as a guide during centralized training and learn the optimal policy for each agent by optimizing the joint value function. There are also distributed decision methods that rely on communication for collaboration, such as DIAL [25], CommNet [26], and IC3Net [27]. Such approaches share information from the individual agent, such as observation, intent, and policy information, in a communicative manner among the agent to help each vehicle make globally collaborative actions. These methods all directly generate actions for each agent so that synergy occurs only at the action level. However, there are some simpler scenarios in collaborative tasks that allow the agent to collaborate well even if no or little collaboration occurs, and the decision granularity of these collaborative methods creates a significant waste of computational resources.

Hierarchical reinforcement learning was first proposed by Sutton et al. in [28], which formalized the formulation of hierarchical reinforcement learning through the introduction of a kind of temporal abstraction action OPTION and the definition of a semi-Markovian decision process. In the field of deep reinforcement learning, the combination of hierarchical reinforcement learning methods with deep neural networks has given rise to option-based methods such as option-critic methods [29, 30] and subtarget-based methods [31–33]. Since subtargets have clearer semantics, we therefore choose subtarget-based methods to build our multiagent hierarchical collaborative framework.

3. H-CommNet

This section presents our multiagent hierarchical collaborative algorithm, H-CommNet. As shown in Figure 1, our algorithm is divided into two levels, where the upper-level controller called "metacontroller" is responsible for making collaborative decisions and setting personalized subtasks for the lower-level agents. The lower-level controller called "controller" is responsible for controlling the underlying actions of the agent and implementing the subtargets set by the upper level. Here, the CommNet [26] algorithm is chosen to implement our upper layer collaborative network; thus, our method is called H-CommNet (hierachical CommNet). To make it easier for readers to understand, the symbol usage in the left part of Figure 1 are the same as in paper [26].

With the help of the hierarchical structure, H-CommNet decomposes the complex collaborative task horizontally and vertically, splitting the collaborative task $\{o_1, \cdots, o_J\}$ into multiple collaborative subtasks $\{g_1, \cdots, g_J\}$ at the upper level in the horizontal relationship while splitting the collaborative subtasks $\{g_1, \cdots, g_J\}$ into multiple single-agent reinforcement learning tasks $\{a_1, \cdots, a_J\}$ at the lower level in the vertical relationship. Such task decomposition makes the computational task lighter at a single level so that the desired strategies can be learned more easily. Also, since the upper layer only makes decisions at the subtask level, it increases the flexibility of decision granularity for collaborative decision-making, thus showing better flexibility in different scenarios. Next, we describe the upper-level controller and the lower-level controller in detail.

3.1. Metacontroller. The metacontroller (as in Figure 1) is implemented using the CommNet algorithm, which allows the fusion of messages during the forward propagation of the neural network, and through multiple rounds of fusion, an approximate perception of the global situation can be achieved and based on this perception, a collaborative decision is made to obtain the subtasks of each agent. Its strategy can be expressed as

$$\pi_\theta \left(o_1^t, o_2^t, \cdots, o_J^t\right) = \left(g_1^t, g_2^t, \cdots, g_J^t\right), \tag{1}$$

where o is the local observation of the agent and g is the subtarget set by each agent. The superscript t indicates the moment, and the subscript indicates the agent number. π_θ is the policy network of the metacontroller, and θ is the parameter of the network. The process of forwarding propagation at each layer can be expressed as

$$h_j^{i+1} = f^i\left(h_j^i, c_j^i\right), \tag{2}$$

$$c_j^{i+1} = \frac{1}{J-1}\sum_{j' \neq j} h_{j'}^{i+1}, \tag{3}$$

where f is the multilayer neural network module, i is the layer, j is agent, h is the hidden layer state of the network, c is the perceptual information after fusion, h_j^{i+1} is the out of layer $i+1$, J is a regularization factor, and the fused information is input to each node in the next layer, and the red arrows in Figure 1 indicate the flow of information after fusion.

The representation of subtasks relies on human experience, which often varies across tasks. The design of subtasks needs to satisfy the need to be able to provide sufficient guidance for the underlying actions and an adapted internal reward to describe the completion of the underlying policy for the subtasks. In the experimental scenario of Telematics, the vehicle passage subtask is designed as the location that the vehicle needs to reach in the next period.

The target of the upper layer is to learn the optimal policy network that maximizes the cumulative discounted reward expectation, so it updates its own policy using the sum of all rewards achieved by the agent over the duration of each subtarget as follows:

$$r_{\text{metacontroller}} = \sum_{i=1}^{n}\sum_{t=0}^{T_i} \gamma^t r_{i,t}, \tag{4}$$

where T_i is the time step for the duration of the ith agent subtarget.

The upper layer strategy provides a temporal decomposition for the overall task, decomposing the total task into subtasks that last for several time steps. Since the decomposition time scale is controllable, collaboration under a hierarchical structure can exhibit adaptiveness to tasks of different complexities: the upper layer provides guidance on larger time scales when the task is relatively simple and on smaller time scales when the task is simpler.

3.2. Controller. The controller (as in Figure 1) needs to rely on its own observations to complete the subtasks set by the upper layer, and at each moment, the controller's input has

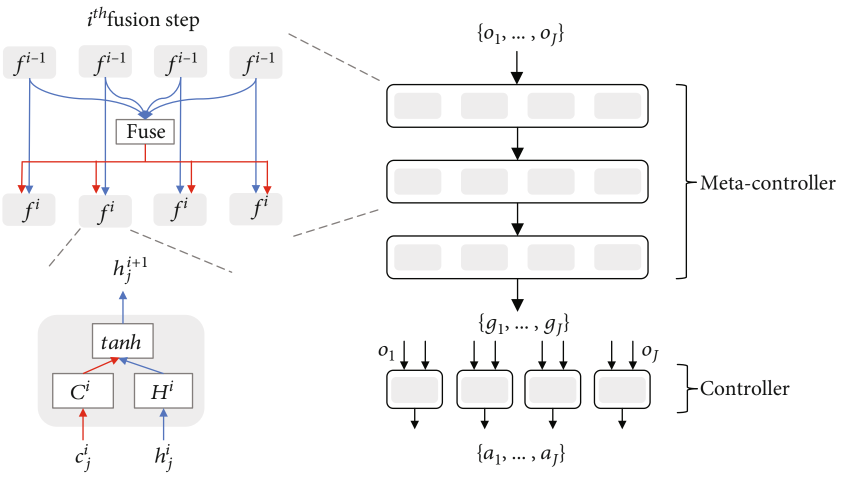


Figure 1: H-CommNet model: it is divided into two levels: metacontroller and controller.

local observations and subtasks for the current moment, and the output is the action of the agent. For the lower layer policy network, we choose the REINFORCE method for implementation. The controller can be represented formally as

$$\pi_\psi(o_{t+k}, g_t) = a_t, \tag{5}$$

where π_ψ is the controller's policy network and ψ is the network parameter. The subscript indicates the moment. Since each vehicle is a mutually equivalent node and the vehicles only need to complete the reinforcement learning task for a single agent, the agent can be viewed as homogeneous, and we use the technique of sharing parameters among the policy networks of the agent.

The controller's target is to complete the subtasks set by the upper level, and in order to provide the lower-level policy with a reward for parameter updates, we need to design an internal reward related to subtask completion:

$$r_{\text{controller}} = d(p_{t-1}, g) - d(p_t, g). \tag{6}$$

According to this reward, a positive reward is awarded when the vehicle approaches the target location, and a penalty is awarded when the vehicle moves away.

The underlying strategy provides a decomposition for subtasks at the scale of the agent. The subtasks that require collaboration are further decomposed into several single-agent reinforcement learning tasks that do not require collaboration. Since the underlying task is relatively simple, it is equivalent to adding only one underlying policy network when we increase the number of the agent in the environment, thus also improving the scalability of the collaborative task.

In multiagent collaborative tasks, the difficulty of the task changes continuously with the collaborative scenario and the number of the agent, and the adaptive and scalable nature of the hierarchical structure for different scenarios also makes it more adaptable to rapidly changing environments.

3.3. Extended Design. Encoding of observations using RNN, the input to the upper layer network is only the observation at the current moment, but the upper layer network does not make decisions at every moment, so there will be some moments when the state is ignored. To capture the temporal information of the observations, the observations of the agent can be processed first using recurrent neural networks, and then, the hidden state h of the agent is used as input to the collaborative network instead of the local observation o:

$$h_t^i = \text{GRU}\left(o_t^i, h_{t-1}^i\right). \tag{7}$$

Adding supporting information for collaborative decision-making, in some environments, there may be some auxiliary decision information in addition to the observation of each smart body, for example, the sensory information of the middle-side devices in the vehicle networking scenario can also help the vehicle to make better decisions. In order to add this auxiliary information to the collaborative decision network without affecting the structure of the network, we increase the number of nodes in each layer by 1 when fusing the information; however, when making action decisions, we still use only the number of nodes of each agent body.

3.4. Semicentralized and Semidistributed Vehicle-Road Cooperation System. As mentioned above, we deploy the upper-layer collaborative decision network to work in a centralized manner on the edge computing nodes and deploy the lower-layer policies to work in a distributed manner on the vehicle side, resulting in the system architecture shown in Figure 2. The vehicle encodes its own sensory information as a message vector to the edge device, and the edge device fuses the vehicle's messages to obtain an approximate global perception through the perception fusion module. Based on this perception, the decision node can develop a personalized passage subtarget for each vehicle and then distribute this subtarget to the vehicle through the communication channel, and the vehicle will rely on its own onboard agent

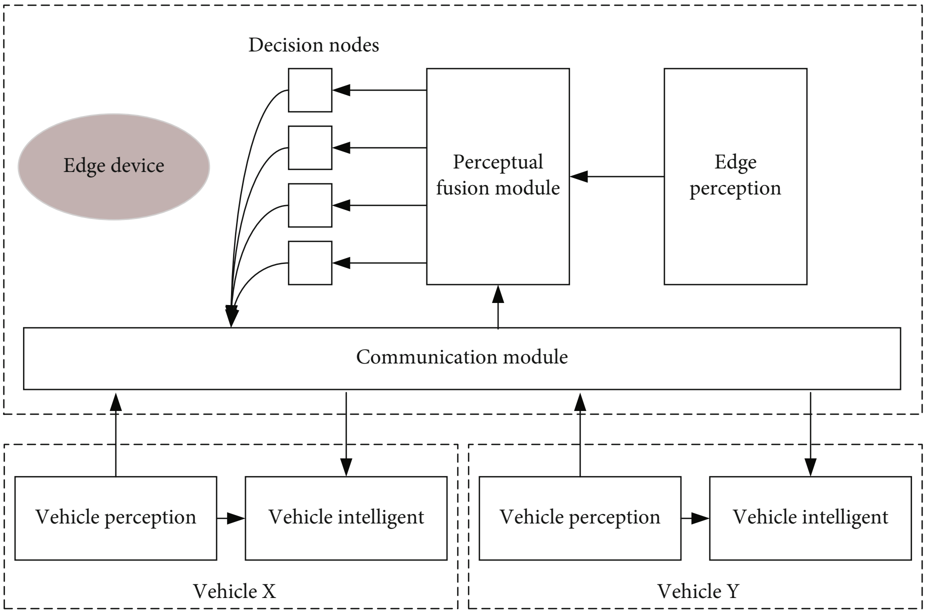


Figure 2: Vehicle-road cooperation system of H-CommNet: it consists of two layers, the upper-layer works on the edge device and the lower-layer works on the vehicle side.

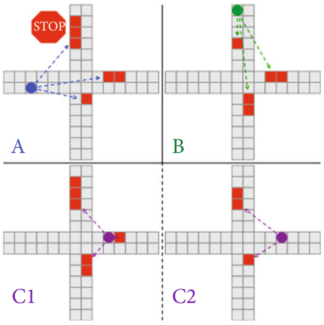


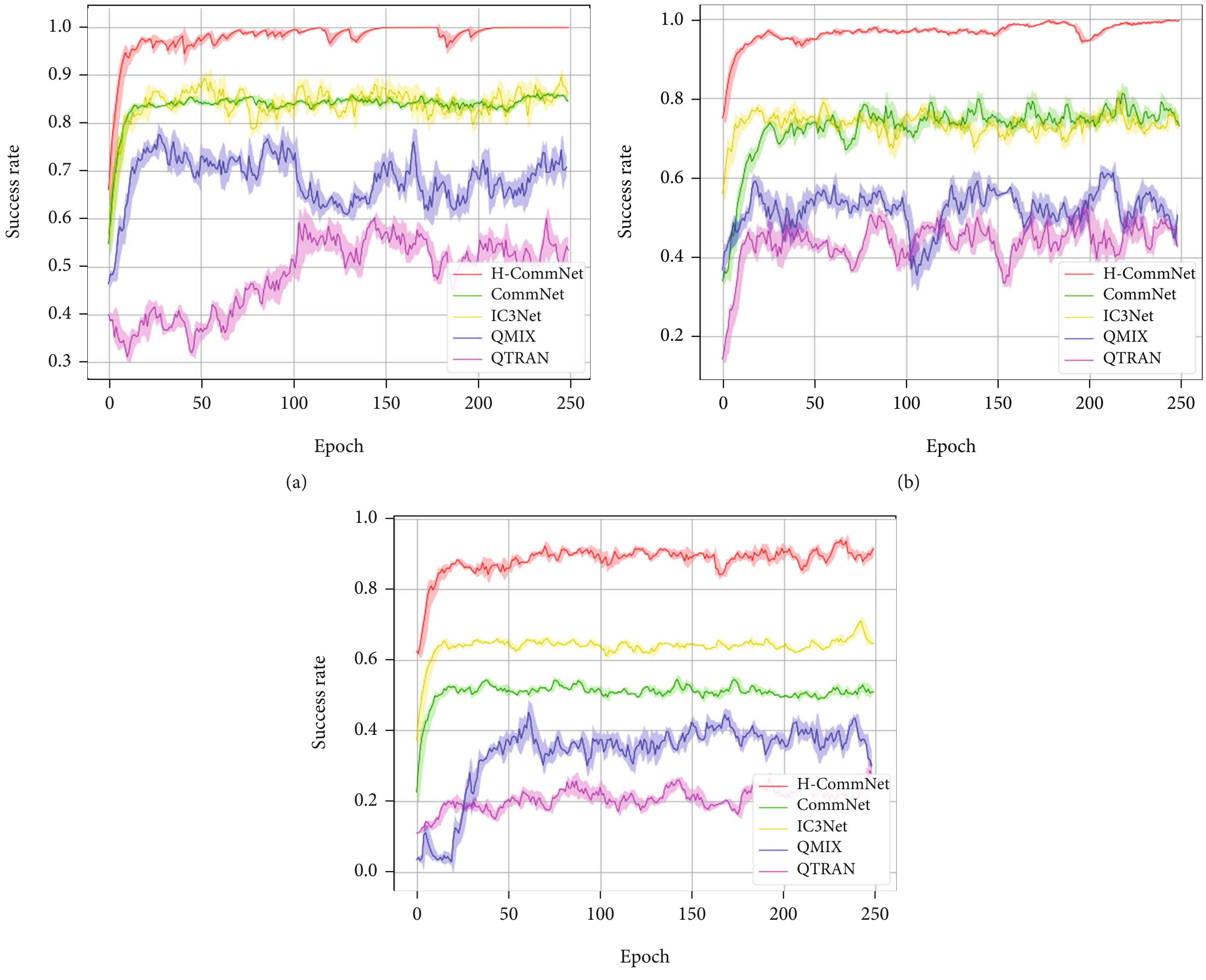
Figure 3: Multijunction without traffic signal light network environment.

and onboard perception for a single-vehicle navigation task after receiving this subtarget.

The hierarchical structure brings many advantages for multiagent collaboration; however, such a hierarchical structure also brings great challenges: differences in the time scales of upper and lower levels of decision-making can create problems of synchronization. Although upper-level decisions are required only when lower-level agent need subtarget guidance, each agent is different for the specific time step required to achieve the subtarget, which makes upper-level decisions required once every time an agent completes the current subtarget. How to design the corresponding synchronization pattern is crucial for the hierarchical structure. The simplest way is to fix the upper-level agent to also make decisions at each time step, generating subtasks for n agent. However, subtasks are not valid for every underlying agent, and we replace them with new subtasks only when the subtask of the current agent happens to be completed. If no agent completes its subtarget at the current moment, then the result of the upper-level decision at the current moment is nullified. Obviously, such an approach is extremely wasteful for both computational and communication resources, so we designed a request-response-based synchronization approach: If the agent completes its subtask at the current moment, it sends its own local observations to the agent as a scheduling request. After receiving the request, the edge device then communicates to collect the observation information of all the agents in the region to make a collaborative decision. However, such a request-response requires three rounds of communication. To reduce the communication delay, we eliminate the second round of communication and use historical observation data as the basis for decision-making. Specifically, when the edge device receives a request signal from a vehicle, it first updates the state of that vehicle in the information space, relying on the saved vehicle state rather than the communication-acquired vehicle state when making decisions. In order to avoid the vehicle's status not being updated for too long, we can set the minimum update time step so that every time the vehicle sends a request or reaches the maximum update time step it needs to communicate with the edge node once to update the vehicle's information. The complexity of communication is reduced from $o(n)$ to $o(1)$, and the system can realize the collaborative scheduling between vehicles with as few resources as possible.

4. Performance Evaluation

4.1. Experimental Scenario. Our experiments are conducted in a simulation environment of a traffic network without traffic signals containing several intersections, as shown in Figure 3. Each vehicle in this environment is modeled as an agent body that makes action decisions according to a multiagent reinforcement learning algorithm. At each time



(a)

(b)

(c)

Figure 4: (a) Easy map, vehicles can only go straight. (b) Medium map, vehicles can go straight and turn. (c) Hard map, includes multiple intersections and the road is two-way.

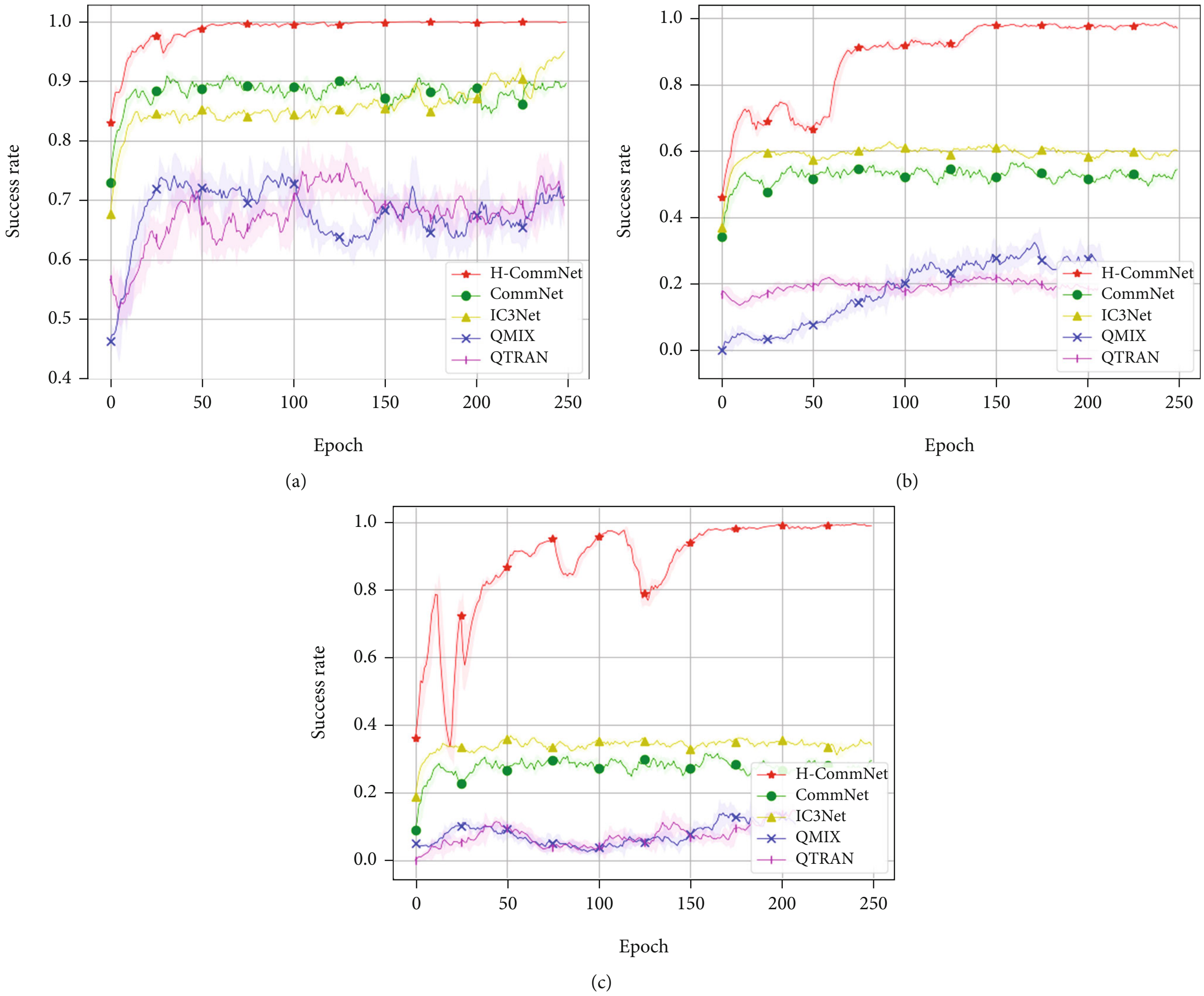
Table 1: The average value of the reward after convergence.

Methods	H-CommNet	CommNet	IC3Net	QMIX	QTRAN
Easy	-0.0028	-0.7839	-0.6026	-74.9135	-164.7118
Medium	-0.1130	-1.9675	-0.7237	-134.8274	-210.2246
Hard	-0.2903	-4.8989	-1.6031	-210.4081	-262.4223

Table 2: The average time of metacontroller provides guidance to controller.

Difficulty	Time
Easy	4.36
Medium	2.84
Hard	1.053

step, the simulation environment randomly generates an initial vehicle at the edge of the road with a certain probability, after which the vehicle will pass along a given route to reach its destination. The information of each vehicle is represented by a one-hot code (n, l, r), which indicates the vehicle number, the current location of the vehicle, and the code of its travel route, respectively. In addition to the information about the vehicle itself, the observed information about the vehicle at each moment includes the vehicle movements at the previous moment and the map information in the observation domain. The vehicle's action set consists of two actions, forward and stay. The goal of each vehicle is to try to avoid collision through the intersection to its destination, and the environment specifies that two vehicles are considered to have collided when their positions overlap. The simulation environment, therefore, sets a penalty of 10 for each collision of the vehicle, while to encourage the agent body to pass the intersection faster to avoid stopping, the environment also designs a cumulative penalty of 0.01 for the



(a)

(b)

(c)

Figure 5: (a) 0.1 increase rate of adding vehicles to the environment. (b) 0.3 increase rate of adding vehicles to the environment. (c) 0.5 increase rate of adding vehicles to the environment.

vehicle's stopping (0.01 for the first stopping, 0.02 for the second stopping...). Thus at moment t, the vehicle achieves a reward:

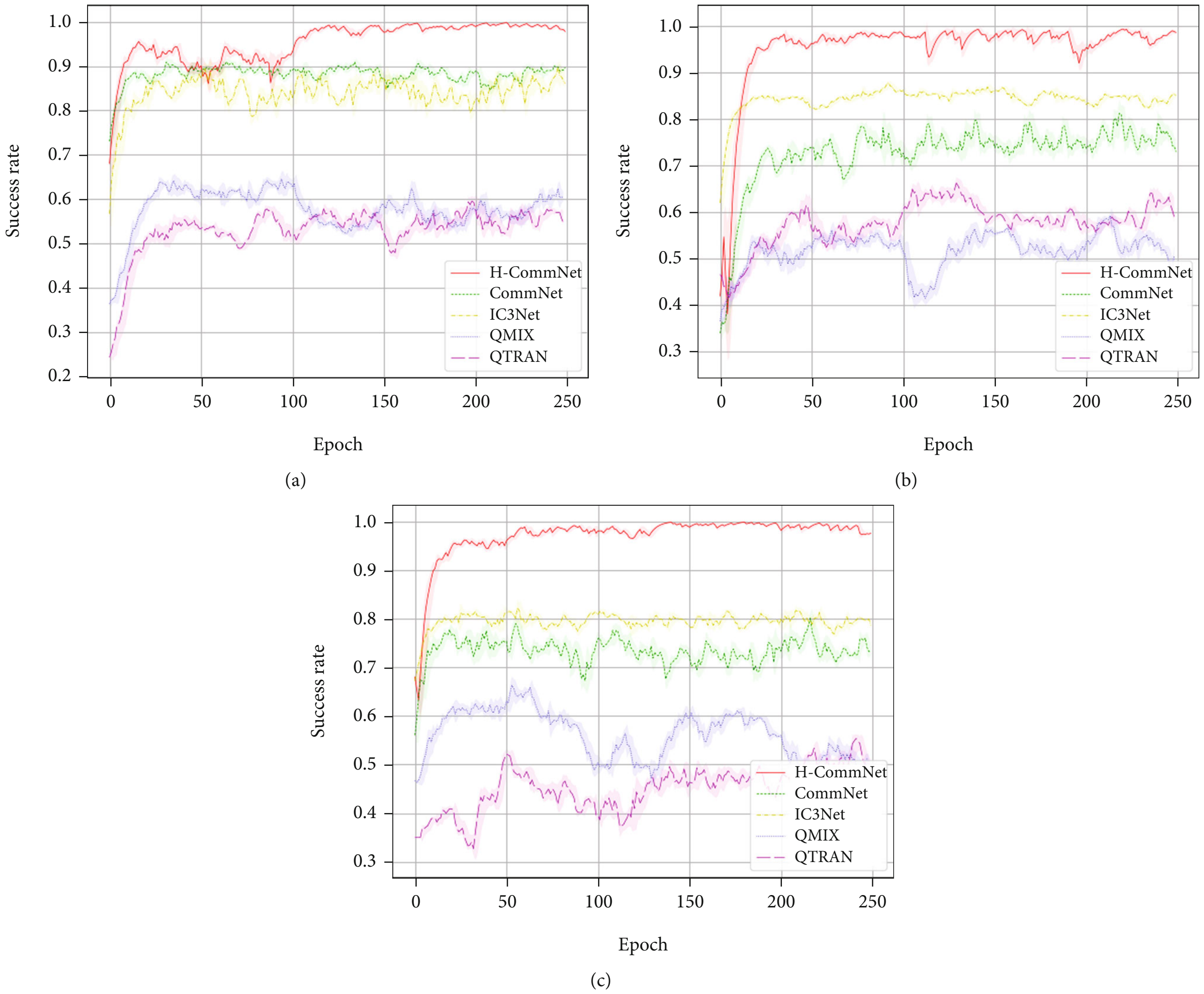
$$r(t) = C^{t} r_{\mathrm{coll}} +_{t} \tau r_{\mathrm{stay}}, \qquad (8)$$

where C indicates whether a collision has occurred and τ indicates the cumulative number of times the vehicle has stayed at this moment. After reaching the end, the vehicle is immediately removed from the environment, while collisions have no effect on our experiments except for the corresponding rewards, and the simulation continues after a collision occurs. The goal of our algorithm is to maximize the sum of the rewards for all vehicles.

In our evaluation metric, in addition to the cumulative average reward, we also include the success rate of each batch: if there is no collision between vehicles in an episode, then we consider the episode to be a success; otherwise, it is a failure. The percentage of successful episodes in a batch is counted as the success rate of a batch. This index is used as the basis for evaluating the method.

The hyperparameters related to the simulation environment in the experiment are as follows: ① the difficulty of the task, the simulation environment provides different shapes of maps, and the number of intersections contained in the map varies, thus providing the experiment with the different difficulty of the simulation map; ② the dimension of the map, i.e., the length and width of the map; ③ the probability of adding vehicles to the map at each moment, through which we can control the rate of change of the number of vehicles in the environment; and ④ the maximum number of vehicles, through which we can control the maximum number of vehicles in the environment and thus determine the density of agent bodies in the environment.

4.2. Comparison Methods. We compared the layered approach to some classic multiagent reinforcement learning



(a)

(b)

(c)

Figure 6: (a) The number of agents is 5. (b) The number of agents is 10. (c) The number of agents is 15.

methods, including CommNet, IC3Net, QMIX, and QTRAN.

CommNet is a classic multiagent reinforcement learning method that relies on communication for collaboration, IC3Net adds a gating mechanism on the basis of CommNet, and each agent will judge whether the observation is valuable for other vehicle decisions according to its own observations before communication so as to control whether the message is broadcast. Both QMIX and QTRAN are methods of centralized training distributed execution, and in centralized training, it is necessary to allow the algorithm to use the global state to guide the strategy of the agent, so we need to stitch the map information and vehicle information as a global state.

In the comparison tests, we conducted different ablation experiments for three aspects of performance:

(I) For the adaptation to scenarios of different complexities, we compared by adjusting the maps of different difficulties provided by the environment

(II) For the adaptation to different environmental change rates, we compared by adjusting the vehicle joining rate in the environment

(III) For the adaptation to different vehicle densities, we compare by adjusting the maximum number of vehicles in the environment

4.3. Experimental Results

4.3.1. Different Difficulty Scenarios. We compare the above algorithms in map scenes of different difficulties, Figures 4(a)–4(c) are the success rate curves of using different algorithms under maps of different difficulties. In an easy map, vehicles can only go straight, so you only need to avoid collisions where the vehicle behind you is chasing the vehicle in front of you. In medium map, the road becomes a two-way street, and the vehicle can turn, and the difficulty of the experiment increases significantly compared with the simple mode. Under the difficult road network environment, i.e., in hard map, the road is not only a two-way street but also contains a number of red light-free intersections. As you can see from the figures, regardless of the difficulty of the experiment, the layered method has advantages over other methods, and the advantages of stratification become more obvious as the difficulty increases.

Table 1 shows the average of the 50 epoch rewards after the algorithm converges for different difficulty environments. Judging from the situation of rewards, the layered method, and CommNet, IC3Net can basically avoid collisions, but H-CommNet has significantly improved its traffic efficiency, and its stay penalty is much smaller than CommNet and IC3Net. In contrast, the QMIX and QTRAN algorithms perform relatively poorly in such tasks with only local observations due to the need for centralized training, and the performance degrades fastest as the difficulty increases.

We also counted the average length of time that subtasks set by the upper layer provide guidance to the bottom layer under different experimental difficulties, as shown in Table 2. In simpler task scenarios, the upper layer tended to give longer step-by-step guidance, while when the task became complex, the upper layer tended to give single-step guidance. This shows that H-CommNet can adjust the granularity of its synergy for scenarios of different complexity and achieve the ability of complexity-adaptive synergy.

4.3.2. Rate of Change in Different Environments. In this ablation experiment, we vary the rate of adding a vehicle to the environment, thereby affecting the rate of change in the number of the agent in the environment. From Figures 5(a)–5(c), we can see that although the convergence process of the H-CommNet success rate curve is affected by the rapid change in the environment, the final convergence result can still achieve good results. The performance of CommNet and IC3Net degrades rapidly in the process of rapid environmental changes and gradually accelerates and can only reach one-third of the layered method; The performance of QMIX and QTRAN decreases to near 0. This experiment shows that the layered approach shows excellent adaptability to rapid changes in the environment compared to other approaches.

4.3.3. Different Vehicle Densities. In this set of experiments, we adjust the number of maximum agents in the environment and thus control the density of agents in the environment. From Figures 6(a)–6(c), it can be seen that the effect of increasing vehicle density on CommNet and IC3Net is small, with the success rate decreasing by less than 0.1; the effect on QMIX and QTRAN also exists, but the effect of vehicle density is not significant compared to the rate of environmental change. For the hierarchical method H-CommNet, the increase in vehicle density has no effect at all. This reflects the good scalability of the layered approach to the number of agents in collaboration, resulting in good adaptability to the density of agents.

5. Summary

Aiming at the problem that the same collaboration mode and granularity can not be applied to the intersection decision with different road attributes and it will cause the waste of computing resources, this paper proposes a multiagent collaboration decision-making method for adaptive intersection complexity based on hierarchical RL—H-CommNet.

H-CommNet is implemented in a hierarchical way. By using edge devices and onboard intelligence, the adaptive collaborative decision-making of multiple vehicles in different intersection scenarios is accomplished through task segmentation and task assignment. The experimental results show that the proposed method can not only improve the utilization of computing resources but also improve the collaborative granularity of decision-making. In the future research plan, we will dig deeply into the behavioral intention of vehicles to realize collaborative decision-making for the whole traffic network.

Data Availability

The data that support the findings of this study are available from the corresponding author upon reasonable request.

Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

Acknowledgments

This study was supported by the (1) Scientific and technological key project of Henan Province in 2022 (Project Number: 222102240049) and (2) Scientific research project of Henan Polytechnic in 2022 (Project Number: 2022ZK35).

References

[1] W. Wen, “A dynamic and automatic traffic light control expert system for solving the road congestion problem,” *Expert Systems with Applications*, vol. 34, no. 4, pp. 2370–2381, 2008.

[2] B. De Schutter and B. De Moor, “Optimal traffic light control for a single intersection,” in *Proceedings of the 1999 American Control Conference (Cat. No. 99CH36251)*, San Diego, CA, USA, June 1999.

[3] Z. Li, H. Yu, G. Zhang, S. Dong, and C. Z. Xu, “Network-wide traffic signal control optimization using a multi-agent deep reinforcement learning,” *Transportation Research Part C Emerging Technologies*, vol. 125, no. 3, p. 103059, 2021.

[4] Z. Wei, T. Peng, and S. Wei, “A robust adaptive traffic signal control algorithm using Q-learning under mixed traffic flow,” *Sustainability*, vol. 14, no. 10, p. 5751, 2022.

[5] A. A. Laghari, K. Wu, R. A. Laghari, M. Ali, and A. A. Khan, “A review and state of art of Internet of Things (IoT),” *Archives of Computational Methods in Engineering*, vol. 25, pp. 1–19, 2021.

[6] A. A. Laghari, A. K. Jumani, and R. A. Laghari, “Review and state of art of fog computing,” *Archives of Computational Methods in Engineering*, vol. 1, no. 5, pp. 3631–3643, 2021.

[7] M. Ibrar, J. Mi, S. Karim, A. A. Laghari, S. M. Shaikh, and V. Kumar, "Improvement of large-vehicle detection and monitoring on CPEC route," *Research*, vol. 9, no. 3, p. 45, 2018.

[8] M. Shafiq, Z. Tian, Y. Sun, X. du, and M. Guizani, "Selection of effective machine learning algorithm and bot-IoT attacks traffic identification for internet of things in smart city," *Future Generation Computer Systems*, vol. 107, pp. 433–442, 2020.

[9] M. Shafiq, Z. Tian, A. K. Bashir, A. Jolfaei, and X. Yu, "Data mining and machine learning methods for sustainable smart cities traffic classification: a survey," *Sustainable Cities and Society*, vol. 60, p. 102177, 2020.

[10] Z. Alansari, S. Soomro, M. R. Belgaum, and S. Shamshirband, "The rise of Internet of Things (IoT) in big healthcare data: review and open research issues," *Progress in Advanced Computing and Intelligent Engineering*, vol. 4, pp. 675–685, 2018.

[11] A. B. Tufail, I. Ullah, W. U. Khan et al., "Diagnosis of diabetic retinopathy through retinal fundus images and 3D convolutional neural networks with limited number of samples," *Wireless Communications and Mobile Computing*, vol. 2021, Article ID 6013448, 15 pages, 2021.

[12] A. B. Tufail, Y. K. Ma, Q. N. Zhang et al., "3D convolutional neural networks-based multiclass classification of Alzheimer's and Parkinson's diseases using PET and SPECT neuroimaging modalities," *Brain Informatics*, vol. 8, no. 1, pp. 1–9, 2021.

[13] S. Ahmad, T. Ullah, I. Ahmad et al., "A novel hybrid deep learning model for metastatic cancer detection," *Computational Intelligence and Neuroscience*, vol. 2022, Article ID 8141530, 14 pages, 2022.

[14] X. Wang, S. Yin, M. Shafiq et al., "A new V-Net convolutional neural network based on four-dimensional hyperchaotic system for medical image encryption," *Security and Communication Networks*, vol. 2022, Article ID 4260804, 14 pages, 2022.

[15] B. K. Yousafzai, S. A. Khan, T. Rahman et al., "Student-Performulator: student academic performance using hybrid deep neural network," *Sustainability*, vol. 13, no. 17, p. 9775, 2021.

[16] A. B. Tufail, I. Ullah, R. Khan et al., "Recognition of Ziziphus lotus through aerial imaging and deep transfer learning approach," *Mobile Information Systems*, vol. 2021, Article ID 4310321, 10 pages, 2021.

[17] M. Shafiq, Z. Tian, A. K. Bashir, X. Du, and M. Guizani, "IoT malicious traffic identification using wrapper-based feature selection mechanisms," *Computers & Security*, vol. 94, article 101863, 2020.

[18] S. Karim, Y. Zhang, A. A. Laghari, and M. R. Asif, "Image processing based proposed drone for detecting and controlling street crimes," in *2017 IEEE 17th International Conference on Communication Technology (ICCT)*, pp. 1725–1730, Chengdu, China, October 2017.

[19] L. Wang, Y. Shoulin, H. Alyami et al., "A novel deep learning-based single shot multibox detector model for object detection in optical remote sensing images," ***Geoscience Data Journal***, 2022.

[20] A. Tampuu, T. Matiisen, D. Kodelja et al., "Multiagent cooperation and competition with deep reinforcement learning," ***PLOS ONE***, vol. 12, no. 4, article e0172395, 2017.

[21] T. Rashid, M. Samvelyan, C. Schroeder, G. Farquhar, J. Foerster, and S. Whiteson, "QMIX: monotonic value function factorisation for deep multi-agent reinforcement learning," *International conference on machine learning*, vol. 80, no. 3, pp. 4295–4304, 2018.

[22] K. Son, D. Kim, W. J. Kang, D. E. Hostallero, and Y. Yi, "QTRAN: learning to factorize with transformation for cooperative multi-agent reinforcement learning," *International conference on machine learning*, vol. 97, pp. 5887–5896, 2019.

[23] R. Lowe, Y. I. Wu, A. Tamar, J. Harb, O. Pieter Abbeel, and I. Mordatch, "Multiagent actor-critic for mixed cooperative-competitive environments," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[24] J. Foerster, G. Farquhar, T. Afouras, N. Nardelli, and S. Whiteson, "Counterfactual multi-agent policy gradients," ***Proceedings of the AAAI Conference on Artificial Intelligence***, vol. 32, no. 1, 2018.

[25] J. Foerster, I. A. Assael, N. De Freitas, and S. Whiteson, "Learning to communicate with deep multi-agent reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 29, 2016.

[26] S. Sukhbaatar and R. Fergus, "Learning multiagent communication with backpropagation," *Advances in Neural Information Processing Systems*, vol. 29, 2016.

[27] A. Singh, T. Jain, and S. Sukhbaatar, "Learning when to communicate at scale in multiagent cooperative and competitive tasks," 2018, https://arxiv.org/abs/1812.09755.

[28] R. S. Sutton, D. Precup, and S. Singh, "Between MDPs and semi-MDPs: a framework for temporal abstraction in reinforcement learning," *Artificial Intelligence*, vol. 112, no. 1-2, pp. 181–211, 1999.

[29] P. L. Bacon, J. Harb, and D. Precup, "The option-critic architecture," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, p. 1, 2017 February.

[30] R. Chunduru and D. Precup, "Attention option-c6ritic," 2022, https://arxiv.org/abs/2201.02628.

[31] A. S. Vezhnevets, S. Osindero, T. Schaul et al., "Feudal networks for hierarchical reinforcement learning," in *International Conference on Machine Learning*, pp. 3540–3549, Sydney, NSW, Australia, August 2017.

[32] C. Liu, F. Zhu, Q. Liu, and Y. Fu, "Hierarchical reinforcement learning with automatic sub-goal identification," *IEEE/CAA Journal of, Automatica Sinica*, vol. 8, no. 10, pp. 1686–1696, 2021.

[33] J. Erskine and C. Lehnert, "Developing cooperative policies for multi-stage reinforcement learning tasks," ***IEEE Robotics and Automation Letters***, vol. 7, no. 3, pp. 6590–6597, 2022.