A Decentralized Communication Framework based on Dual-Level Recurrence for Multi-Agent Reinforcement Learning

Xuesi Li, Jingchen Li, Haobin Shi*, and Kao-Shing Hwang, *Senior Member, IEEE*

***Abstract*—Designing communication channels for multi-agent is a feasible method to conduct decentralized learning, especially in partially observable environments or large-scale multi-agent systems. In this work, a communication model with dual-level recurrence is developed to provide a more efficient communication mechanism for the multi-agent reinforcement learning field. The communications are conducted by a gated-attention-based recurrent network, in which the historical states are taken into account and regarded as the second-level recurrence. We separate communication messages from memories in the recurrent model so that the proposed communication flow can adapt changeable communication objects in the case of limited communication, and the communication results are fair to every agent. We provide a sufficient discussion about our method in both partially observable and fully observable environments. The results of several experiments suggest our method outperforms the existing decentralized communication frameworks and the corresponding centralized training method.**

***Index Terms*—multi-agent system, multi-agent reinforcement learning, gated recurrent network.**

I. Introduction

THE emerging multi-agent reinforcement learning techniques are widely used in various scenarios, such as multi-UAV formation [1], traffic signal control [2], and engineering system management [3]. Multi-agent reinforcement learning has aroused intense scholarly interest, especially in the fields of robotics [4], [5] and cyber-physical systems [6]. In some works, multi-agent reinforcement learning is used to solve large-scale decision-making tasks. These works broke a complicated decision-making model down into several sub-decision-making processes, using multi-agent reinforcement learning algorithms to optimize the policies for every process [7]. With more advanced optimization and deep learning techniques [8], multi-agent reinforcement learning allows agents to collaborate with each other and respond to the environment timely, which the traditional machine learning techniques and heuristic methods lack [9]. All such works suggest that multi-agent reinforcement learning has been an important and widely applicable paradigm [10].

The main challenge in multi-agent reinforcement learning is that the environment becomes unstable for every agent due to the existence of its peers [11]. For large-scale multi-agent systems, this challenge can be ignored because the foremost goal approximation and stability [12]. But for a general multi-agent scenario, the incomplete Markov decision process (MDP) leads to misconvergence for each agent in the case of independent learning, while the joint learning is difficult due to the curse of dimensionality [13]. The main direction for multi-agent reinforcement learning is centralized learning because decentralized multi-agent reinforcement learning lacks the couplings among the behaviors of different agents. For designing a more lightweight centralized learning model, some researchers proposed a centralized training and decentralized execution (CTDE) mechanism to train multi-agent system [14]. This mechanism allows an agent to make decisions according to its own observation or perception, while its peers' behaviors should be considered when evaluating its policy. The CTDE mechanism is always based on an actor-critic framework, separating the execution from the evaluation. However, centralized learning is not suitable for distributed control [15], especially in large-scale or partially observable multi-agent environments. Communication is mainstream in the case of decentralized training. Lots of researchers have attempted to build communication channels for multi-agent systems: Sukhbaatar et al. [16] utilized an additional neural network module to realize continuous communication for multiple agents, aiming at fully cooperative scenarios; Peng et al. [17] leveraged bidirectional-recurrent network to share latent states for agents, and validated their model on StarCraft testbed; Jiang et al. [18] introduced attention mechanism into multi-agent communication, in which the gated attention module is used to judge whether an agent should communicate with others. As a feasible way to share information among agents, communication can enhance policy coordination and make agents gain more decision bases in partially observable environments.

Different from traditional multi-agent control, the communication for multi-agent (deep) reinforcement learning occurs on the latent space transformed by neural networks. The existing communication modules can be divided into two types: state integration module and recurrent model-based module. The former involves centralized communication, using a centralized network to combine the states for all agents [19]. For example, Targeted Multi-Agent Communication architecture (TarMAC) [20] leveraged a soft attention module to conduct a targeted communication behavior, by which every two agents are assigned a communication channel. The latter regards

X. Li, J. Li and H. Shi are with the School of Computer Science and Engineering of Northwestern Polytechnical University, Xi'an, Shaanxi 710129 China.

X. Li and J. Li contributed equally to this work.

K.-S. Hwang is with National Sun Yat-Sen University, Kaohsiung, Taiwan.

Corresponding author: H. Shi (email: shihaobin@nwpu.edu.cn).

the agent group as a sequence, using recurrent models to process the latent state for the agent sequence. Deep distributed recurrent Q-networks (DDRQN) [21] is an earlier recurrent model-based method aiming to solve communication-based coordination tasks without any pre-designed communication protocol. Then several improved models are developed, such as CommNet (Communication Net) [16] and BicNet (Bidirectionally-Coordinated Net) [17]. Some works also utilized attention mechanisms to enhance recurrent model-based methods. For example, Attentional and Recurrent Message Integration (ARMI) calculated the correlation between the message and the observation by attention mechanism. At the same time, Liu et al. [22] leveraged a self-attention mechanism to build a communication framework, which can learn both to construct communication groups and decide when to communicate for agents.

The existing communication mechanisms still have limitations. State integration modules result in a massive communication network for a too-large multi-agent system [23]. Although the observations can be transformed into smaller latent states, the joint latent state space rises exponentially as the number of agents increases. Moreover, the state integration modules cannot be used in the case of limited communication due to the centralized communication module. Some work built state integration modules between every two agents. However, when training a large-scale heterogeneous multi-agent system [24], the built communication modules cannot share parameters with each other, still leading to an enormous training structure. As for recurrent model-based communications, the order of agents in the communication sequence may be a hidden danger. Although some works have introduced bidirectional recurrent models into the communication modules, the agents at the two ends of the communication sequence are still hard to send messages. Using an gated-attention-based recurrent model [25] may be a feasible method, but no work has been devoted to introducing attention-based recurrent models into multi-agent communication.

In a partially observable environment, researchers always assume that agents have the ability to share observation information with each other by the proposed communication modules. In a single-agent reinforcement learning environment, an agent needs to combine its observation in the time series to make decisions. Although an agent can handle its partial observations by other mechanisms, the price is vast computation and a too-long training process [26]. In a multi-agent system with a partially observable environment, the task is always modeled as a decentralized partially observable Markov decision process (Dec-POMDP). The followed problem is how to transfer the historical observations to an agent. This problem had been a largely under-explored domain. However, from the experience of single-agent reinforcement learning, we know that historical observations are also necessary for agents in partially observable environments.

In this work, we propose a dual-recurrent communication model (2ReCom) for multi-agent reinforcement learning. We consider fully cooperative multi-agent systems, and the agents can be both homogeneous and heterogeneous. The proposal is a circular recurrent communication module. In our method, agents share messages by the first-level recurrence, while the historical observations are taken into account by the second-level recurrence. Compared with bidirectional recurrent models, our model is fair to all agents in the communication sequence. In 2ReCom, we separate communication messages from memories for every agent so that agents can adapt changeable communication objects in the case of limited communication. Moreover, we proposed a new attention-based recurrent network that can avoid missing important messages in the communication process when acting on a large-scale multi-agent system [10]. With the developed 2ReCom, agents can combine both the current and historical messages, by which the entire environment can be precepted more efficiently.

The main contributions of this work are summarized as follows:

- We discuss the challenges in multi-agent communication, proposing a dual-recurrent communication model to enable multi-agent deal changeable communication objects.
- To reduce the impacts of communication information on individual memories of agents, We propose an attention-based recurrent network to conduct communication in our model. Benefiting from the attention mechanism, the proposed recurrent model can guarantee the communication qualities.
- We conduct experiments in both partially observable and fully observable environments, and provide sufficient discussions to analyze the experimental results. Furthermore, we execute an ablation experiment to validate our method.

The rest of the paper is organized as follows: the background is given in the second section. In the third section, we present the dual-recurrent communication model, while the proposed attention-based recurrent network is described. Next, in the fourth section, we present and discuss the results of our experiments. Finally, we conclude in the fifth section and give directions for future research.

II. Background

In recent years, multi-agent communication has been a hotspot in the multi-agent reinforcement learning community [27]. Although CTDE mechanisms without communication achieved satisfactory results on simulated scenarios [28], the industrial community desires distributed learning to deploy policies in low-power edge servers. For this reason, decentralized learning becomes mainstream for real-world tasks [29]. Researchers have attempted to achieve observation sharing among agents by lightweight communication modules, developing various communication frameworks for multi-agent systems, such as multiple vehicles [30] and industrial system management [31]. In those works, only cooperative multi-agent systems are considered because the consistent goal for agents is the premise of high-quality communications [32].

In partially observable environments, multi-agent reinforcement learning models are always regarded as Dec-POMDP models [33]. That is, every agent is allowed to make decisions according to its own state (observation/perception). Dec-POMDP guarantees the learning will not be impacted by

This article has been accepted for publication in IEEE Transactions on Cognitive and Developmental Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCDS.2023.3281878

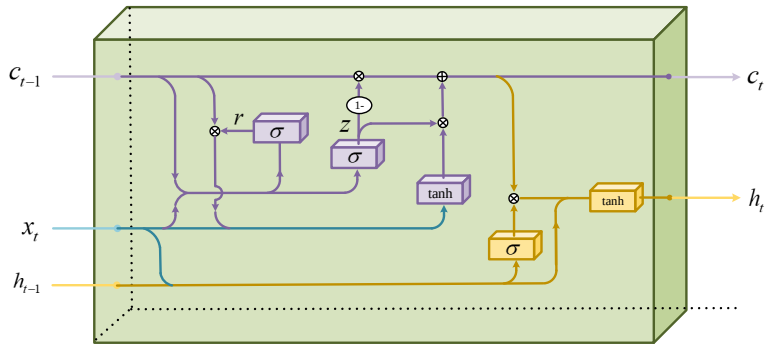


Fig. 1. The structure of the attention-based recurrent network.

too-large joint state space and action space. Based on Dec-POMDP, many works developed communication frameworks in various ways. However, fewer of them can be all-purpose for different scenarios. The existing works more or less have shortcomings. TarMAC (Targeted Multi-Agent Communication) [20] leveraged GRU [34] to combine the current observation for an agent and the previous observations for others, but the current observations for the peers are neglected. BicNET (Bidirectionally-Coordinated Nets) [17] built communication channels through a bidirectional-recurrent network. However, an agent still cannot interact with the other far from it in the communication sequence. As for other communication frameworks, such as Comment [16] and ATOC [18], these shortcomings are also not overcome. Moreover, when working in a fully observable environment, the existing communication frameworks fall behind the corresponding CTDE-based methods that have no communication [28].

In our opinion, there are three requirements for an excellent multi-agent communication framework: In a partially observable environment, both the current and previous observation of other agents should be considered as elements for the communication, which is beneficial to build entire perception for the environment; The learned joint policy should not be impacted by the communication sequence in the communication framework, in other words, the framework is fair to every agent, which is also the premise when working for a heterogeneous multi-agent system; in the case of limited communication, agents need to adapt changeable communication objects. Considering these three requirements, we conduct this work, developing a more efficient communication framework for multi-agent systems.

III. The Proposed Method

In this section, we present a dual-recurrent communication framework (2ReCom). First, we describe the attention-based recurrent network used in 2ReCom. Then the entire communication framework is given, and we discuss the application of 2ReCom in the case of limited communication. Finally, we discuss the module sharing in both homogeneous and heterogeneous multi-agent systems.

A. Gated Recurrent Model for Communication

The entire structure of the proposed attention-based recurrent network is shown in Fig. 1. In this section, we just describe the design for the recurrent network. The reason for this design will be presented later after the communication framework is given.

As shown in Fig. 1, there are three inputs for our attention-based recurrent network: the cell state c_{t-1}, the current input x_t, and the communication message h_{t-1}. At time t, the previous cell state and communication message are fed together with the current input. Because we hope the cell state can remember both the long-term memory and the short-term dependence, we use GRU as the prototype, while the output is separated from the cell state by an additional attention module. Similar to GRU, the cell state is updated by the current input. First, a reset gate processes c_{t-1} and x_t, calculating the reset information r:

$$r = \text{Sigmoid}(W_r \cdot [c_{t-1}, x_t]), \tag{1}$$

where W_r denotes the transformation matrix in the reset gate. Then the update gate is used to change the cell state. Let

$$z = \text{Sigmoid}(W_z \cdot [c_{t-1}, x_t]), \tag{2}$$

where W_z is the transformation matrix in the update gate. z determines which data should be forgotten in the cell state and which information in the input should be remembered. The new data that should be stored by the cell state is calculated as

$$\widetilde{c} = \tanh(W_u \cdot [r \cdot c_{t-1}, x_t]), \tag{3}$$

where W_u is the transformation matrix for the remember. After that, a new cell state is generated:

$$c_t = (1 - z) \cdot c_{t-1} + \widetilde{c}. \tag{4}$$

It should be noticed that we do not regard the new cell state as the final output of the recurrent model. The additional attention gate, which is the brown part in Fig. 1, separating the output from the cell state. The attention value is generated through a transformation matrix W_a fed by x_t and h_{t-1}:

$$a = \text{Softmax}(W_a \cdot [x_t, h_{t-1}]). \tag{5}$$

The generated attention value acts on the new cell state. That is, the attention gate determines which data is important in the

cell state. We assume the communication message h_{t-1} and cell state x_t are in the same feature space, and the attention gate reflects the position for the important data. We will explain why not feed the new cell state with W_a later.

The final output is calculated by output gate W_o:

$$h_t = \tanh(W_o \cdot [a \cdot c_t, h_{t-1}]). \tag{6}$$

Because our recurrent network is used to share messages among agents, the output should be the combination of both cell state and communication message. Different from general recurrent models, h_t has independence from c_t, but makes connections with c_t through x_t.

From the perspective of the gating mechanism, gates transit no information from the original data, generating a set of weights or masks to control information transition. In our recurrent module, the cell state c_{t-1} receives new memories from input x_t through the update gate, so that there is a direct interaction from x_t to c_t. However, the communication message at the previous time step has no forward information transmission. That is, h_t will not affect the memory c_t. On the contrary, the attention gate calculates weights for the output, filtrating the information from the new cell state c_t and improving the output h_t. Under this configuration, the cell state will not be affected by the communication. This is important for our main purpose, that is, the changeable communication object will not decrease the stability of the memory.

B. Dual-Recurrent Communication

In this section, we present our 2ReCom framework detailedly. In a multi-agent environment with N agents, we use o_t^i to denote the observation of the i-th agent at time t. As shown in Fig. 2, we assign every agent with a state encoding module, an attention-based recurrent network mentioned before, and a policy network. In our 2ReCom framework, agents share messages through a dual-recurrent model. The state encoding modules transform observations and output hidden states, by which the observations can be mapped into the same feature space. $f(\cdot;\theta_i^r)$ denotes the recurrent model for the i-th agent.

Fig. 2 (a) shows the communication flow for the first agent. After hidden state s_t^1 is calculated, the recurrent model for the first agent updates its cell state and generate a temporary communication vector:

$$c_t^1, \hat{h}_t^1 = f(c_{t-1}^1, s_t^1, h_{t-1}^1; \theta_1^r), \tag{7}$$

where c_t^i is the cell state for the i-th agent at time t, and h_t^i denotes the final communication vector for the i-th agent. In this process, the new cell state for the first agent c_t^1 is retained and used at the next time step $t+1$. Then the temporary communication vector $\hat{h}_t^1$ is transmitted to $f(\cdot;\theta_2^r)$, updating itself by c_{t-1}^2 and s_t^2:

$$c_t^2, \hat{h}_t^1 = f(c_{t-1}^2, s_t^2, \hat{h}_t^1; \theta_2^r). \tag{8}$$

It should be noticed that in this process, the generated c_t^2 is not retained. The cell state for the i-th agent is just updated in the i-th agent's communication flow. $\hat{h}_t^1$ is updated continually through all other agents' recurrent model in this way until the last agent's recurrent model output the final communication feature vector for the first agent:

$$h_t^1 = g_t^N(g_t^{N-1}(\cdots(g_t^1(h_{t-1}^1))\cdots)), \tag{9}$$

where $g_t^i(x) = f(c_{t-1}^i, s_t^i, x; \theta_i^r) \setminus c_t^i$.

For the i-th agent, its communication flow is shown in Fig. 2 (b). First, a temporary communication vector $\hat{h}_t^i$ is output by $f(\cdot;\theta_i^r)$, while the new cell state c_t^i is retained for the communication at the next time step. Then $\hat{h}_t^i$ is updated continually through the recurrent models of latter agents. After $f(\cdot;\theta_N^r)$ updates $\hat{h}_t^i$, the temporary communication vector is transmitted to $f(\cdot;\theta_1^r)$. Till $\hat{h}_t^i$ is transmitted to $f(\cdot;\theta_{i-1}^r)$, the final communication vector is generated:

$$h_t^i = g_t^{i-1}(g_t^{i-2}(\cdots g_t^1(g_t^N(g_t^{N-1}(\cdots g_t^i(h_{t-1}^i)\cdots)))\cdots)). \tag{10}$$

In a partially observable environment or large-scale multi-agent system [35], the limited communication ability should be considered. An agent may have no communication channel with the peers that cannot be observed by it in a partially observable environment. In a large-scale multi-agent system, communicating with all peers is needless for an agent. So we discuss the application of the developed 2ReCom in the case of limited communication. Because the communication vector is separated from the cell states in the communication flow, 2ReCom can adapt to changeable communication objects. In a partially observable environment, an agent can just communicate with the peers that can be observed by it. Let N_i denote the number of the peers observed by the i-th agent, the communication flow for the i-th agent is:

$$h_t^i = g_t^{j_{N_i}}(g_t^{j_{N_{i-1}}}(\cdots g_t^i(h_{t-1}^i)\cdots)), (j_1, \cdots, j_{N_i}) \in \widetilde{i}. \tag{11}$$

where $\widetilde{i}$ is the set of the observed agents. In a large-scale multi-agent environment, researchers always leveraged approximation to simplify the interaction among agents [36], [37]. When using 2ReCom to train a large-scale multi-agent system, $\widetilde{i}$ can be regarded as the neighbor agents for the i-th agent.

Due to the separation of the communication results and cell states, the change of communication objects has no negative effect on the communication results. Even if a new agent becomes a neighbor or observable peer for the i-th agent, its cell state contains just the memory of its own historical hidden states, which have no relevance to its historical communications.

C. Interpretation for 2ReCom

There are two recurrent processes in the developed 2ReCom framework. In the first level recurrence, the hidden states of all agents are regarded as a sequence. For each agent, its own recurrent model first generates a temporary communication vector, and then all other recurrent models update this vector in turn. In this process, the communication vector gains messages from every agents' memory (cell state) by the brown part in Fig. 1. In the second-level recurrence, the hidden state sequence for each agent is integrated into the cell state. In our attention-based recurrent model, the communication vector

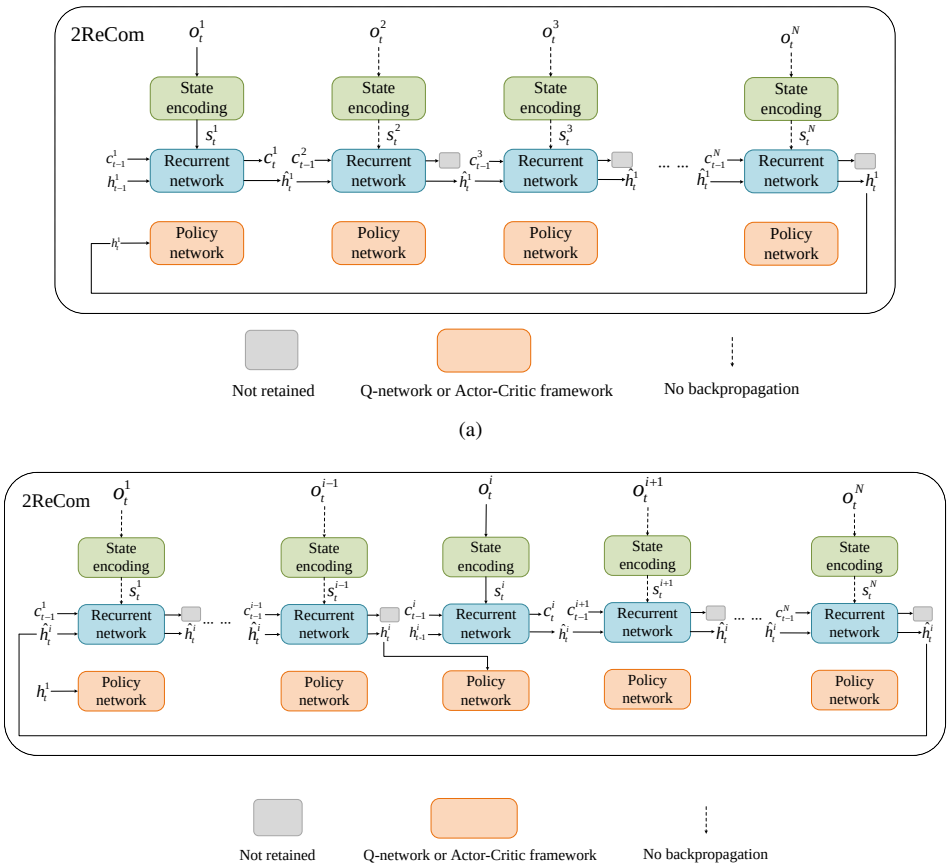


Fig. 2. The 2ReCom framework for N agents. (a) is the communication flow for the first agent, while (b) is the communication flow for the i-th agent.

is separated from the cell state, so that the cell states for agents are independent of each other. The cell state contains both long-term memory and short-term dependency instead of dividing them. That is why GRU rather than LSTM is used as the prototype. Moreover, the historical communication vectors of an agent also form a sequence that remains the communication results, which is necessary for the agent to perceive the entire environment.

The attention-based recurrent model is specially developed for the 2ReCom framework. In our recurrent model, the new cell state is calculated by just the previous cell state and hidden state. In other words, the communication vector does not participate in this process. In the communication flow, just (temporary) communication vector and hidden state co-determine which information in the cell state should be integrated into the communication vector. Due to the new cell state has combined the hidden state and the previous cell state, so that the communication result is calculated just by itself and the attentive new cell state.

For different agents, although their communication sequences are different, each of them needs to receive messages from others or the observed (neighbor) agents. Unlike state integration frameworks, 2ReCom needs no too-large module to share information in a centralized way. Compared with other recurrent model-based communication frameworks, 2ReCom allows agents to have similar communication flows. Although in different communication flows, there are slight distinctions among the communication sequences, we leverage an attention mechanism to compress the messages, by which these distinctions can be overlooked.

At time t, after all agents select their actions and get rewards, an experience tuple $< O_t, O_{t+1}, C_{t-1}, C_t, H_{t-1}, H_t, A_t, R_t >$ is stored in replay buffer, where $O_t = (o_t^1, o_t^2, \ldots, o_t^N)$ is the observations for all agents, $C_t = (c_t^1, c_t^2, \ldots, c_t^N)$ denotes the cell states, $H_t = (h_t^1, h_t^2, \ldots, h_t^N)$ is the final communication vectors at time t, $A_t = (a_t^1, a_t^2, \ldots, a_t^N)$ is the joint actions at time t, and $R_t = (r_t^1, r_t^2, \ldots, r_t^N)$ is the rewards vector given by the environment. It should be noticed that in the update process for an agent, all other agents' hidden states do not back-propagate gradients, as shown in Fig. 2. In a communication flow, the communication vector needs to gain messages through hidden states for all agents. However, there is no coupling between the state encoding modules for other agents and the policy for the current agent, so that the state encoding module for an agent is updated together with just the policy network for it.

D. Further Discussion

The proposed 2ReCom satisfies the three requirements mentioned in Section 2. The dual-recurrent framework can not only share current messages for all agents but also provide

a feasible way to combine the historical states for further communication. Moreover, the 2ReCom framework is fair to every agent. The communication sequences for all agents are similar, so that the sequence of agents will not impact the communication results, and we will validate this in the experiment section. In fully observable environments, the proposed 2ReCom makes agents get enough communication and combine the historical state sequences since the 2ReCom framework can rival the corresponding CTDE-based model that uses a centralized critic-network to evaluate policies.

Module sharing is a common method to simplify the multi-agent reinforcement learning model. Some state integration models built a shared module for all agents, by which the complexity of the model can be reduced. In CTDE-based methods, researchers also attempted to simplify models in this way. For example, in Actor-Attention-Critic, all agents use the same attention module. For 2ReCom, module sharing can also be used. In a homogeneous multi-agent system, all agents have the same state space and action space, so that a state encoding module can be used for all agents. However, the same policy network may damage the diversity of policies, resulting in a locally optimal solution. As for the communication module, the premise for sharing recurrent models is that all hidden states should be in the same feature space. Hence, in a homogeneous multi-agent system, the communication module should be consistent with the state encoding module. If we use a shared state encoding module, the communication module is also shared for agents. In a heterogeneous multi-agent system, it is no doubt that each agent is assigned with a state encoding module. In this case, the communication module cannot be shared. Because the action spaces for heterogeneous agents are also different, the policy networks also cannot share parameters.

IV. Experiment

In this section, we first conduct experiments in partially observable environments to compare our 2ReCom with several baseline methods. Then we investigate the performances of 2ReCom in a fully observable environment. Finally, an ablation experiment is used to validate our opinions. The Multi-Agent Particle Environment is used as the experimental platform. In these experiments, the policy network in 2ReCom and the baseline methods take DDPG as the prototype. The source code of our experiments can be accessed on https://github.com/LiJingchen1212/2ReCom.git.

A. Baseline Methods

Five algorithms are used as the baseline methods: ATOC, BicNet, CommNet, MADDPG, DDPG. The first three algorithms are communication-based methods, and MADDPG is the corresponding centralized learning model, while the last one is the independent learning method.

ATOC (Attentional communication model) [18] designs an attention unit to receive hidden states and action intention for each agent. An agent determines whether to communicate with other agents according to the attention unit. ATOC leverages a bidirectional LSTM unit as the communication channel.

BicNet (Bidirectionally-coordinated net) [17] uses a bidirectional RNN as the communication channel, allowing agents to share latent states. BicNet provides a vectorized extension for the actor-critic formulation, and it also introduces module sharing to solve the scalability issue.

CommNet (Communication Neural Net) [16] uses continuous communication to coordinate multi-agent system. CommNet is the typical work to replace manually specified communication protocol with a deep feed-forward network.

MADDPG (Multi-agent deep deterministic policy gradient) [14] proposed CTDE mechanism to train multi-agent system in a centralized way, in which the actor-network is decentralized, and the critic network is centralized. Because this work uses DDPG as the reinforcement learning model for 2ReCom, MADDPG can be regarded as the corresponding CTDE method.

DDPG (Deep deterministic policy gradient) is a decentralized learning method, and each agent in DDPG updates its policy independently.

B. Experiments Settings

In these experiments, the learning rate is 0.001, the discounted factor is set to 0.99, and the batch size is 1024. The observation for an agent includes its position, its velocity, the 3 nearest peers, and the 3 nearest landmarks in Cooperative Navigation or the prey in Predator Prey.

The state encoding module is a linear layer with 64 nodes, and a leaky ReLU is followed for non-linear activation. The output of the attention-based recurrent model is also 64-dimensional. As for the policy network, the actor-network is a linear layer. The critic-network first encodes the action and messages to two 64-dimensional tensors with leaky ReLU functions, and then the two tensors are concatenated and fed to a linear layer.

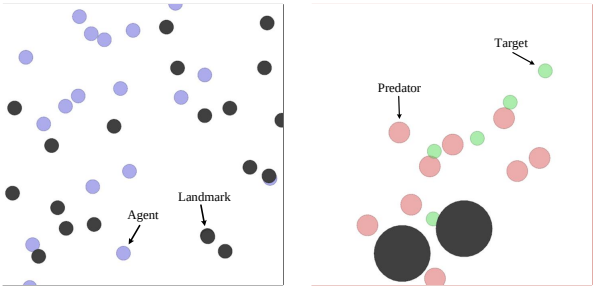
C. Partially Observable Environment

1) Scenarios: There are two scenarios in the test on partially observable environments: Cooperative Navigation and Predator Prey. Both of them are in a two-dimensional world with continuous space and discrete, and we modify them to partially observable environments. These experiments is used to investigate the performance of 2ReCom in the case of limited communication.

Cooperative Navigation scenario has 20 agents and 20 landmarks. As shown in Fig. 3 (a), every agent needs to reach a landmark as soon as possible while colliding is not allowed. In this scenario, each agent has its independent reward. The reward is negatively correlated with the relative distance between the agent and the nearest landmark, and an agent will deserve punishment if it collides with another. At each time step, an agent can observe just 3 nearest agents and 3 nearest landmarks, so that the agents need to communicate with each other to precept the entire environment. In this experiment, module sharing does not exist in any module.

Predator Prey scenario has 10 predators and 5 targets. As shown in Fig. 3 (b), the predators need to catch the targets while colliding is not allowed. The speed of targets is twice

This article has been accepted for publication in IEEE Transactions on Cognitive and Developmental Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCDS.2023.3281878



(a) Cooperative Navigation (b) Predator Prey

Fig. 3. The two scenarios used in test on partially observable environments.

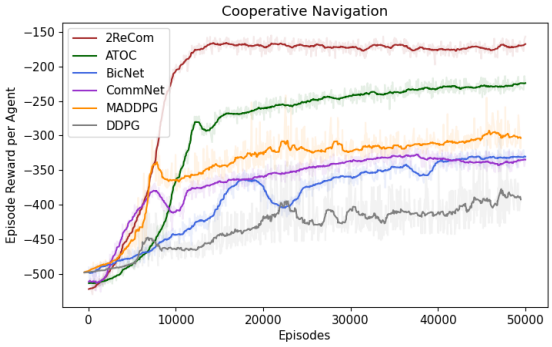


Fig. 4. The results on Cooperative Navigation, where the curves denote the evaluation rewards and translucent areas represent the training rewards.

that of predators, so that predators need to cooperate with each other. In this scenario, we pre-trained the policies for the targets, while our 2ReCom and the baseline methods are used to train the predators. The policies for targets are not updated in the learning processes. At each time step, a predator can observe just 3 nearest peers and 3 nearest targets, so that predators need to communicate with each other for more effective hunting. In this experiment, module sharing exists in the state encoding module and communication module, and all agents have a shared reward.

2) Results and Analyses: As shown in Fig. 4, our 2ReCom achieves the best result on Cooperative Navigation. In a partially observable environment, CTDE-based methods use joint observation and action to evaluate policies, so that the coupling among agents can be captured easily. However, the too-large joint state space and action space result in ineffective learning. The episode reward on MADDPG is just -300, which is far smaller than ATOC and our 2ReCom. ATOC leverages an attention mechanism to conduct communication, by which agents can get suitable communication objects in large-scale environments. The developed 2ReCom adapts changeable communication objects by separating communication messages from cell states, so that 2ReCom can also achieve efficient communication. Moreover, our 2ReCom can combine the communication objects' historical observations by the two-level recurrent model, which is important to build an entire perception in partially observable environments, so that our 2ReCom outperforms ATOC. The other two communication frameworks, BicNet and CommNet, are far behind our 2ReCom because the communication channels in them lack the control of communication objects. DDPG gets the worst performance. It is no doubt that independent learning cannot handle large-scale multi-agent reinforcement learning.

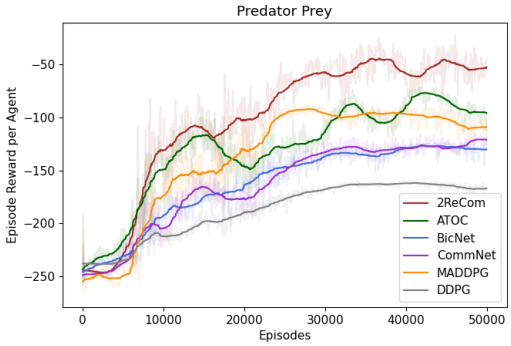


Fig. 5. The results on Predator Prey, where the curves denote the evaluation rewards and translucent areas represent the training rewards.

Fig. 5 shows the results on the Predator Prey scenario. The results are consistent with those on Cooperative Navigation: Our 2ReCom outperforms all baseline methods, while other communication frameworks (expect ATOC) fall behind the corresponding CTDE-based method (MADDPG). Because this task has a shared reward for all agents, agents have to learn more advanced cooperation through communication. In our 2ReCom, agents can get messages from the historical states of others, so that the cooperation can be learned more quickly. In this experiment, all agents share the same state encoding module and communication module, and the results suggest that the module sharing can be used for 2ReCom. With the two experiments, the superiority of 2ReCom in the case of limited communication is validated convincingly.

The complete experiment results on partially observable environments are given in Table I, from which we know that our 2ReCom outperforms baseline methods in the two scenarios. To show the superiors of 2ReCom more convincingly, we use confidence intervals and t-tests to validate the agent rewards of the 6 control groups (2ReCom and baselines). Table I shows the confidence intervals of agent rewards. With the confidence set to 95%, we find that the proposed 2ReCom provides the fairest policies for the agents. The confidence interval of 2ReCom in Cooperative Negation is [-170.52,-170.48], while that in Predator Prey is [-51.46,-51.34]. Executing independent samples t-test for the rewards per agent of 2ReCom and each baseline method. Because they have unequal variances, unequal variances t-tests are conducted. The t-test results are given in Table II. In the case of the significance level $\alpha = 0.05$, the t values are far less than the critical values, which further validates the improvement of our 2ReCom. Compared with other recurrent model-based communication frameworks, our 2ReCom is fair to all agents due to the clever communication flow. As mentioned before, although the bidirectional recurrent model proves that every agent can receive messages from each other, the communication sequence still results in different communication conditions. In 2ReCom, the communication

Scenario	Cooperative Negation			Predator Prey		
	Reward per Agent	STD	Confidence Interval (95%)	Reward per Agent	STD	Confidence Interval (95%)
2ReCom	**-170.5**	±0.04	[-170.52,-170.48]	**-51.4**	±0.11	[-51.46,-51.34]
ATOC	-228.7	±0.07	[-228.73, -228.67]	-92.3	±0.48	[-92.58,-92.02]
BicNet	-334.2	±1.29	[-334.81,-333.59]	-130.7	±1.62	[-131.64,-129.76]
CommNet	-339.0	±0.97	[-339.45,-338.55]	-124.4	±1.36	[-339.79,-336.21]
MADDPG	-303.1	±0.04	[-303.12, -303.08]	-112.5	±0.35	[-112.70,-112.29]
DDPG	-378.8	±8.44	[-382.74,-374.86]	-168.7	±11.43	[-175.31,-162.09]

TABLE I
THE EXPERIMENT RESULTS ON PARTIALLY OBSERVABLE ENVIRONMENTS

Scenario	Cooperative Negation		Predator Prey	
	t value	degrees of freedom	t value	degrees of freedom
ATOC	3228.36	30.21	262.64	9.94
BicNet	567.24	19.04	154.44	9.08
CommNet	776.20	19.06	169.19	9.12
MADDPG	10482.95	38.00	526.65	10.76
DDPG	110.37	19.00	32.45	9.00

TABLE II
THE t-TEST RESULTS OF 2RECOM AND EACH BASELINE METHOD ON THE TWO SCENARIOS.

Agent	1	2	3	4	5	6	7	8	9	10
1	-	3	0	17	12	9	0	11	19	4
2	4	-	7	1	9	6	21	8	0	19
3	1	7	-	15	11	2	9	14	2	14
4	15	0	14	8	7	0	2	13	5	11
5	16	6	9	13	-	1	4	16	0	10
6	11	4	0	2	0	-	19	13	17	9
7	0	22	11	1	5	18	-	6	6	2
8	12	5	14	9	12	17	3	-	3	0
9	17	0	2	4	1	19	8	5	-	19
10	5	16	11	7	8	12	6	0	10	-

TABLE III
THE COMMUNICATION COUNT ON PREDATOR PREY.

flows for all agents are similar, and the changeable communication flows guarantee that the uncertain communication objects can not impact the communication qualities. That is why our 2ReCom is fair to all agents.

However, a latent risk is that agents may form several groups spontaneously to avoid communicating with all others. To investigate this phenomenon, we count the communication objects for every agent, judging whether agents can communicate with all peers by changeable communication flows. The result is given in Table III, in which the line i column j is the number of occurrences of the j-th agent within the communication flow for the i-th agent. From that, we can know each agent can communicate with most peers instead of fixed communication objects. It should be noticed that the communication flows for agents are not symmetrical. For example, the second agent appeared in the communication flow for the first agent three times, but the first agent just appeared four times in the communication flow for the second agent. This is because an agent just communicates with the three nearest peers. If an agent is surrounded by many peers, just three of the peers will appear in its communication flow while it may appear in several peers' communication flows. Our 2ReCom allows agents to adapt changeable communication objects, and this result suggests that agents do not form fixed groups to avoid the change of communication objects.

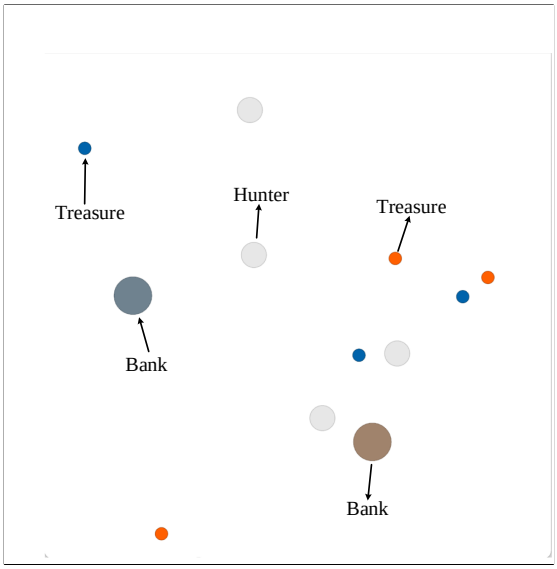


Fig. 6. The scenario used in the test on fully observable environment.

D. Fully Observable Environment

1) Scenario: In this experiment, we compare our 2ReCom with baseline methods on the Cooperative Treasure Collection scenario [28]. This scenario has two types of agents: hunters and banks. As shown in Fig. 6, the 6 hunters need to collect treasures and deposit the treasures with the corresponding bank agents. In this scenario, colliding among hunters is not allowable, and all agents can observe the position of each other. We set an individual reward for every agent instead of a shared reward. Compared with the former two scenarios, this scenario requires lower-level cooperation. Because there are just 8 agents in Cooperative Treasure Collection, all agents communicate with each other when trained by 2ReCom. That is, the communication flow for every agent is fixed, and all other agents appear in it.

2) Results and Analyses: The results on Cooperative Treasure Collection are given in Fig. 7. Our 2ReCom is the

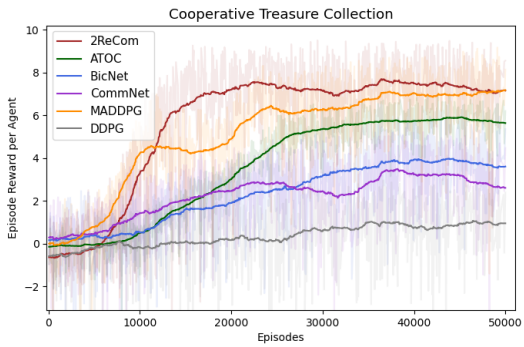


Fig. 7. The results on Cooperative Treasure Collection, where the curves denote the evaluation rewards and translucent areas represent the training rewards.

agent sequence	sequence 1	sequence 2	sequence 3	sequence 4
2ReCom	7.319	7.326	7.292	7.391
BicNet	3.977	3.016	3.928	3.444

TABLE IV
THE RESULTS ON DIFFERENT AGENT SEQUENCES

only decentralized communication method that outperforms MADDPG. The mean episode reward for 2ReCom reaches 7.3, while those for other communication frameworks are less than 6.0. Although the state-of-the-art centralized method [28] got a larger score in this scenario, this experiment still suggests the developed 2ReCom is better than other decentralized communication methods in fully observable environments, even rivals the corresponding centralized learning model (MADDPG). In this experiment, an agent communicates with all others at every time step. The fixed communication flows make us investigate whether the sequence of agents impacts the communication results in 2ReCom. In a heterogeneous multi-agent system, the perceptions of agents are different, so that the risk of totally different communication results may exist when the sequence of agents is changed. To validate that, we conduct this experiment several times with different agent sequences to investigate whether 2ReCom gets different results. The results are shown in Table IV. On all the four control groups, the results for our 2ReCom are about 7.3, which suggests that the efficiency of our 2ReCom is not impacted by the agent sequence in the case of heterogeneous multi-agent systems. Conversely, when the agent sequence changes, BicNet gets a very different result. In our opinion, the communication messages extracted by gated recurrent models are more effective than those extracted by RNN. The attention-based recurrent model specially designed by us strengthens this superiority.

E. Ablation Experiment

In the previous experiments, we have validated that our 2ReCom is fair to all agents, and the communication results are not impacted by the agent sequences. In this section, we design an ablation experiment to validate the separation of communication messages from cell states. In the control group, the attention module acts on the cell state directly, and the new cell state is used as the communication result or temporary communication vector. The control group is used to train multi-agent systems in all three scenarios, and the final results are given in Table V.

	2ReCom	Control Group
Cooperative Navigation	-170.5	-213.7
Predator Prey	-51.7	-80.4
Cooperative Treasure Collection	7.331	6.934

TABLE V
THE RESULTS OF THE ABLATION EXPERIMENT

From the comparison between 2ReCom and the control group, we know that the separation of communication messages from cell states is necessary for 2ReCom. In partially observable environments, the separation guarantees the current communication for an agent will not be affected by previous communications. Although the previous communication results may be useful for the current decision-making, the changeable communication object makes an agent hard to understand the previous communication messages. In the case of a fully observable environment, the effect is less than that in partially observable environments because the communication flows for agents are fixed in the fully observable environment. This ablation experiment validates that separating communication from memory plays an important role in our dual-recurrent communication framework, especially in the case of changeable communication flows.

V. CONCLUSION

In this work, we develop a new communication framework for decentralized multi-agent reinforcement learning. Our 2ReCom has two main superiorities: We regard historical states of agents as a part of communication information, proposing a dual-recurrence for decentralized multi-agent systems; the developed 2ReCom separates communications from memories, making agents adapt to changeable communication objects. We analyze applications of the proposed 2ReCom in the case of different multi-agent systems, and a sufficient discussion about module sharing is provided. Compared with other communication frameworks, our 2ReCom is fair to all agents, and the agent sequence makes no impact on the communication results. The experiments on both partially and fully observable environments proved that our 2ReCom is better than the existing communication frameworks and the corresponding centralized learning method.

In fact, decentralized learning for multi-agent systems still faces many challenges. The policy evaluation for decentralized learning requires steady communication results. Centralized learning can achieve that by back-propagation directly, but decentralized learning needs an efficient communication framework. In future work, we will improve 2ReCom step by step, aiming at rivaling the start-of-the-art centralized learning method.

ACKNOWLEDGMENT

This work is supported by Major research Project of National Natural Science Foundation of China under Grant 92267110, National Natural Science Foundation of China

under Grant 61976178 and 62076202, Open Research Projects of Zhejiang Lab (NO.2022NB0AB07), Shaanxi Province Key Research and Development Program of China under Grant 2022GY-090 and 2023-YBGY-354, CAAI-Huawei MindSpore Open Fund (NO.CAAIXSJLJJ-2021-041A), and the Doctor's Scientific Research and Innovation Foundation of Northwestern Polytechnical University (CX2022016).

REFERENCES

[1] A. Shamsoshoara, M. Khaledi, F. Afghah, A. Razi, and J. Ashdown, "Distributed cooperative spectrum sharing in UAV networks using multi-agent reinforcement learning," in *2019 16th IEEE Annual Consumer Communications & Networking Conference (CCNC)*. IEEE, 2019, pp. 1–6.

[2] H. Zhang, S. Feng, C. Liu, Y. Ding, Y. Zhu, Z. Zhou, W. Zhang, Y. Yu, H. Jin, and Z. Li, "Cityflow: A multi-agent reinforcement learning environment for large scale city traffic scenario," in *The World Wide Web Conference*, 2019, pp. 3620–3624.

[3] C. Andriotis and K. Papakonstantinou, "Managing engineering systems with large state and action spaces through deep reinforcement learning," *Reliability Engineering & System Safety*, vol. 191, p. 106483, 2019.

[4] Y. Duan, B. X. Cui, and X. H. Xu, "A multi-agent reinforcement learning approach to robot soccer," *Artificial Intelligence Review*, vol. 38, no. 3, pp. 193–211, 2012.

[5] H. Shi, L. Shi, M. Xu, and K.-S. Hwang, "End-to-end navigation strategy with deep reinforcement learning for mobile robots," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 4, pp. 2393–2402, 2019.

[6] S. Wang, J. Wan, D. Zhang, D. Li, and C. Zhang, "Towards smart factory for industry 4.0: a self-organized multi-agent system with big data based feedback and coordination," *Computer Networks*, vol. 101, pp. 158–168, 2016.

[7] X. Liu, J. Yu, Z. Feng, and Y. Gao, "Multi-agent reinforcement learning for resource allocation in IoT networks with edge computing," *China Communications*, vol. 17, no. 9, pp. 220–236, 2020.

[8] J. Li, H. Shi, and K.-S. Hwang, "An explainable ensemble feedforward method with Gaussian convolutional filter," *Knowledge-Based Systems*, p. 107103, 2021.

[9] H. J. Bae and P. Koumoutsakos, "Scientific multi-agent reinforcement learning for wall-models of turbulent flows," *Nature Communications*, vol. 13, no. 1, p. 1443, 2022.

[10] J. Li, H. Shi, and K.-S. Hwang, "Using fuzzy logic to learn abstract policies in large-scale multiagent reinforcement learning," *IEEE Transactions on Fuzzy Systems*, vol. 30, no. 12, pp. 5211–5224, 2022.

[11] K. Zhang, Z. Yang, and T. Başar, "Multi-agent reinforcement learning: A selective overview of theories and algorithms," *Handbook of Reinforcement Learning and Control*, pp. 321–384, 2021.

[12] Y. Liu, X. Liu, Y. Jing, H. Wang, and X. Li, "Annular domain finite-time connective control for large-scale systems with expanding construction," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 51, no. 10, pp. 6159–6169, 2020.

[13] L. Buşoniu, R. Babuška, and B. De Schutter, "Multi-agent reinforcement learning: An overview," *Innovations in multi-agent systems and applications-1*, pp. 183–221, 2010.

[14] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 2017, pp. 6382–6393.

[15] S. Chen, D. W. C. Ho, L. Li, and M. Liu, "Fault-tolerant consensus of multi-agent system with distributed adaptive protocol," *IEEE Transactions on Cybernetics*, vol. 45, no. 10, pp. 2142–2155, 2015.

[16] S. Sukhbaatar, A. Szlam, and R. Fergus, "Learning multiagent communication with backpropagation," in *Proceedings of the 30th International Conference on Neural Information Processing Systems*, 2016, pp. 2252–2260.

[17] P. Peng, Y. Wen, Y. Yang, Q. Yuan, Z. Tang, H. Long, and J. Wang, "Multiagent bidirectionally-coordinated nets: Emergence of human-level coordination in learning to play starcraft combat games," *arXiv preprint arXiv:1703.10069*, 2017.

[18] J. Jiang and Z. Lu, "Learning attentional communication for multi-agent cooperation," in *Advances in Neural Information Processing Systems*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds., vol. 31. Curran Associates, Inc., 2018.

[19] W. Mao, K. Zhang, E. Miehling, and T. Başar, "Information state embedding in partially observable cooperative multi-agent reinforcement learning," in *2020 59th IEEE Conference on Decision and Control (CDC)*. IEEE, 2020, pp. 6124–6131.

[20] A. Das, T. Gervet, J. Romoff, D. Batra, D. Parikh, M. Rabbat, and J. Pineau, "TarMAC: Targeted multi-agent communication," in *Proceedings of the 36th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97. PMLR, 09–15 Jun 2019, pp. 1538–1546.

[21] J. N. Foerster, Y. M. Assael, N. de Freitas, and S. Whiteson, "Learning to communicate to solve riddles with deep distributed recurrent q-networks," *arXiv preprint arXiv:1602.02672*, 2016.

[22] Y.-C. Liu, J. Tian, N. Glaser, and Z. Kira, "When2com: Multi-agent perception via communication graph grouping," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020, pp. 4106–4115.

[23] H. Shi, J. Li, J. Mao, and K.-S. Hwang, "Lateral transfer learning for multiagent reinforcement learning," *IEEE Transactions on Cybernetics*, vol. 53, no. 3, pp. 1699–1711, 2023.

[24] S. Zuo, Y. Song, F. L. Lewis, and A. Davoudi, "Output containment control of linear heterogeneous multi-agent systems using internal model principle," *IEEE Transactions on Cybernetics*, vol. 47, no. 8, pp. 2099–2109, 2017.

[25] B. Wang, K. Liu, and J. Zhao, "Inner attention based recurrent neural networks for answer selection," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2016, pp. 1288–1297.

[26] J. Li, K. Kuang, B. Wang, F. Liu, L. Chen, C. Fan, F. Wu, and J. Xiao, "Deconfounded value decomposition for multi-agent reinforcement learning," in *International Conference on Machine Learning*. PMLR, 2022, pp. 12 843–12 856.

[27] W. Du and S. Ding, "A survey on multi-agent deep reinforcement learning: from the perspective of challenges and applications," *Artificial Intelligence Review*, pp. 1–24, 2020.

[28] S. Iqbal and F. Sha, "Actor-attention-critic for multi-agent reinforcement learning," in *Proceedings of the 36th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97. PMLR, 09–15 Jun 2019, pp. 2961–2970.

[29] Y.-Y. Qian, L. Liu, and G. Feng, "Distributed event-triggered adaptive control for consensus of linear multi-agent systems with external disturbances," *IEEE Transactions on Cybernetics*, vol. 50, no. 5, pp. 2197–2208, 2020.

[30] Y. Liu, D. Yao, H. Li, and R. Lu, "Distributed cooperative compound tracking control for a platoon of vehicles with adaptive nn," *IEEE Transactions on Cybernetics*, vol. 52, no. 7, pp. 7039–7048, 2021.

[31] J. Li, F. Wu, H. Shi, and K.-S. Hwang, "A collaboration of multi-agent model using an interactive interface," *Information Sciences*, vol. 611, pp. 349–363, 2022.

[32] T. A. Badgwell, J. H. Lee, and K.-H. Liu, "Reinforcement learning–overview of recent progress and implications for process control," in *Computer Aided Chemical Engineering*. Elsevier, 2018, vol. 44, pp. 71–85.

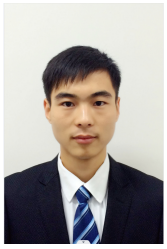
[33] F. A. Oliehoek, "Decentralized PO-MDPs," in *Reinforcement Learning*. Springer, 2012, pp. 471–503.

[34] K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder–decoder for statistical machine translation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1724–1734.

[35] X. Wang, L. Ke, Z. Qiao, and X. Chai, "Large-scale traffic signal control using a novel multiagent reinforcement learning," *IEEE Transactions on Cybernetics*, vol. 51, no. 1, pp. 174–187, 2021.

[36] Y. Yang, R. Luo, M. Li, M. Zhou, W. Zhang, and J. Wang, "Mean field multi-agent reinforcement learning," in *International Conference on Machine Learning*. PMLR, 2018, pp. 5571–5580.

[37] K. Zhu, M. Liu, H. Chen, Z. Zhao, and D. Z. Pan, "Exploring logic optimizations with reinforcement learning and graph convolutional network," in *2020 ACM/IEEE 2nd Workshop on Machine Learning for CAD (MLCAD)*. IEEE, 2020, pp. 145–150.



Xuesi Li received his M.S. degree in the School of Computer Science, Northwestern Polytechnical University, China, in 2017. He is currently working toward Ph.D. degree in the School of Computer Science, Northwestern Polytechnical University, China. His research interests include electrical engineering, intelligent control, machine learning and embedded system design.

JingChen Li received his B.S. degree in the School of Software, Northwestern Polytechnical University, China, in 2017. He get Master degree in the School of Computer Science at Northwestern Polytechnical University. He is currently working toward his Ph.D. in the School of Computer Science, Northwestern Polytechnical University, China. His research interests include mobile robot, intelligent control, and machine learning.

Haobin Shi is a Professor in the School of Computer Science at Northwestern Polytechnical University, China, and visiting scholar of Electrical Engineering Department at National Sun Yat-sen University, Taiwan. He received his Ph.D. degree from Northwestern Polytechnical University, China, in 2008. He is the director of the Chinese Association for Artificial Intelligence. His research interests include intelligent robots, decision support systems, artificial intelligence, multi-agent systems, and machine learning.

Kao-Shing Hwang is an ASE Chair Professor of the Department of Electrical Engineering at National Sun Yat-sen University, and also a professor of the Department of Healthcare Administration and Medical Informatic, Kaohsiung Medical University, Taiwan. He is also a visiting chair professor of the Department of Computer Engineering, Northwestern Polytech. University, Xian, China. He received the M.M.E. and Ph.D. degrees in Electrical and Computer Engineering from Northwestern University, Evanston, IL, U.S.A., in 1989 and 1993, respectively. He had been with National Chung Cheng University in Taiwan from 1993-2011. He was the deputy director of Computer Center (1998-1999), the chairman of the Electrical Engineering Department (2003 2006), and the director of the Opti-mechatronics Institute of the university (2010 2011) in National Chung Cheng University. He has been a member of IEEE since 1993 and a Fellow of the Institution of Engineering and Technology (FIET). His research interest includes methodologies and analysis for various intelligent robot systems, visual servoing, and reinforcement learning for robotic applications.