Multiagent Reinforcement Learning With Graphical Mutual Information Maximization

Shifei Ding, Wei Du, Ling Ding, Jian Zhang, Lili Guo, and Bo An, *Member, IEEE*

***Abstract*— Communication learning is an important research direction in the multiagent reinforcement learning (MARL) domain. Graph neural networks (GNNs) can aggregate the information of neighbor nodes for representation learning. In recent years, several MARL methods leverage GNN to model information interactions between agents to coordinate actions and complete cooperative tasks. However, simply aggregating the information of neighboring agents through GNNs may not extract enough useful information, and the topological relationship information is ignored. To tackle this difficulty, we investigate how to efficiently extract and utilize the rich information of neighbor agents as much as possible in the graph structure, so as to obtain high-quality expressive feature representation to complete the cooperation task. To this end, we present a novel GNN-based MARL method with graphical mutual information (MI) maximization to maximize the correlation between input feature information of neighbor agents and output high-level hidden feature representations. The proposed method extends the traditional idea of MI optimization from graph domain to multiagent system, in which the MI is measured from two aspects: agent features information and agent topological relationships. The proposed method is agnostic to specific MARL methods and can be flexibly integrated with various value function decomposition methods. Considerable experiments on various benchmarks demonstrate that the performance of our proposed method is superior to the existing MARL methods.**

***Index Terms*— Communication learning, graph neural network (GNN), multiagent reinforcement learning (MARL), mutual information (MI).**

I. Introduction

INFORMATION sharing is the key to promote the action coordination of agent teams in multiagent systems, which enables agents to successfully complete the global task through cooperation. Therefore, communication learning has become a hot topic in the MARL domain and achieved great progress in recent years [1], [2], [3]. Most existing MARL methods leverage the centralized training with decentralized execution (CTDE) paradigm for its ability to tackle observability constraint and scalability issue. Communication learning methods [4], [5], [6] leverage communication schemes in the execution stage for further exploiting the CTDE framework.

Graph neural network (GNN) as a class of graph representation method has attracted extensive attention in recent years. GNN can aggregate the information of neighbor nodes and relationships for representation learning [7]. Some MARL methods have leveraged GNN to learn communication among agents. However, most of these methods simply aggregate the information of neighboring agents through GNNs; therefore, the obtained feature representation cannot extract enough useful information, and the relationship topology information is ignored. The high-quality expressive representations are critical for coordinating the actions of agents to complete cooperative tasks. In this sense, it is very critical to preserve and extract as much information as possible from the information of neighbor agents to learn high-quality feature representations.

In this article, to fully inherit the rich information of neighbor agents, we present a Multi-Agent Reinforcement learning (MARL) method with Graphical mutual INformation maximization (MARGIN). The proposed method leverages GNNs to fuse feature information of neighbor agents and obtain high-level feature representations. Besides, the proposed method adopts graphical mutual information (MI) to learn high-quality feature representations. MARGIN uses a straightforward way to calculate the graphical MI without leveraging any readout function or corruption function. As shown in Fig. 1, the graphical MI is directly derived through comparing the subgraph containing the neighbor agents and the feature representation of each agent. MARGIN maximizes the MI of both features and relationships between inputs (subgraph) and outputs (hidden feature representation).

According to the theoretical derivation of [8], the global MI can be factorized to the weighted sum of the local MIs between each input feature and the output hidden feature representation. Therefore, the input agent features can be decomposed and the calculation of the global MI can be tractable. The communication learning based on graphical MI maximization can be easily integrated with the value function decomposition methods via the proposed framework. Therefore, MARGIN can maintain the advantages of scalability and stability of value function decomposition and promotes better action

Manuscript received 30 September 2022; revised 1 January 2023; accepted 5 February 2023. Date of publication 16 February 2023; date of current version 31 October 2025. This work was supported by the National Natural Science Foundation of China under Grant 61976216, Grant 62276265, Grant 62206297, and Grant 61672522. *(Corresponding authors: Wei Du; Ling Ding.)*

Shifei Ding is with the School of Computer Science and Technology, China University of Mining and Technology, Xuzhou 221116, China, and also with the Mine Digitization Engineering Research Center, Ministry of Education, Xuzhou 221116, China (e-mail: dingsf@cumt.edu.cn).

Wei Du, Jian Zhang, and Lili Guo are with the School of Computer Science and Technology, China University of Mining and Technology, Xuzhou 221116, China (e-mail: 1394471165@qq.com; zhangjian10231209@cumt.edu.cn; liliguo@cumt.edu.cn).

Ling Ding is with the College of Intelligence and Computing, Tianjin University, Tianjin 300350, China (e-mail: 414211048@qq.com).

Bo An is with the School of Computer Science and Engineering, Nanyang Technological University, Singapore 639798 (e-mail: boan@ntu.edu.sg).

Digital Object Identifier 10.1109/TNNLS.2023.3243557

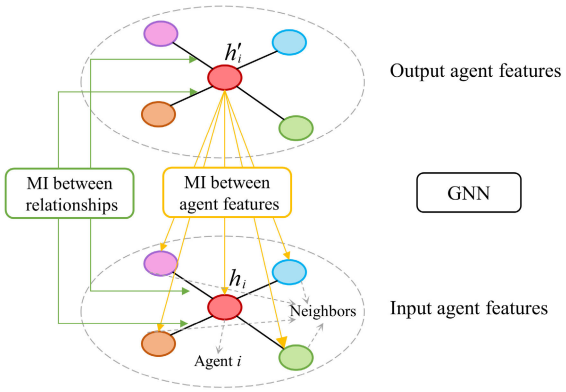

Fig. 1. Overview of graphical MI in MARGIN.

coordination of agents by efficiently utilizing information of neighbors. The primary contributions and novelties of our method are summarized as follows.

1) The proposed method is the first attempt to generalize the graphical MI optimization to the MARL domain for communication learning.

2) The proposed framework integrates the advantage of the GNN, value decomposition, and graphical MI, which lead to efficient communication learning and policy learning.

3) Experiments on various MARL benchmark demonstrate the superiority and feasibility of the proposed method.

II. Related Work

A. MARL With GNN

To further exploit the CTDE framework, a variety of MARL methods design communication protocol among agents. Hu et al. [9] improve the communication efficiency in multiagent systems by designing an event-triggered communication network that enables agents to communicate only when necessary. Xie et al. [10] present a novel communication protocol to efficiently extract local relations and learn communication among neighbor agents by utilizing the capability of depthwise convolution. Recently, various methods utilize GNN to promote communication learning. Jiang et al. [11] adopt graph convolutional networks (GCNs) to model the multiagent scenario for communication learning. Iqbal and Sha [12] use the GNN to develop the centralized critic network and guide the policy learning of decentralized actor networks by the soft actor-critic method. Malysheva et al. [13] introduce a graph generation layer to generate the adjacency matrix of agents. Wang et al. [14] tackle scenarios consisting of multiple types of agents. Sheng et al. [15] utilize the hierarchical GNN to learn effective communication by propagating information among groups and agents.

More recently, Ryu et al. [16] present a hierarchical attention mechanism based on GNNs, which effectively models the relationships among individuals and groups of agents. Liu et al. [17] leverage a complete graph and the proposed two-stage attention mechanism to model the relationship between agents. Niu et al. [18] present an attentional and end-to-end communication mechanism by graph attention networks (GATs) to tackle the issue of how to process messages and when to communicate. Du et al. [19] utilize the heterogeneous GAT to tackle the heterogeneous scenarios, where agents have different attributes or subtasks.

The existing "MARL with GNN" methods have successfully learned communication by modeling the interactions or relationships among agents. However, these methods generally use GNN to aggregate the information of neighboring agents without MI optimization; therefore, the feature representation may not extract enough useful information. The proposed method adopts graphical MI to learn high-quality feature representations. Besides, the proposed method can deal with unfixed-size graphs, which include different numbers of agents.

B. Value Decomposition

In multiagent scenarios, to better coordinate the actions of the cooperative agents, learning a centralized joint value function Q_{tot} under CTDE framework appears to be a valuable solution. Nevertheless, the centralized function is intractable to learn because the joint action space of agents advances exponentially with the expansion of the number of agents. In contrast, learning the decentralized value function Q_i directly for each agent can alleviate scalability problems. However, the decentralized learning method generally ignores the interaction among agents, which often leads to coordination disorder and suboptimal policy. To alleviate this dilemma, the value decomposition methods [20], [21], [22] represent Q_{tot} as a combination of decentralized Q_i based on local information, which has shown effectiveness in complicated tasks. These methods follow the individual-global-max (IGM) principle and use the joint global value function $Q_{\text{tot}}(\tau, a)$ and local value function $[Q_i(\tau_i, a_i)]_{i=1}^n$ to ensure the consistency between the joint optimal action and the selection of the local optimal action

$$\arg\max_{a\in A} Q_{\text{tot}}(\tau, a) = \left[\arg\max_{a_i\in A} Q_i(\tau_i, a_i)\right]_{i=1}^n. \quad (1)$$

Value decomposition network (VDN) [23] uses additivity to decompose the global value function

$$Q_{\text{tot}}^{\text{VDN}}(\tau, a) = \sum_{i=1}^{n} Q_i(\tau_i, a_i). \quad (2)$$

Q mixing network (QMIX) [24] constrains the global value function through the monotonicity condition

$$\forall i \in N, \quad \frac{\partial Q_{\text{tot}}^{\text{QMIX}}(\tau, a)}{\partial Q_i(\tau_i, a_i)} > 0. \quad (3)$$

However, the present value decomposition method mainly focuses on full decomposition, which reduces the complexity of learning centralized action-value function Q_{tot} by learning decentralized individual Q_i first and places the burden of coordinating actions of agents on a mixing network. For large-scale settings with partial observability, no matter how powerful the representation capability of the mixing network is, it is not enough to learn coordinated actions. The full decomposition operation cuts off the dependencies among decentralized individual value functions; thus, agents tend to

produce uncertainty about the actions and states of others. This uncertainty increases over time, leading to severe incoordination and arbitrary performance degradation in decentralized execution.

C. MARL With MI

In recent, many existing MARL methods explicitly maximize the correlation or influence between agents to facilitate collaboration by maximizing the MI between agent actions. Mahajan et al. [25] propose to maximize the MI between future trajectories and the latent variables. The latent variables are extracted from the initial global state. Nevertheless, one disadvantage of this method is that the shared latent variables used in the decentralized execution phase violate the paradigm of CTDE. This makes the method not suitable for some practical scenarios, where global communication is not possible.

Wang et al. [26] present to maximize the MI between communication messages and action selection for coordination, while minimizing the entropy of communication messages among agents. Cao et al. [27] introduce the awareness based on local observation history and maximize the MI between built awareness and the actual trajectory. Li et al. [28] present to maximize the MI associated with high-level collaborative behaviors and minimize the MI with low-level one.

Previous methods generally facilitate collaboration by maximizing MI about agent behavior. However, these studies ignore the fact that highly cooperative agents do not necessarily produce high returns because the actions of agents in suboptimal cooperation can also be highly relevant. Unlike the previous method of establishing MI between actions, our proposed method maximizes the MI between the communication information and the features of neighbor agents to promote collaboration.

III. Proposed Method

In this section, we model the multiagent setting as a graph and propose an MARL method based on GNN via graphical MI optimization. The proposed method contains three components: feature process module, communication module, and value decomposition module. The problem formulation is first described. Then, the framework and components of the proposed method are introduced in detail. The algorithm is presented in the final.

A. Problem Formulation

The MARL issues can be formulated as decentralized partially observable Markov decision process (Dec-POMDPs) [29]. Dec-POMDPs can be represented by a tuple $G = < S, A, P, R, O, N, \gamma >$, in which the state of the partially observable environment is represented as $s \in S$, and the local observation of agent i is represented as $o_i \in O$. The agent i selects its action $a_i \in A$ according to its local observation o_i. The joint action is represented as $a = (a_1, \ldots, a_n)$. The state changes based upon the transition function $P : S \times A \to S$. The objective of the agent i is to maximize its cumulative discounted reward $R_i = \Sigma_{t=0}^{T} \gamma^t r_i^t$,

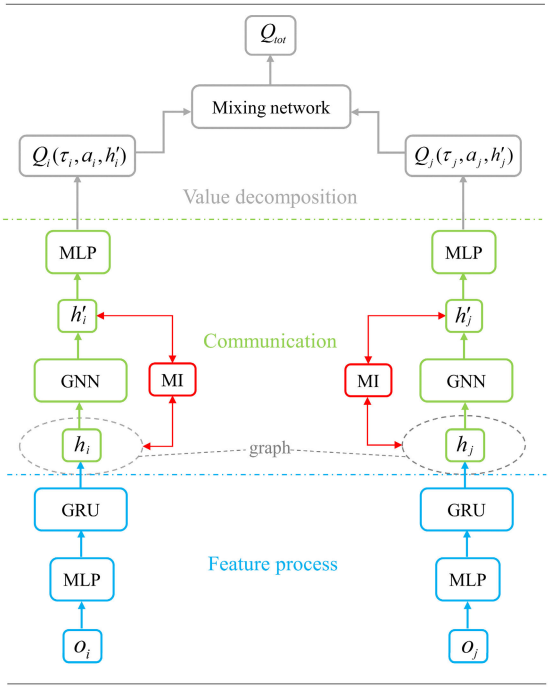

Fig. 2. Framework of MARGIN that consists of feature process module, communication learning module, and value decomposition module.

where $\gamma \in [0, 1]$ denotes a discount factor. The goal is to learn a joint policy $\pi(\tau, a)$ to maximize the global value $Q_{\text{tot}}^{\pi}(\tau, a) = \text{E}_{s,a}[\sum_{t=0}^{\infty} \gamma^t R(s, a) | s_0 = s, a_0 = a]$, where τ denotes the observation history.

B. Framework of MARGIN

In this section, the framework of MARGIN is introduced in detail. MARGIN mainly consists of three module: feature process module, communication learning module, and value decomposition module, as shown in Fig. 2. In the proposed framework, for agent i, we utilize the multilayer perceptron (MLP) and gated recurrent unit (GRU) layer to process the feature h_i based on local observation o_i. MLP uses one fully connected layer to encoder o_i and GRU generates and combines the historical hidden state to output local observation history τ_i, which is considered a feature h_i. Then, we fed the feature h_i into the communication learning module. During the communication learning period, we leverage GNN to generate feature representation h_i' for agent i, which aggregates the information of neighbor agents. Besides, the proposed method adopts graphical MI to learn high-quality feature representations. The global MI can be factorized to the weighted sum of local MI between each input feature h_i and the output hidden feature representation h_i'.

Finally, for agent i, its individual local action-value functions $Q_i(\tau_i, a_i, h_i')$ are calculated based on local observation history τ_i and feature representation h_i' obtained from the communication learning module. Then, the local Q values obtained by different agents are input into a mixing network, (e.g., the network leveraged by QMIX), to obtain an estimation of the global function value. The mixing network only contains MLP, but the weights and biases of the mixing network are obtained

from hypernetwork to satisfy the monotonicity constraints. MARGIN can be integrated with any value decomposition method. The framework uses the paradigm of centralized training and decentralized execution.

C. Communication Learning

The multiagent system can be modeled as graph $G = (V, E)$, where $v_i \in V$ denotes the agent i and $e_{ij} = (v_i, v_j) \in E$ denotes the relationship between agent i and agent j. The agent features are given by $H^{N\times D} = (h_1, \ldots, h_N)$, with assumed empirical probability distribution $P=$, where N denotes the number of agent, D denotes the length of feature vector, and h_i denotes the feature for agent i. The features of neighbor agents of agent i are defined as $H_i^{(|B_i|+1)\times D}$, where $|B_i|$ denotes the number of neighbor agents of agent i. Particularly, $H_i^{(|B_i|+1)\times D}$ consists of all k-hop neighbor agents features of agent i with $k \leq l$ when the communication module uses l-layer GNN and the feature of agent i itself. The corresponding adjacency matrix C_i is constructed for agent i, which is added self-loops. The subgraph expanded by H_i and C_i is defined as a support graph G_i for agent i. With the definition of support graph G_i, the feature representation for each agent i becomes $h'_i = f(G_i) = f(H_i, C_i)$.

We represent the empirical probability distribution of agent features H_i as $p(H_i)$, the probability distribution of h'_i as $p(h'_i)$, and the joint distribution by $p(h'_i, H_i)$. As proved in [8], if the conditional probability $p(h'_i|H_i)$ is multiplicative, the global MI $I(h'_i; H_i)$ can be factorized as a weighted sum of local MIs

$$I\left(h'_i; H_i\right) = \sum_{j}^{|B_i|} w_{ij} I\left(h'_i; h_j\right) \tag{4}$$

where h_j denotes the feature of the jth neighbor of agent i, $|B_i|$ denotes the number of neighbor agents of agent i, and the weight w_{ij} satisfies $1/|B_i| \leq w_{ij} \leq 1$ for each agent j (see [8] for detailed proof).

Inspired by the decomposition in (4), we intend to build trainable weights w_{ij} from the topological view of graphs; therefore, the values of weights w_{ij} can be flexible and capture the inherent relationship of agents. In this end, we present the graphical MI for MARL. The graphical MI between the hidden feature h'_i and its support graph $G_i = (X_i, C_i)$ can be represented as

$$\begin{aligned} I\left(h'_i; G_i\right) &:= \sum_{j}^{|B_i|}\left[w_{ij} I\left(h'_i; h_j\right) + I\left(w_{ij}; c_{ij}\right)\right], \\ w_{ij} &= \sigma\left(h_i'^{T} h'_j\right) \end{aligned} \tag{5}$$

where the definition of h_j and $|B_i|$ is the same as in (4), c_{ij} denotes the elements in the adjacency matrix C_i, and $\sigma(\cdot)$ denotes the sigmoid function.

Concretely, the weight w_{ij} in the first term $w_{ij} I(h'_i; h_j)$ of (5) measures the contribution of local MI $I(h'_i; h_j)$ to global MI $I(h'_i; G_i)$. The contribution of $I(h'_i; h_j)$ is calculated by the similarity between agent feature representations h'_i and h'_j [i.e., $w_{ij} = \sigma(h_i'^{T} h'_j)$]. The second term $I(w_{ij}; c_{ij})$ denotes the MI between w_{ij} and the relationships of agents (edge features c_{ij} of the input graph). By maximizing $I(w_{ij}; c_{ij})$, the weight w_{ij} is constrained to conform to topological relationships. Intuitively, the contribution is consistent with the topological proximity, which is a generally accepted fact that w_{ij} may be larger when agent j is "closer" to agent i. This strategy makes up for the shortcoming that (4) only pays attention to agent features and makes the local MI have adaptive contribution to the global MI.

Previous methods attempt to maximize the MI between the input and output feature of neural networks. However, most of these methods cannot be easily extended to deep neural networks because of the difficulty in computing MI among high-dimensional variables. In recent, MI neural estimation (MINE) [30] proposes to train the statistical network to be the classifier, which is designed to distinguish samples from the joint distribution and the marginal product. Concretely, MINE leverages the exact KL-based formula of MI instead of the alternative method of non-KL, Jensen–Shannon divergence (JSD) [31], which can be used without considering the exact value of MI.

Next, we can maximize the right side of (5) directly based on the idea of MINE. As mentioned above, MINE uses the KL-divergence between the joint distribution and the marginal product to estimate the lower bound of MI. Since we are more concerned with maximizing MI than obtaining its concrete value, other non-KL alternatives can be leveraged to replace it. Specifically, to improve the effectiveness and efficiency of estimation, this article adopts JSD estimator. JSD estimator is more suitable for MARL task because of its insensitivity to negative sampling strategy and its excellent performance in many large-scale scenarios. Concretely, we compute $I(h'_i; h_j)$ of (5) by

$$I\left(h'_i; h_j\right) = -\mathrm{sp}\left(-D_w\left(h'_i; h_j\right)\right) - E_{\tilde{P}}\left[\mathrm{sp}\left(D_w\left(h'_i; \tilde{h}_j\right)\right)\right] \tag{6}$$

where D_w denotes a discriminator leveraged the neural network with parameter $w \cdot \mathrm{sp}(x) = \log(1 + e^x)$ represents the soft-plus function. $\tilde{h}_j$ denotes a negative sampled from $\tilde{P} = P =$. Positive sample h_j is corresponding to feature representation h'_j, and negative sample means that the feature is randomly selected and does not correspond to h'_j.

The second term $I(w_{ij}; c_{ij})$ can be maximized by computing its cross-entropy rather than leveraging the JSD estimator since the graphs constructed in the multiagent environment are unweighted. Formally, it can be calculated as follows:

$$I\left(w_{ij}; c_{ij}\right) = c_{ij} \log w_{ij} + \left(1 - c_{ij}\right)\log\left(1 - w_{ij}\right). \tag{7}$$

By maximizing $I(h'_i; G_i)$ with the sum of (6) and (7) over all feature representations, we can obtain the global objective loss function for MI optimization

$$L_{\mathrm{MI}} = \sum_{i=1}^{N} -I\left(h'_i; G_i\right). \tag{8}$$

In addition, we can add tradeoff parameters in (5) to balance (6) and (7) to increase flexibility.

We can use various GNN structure for information aggregation of neighbor agents. For example, we leverage the standard

GCN to aggregate the information of neighbor agents using the following layerwise propagation rule:

$$H^{(l+1)} = \sigma\left(\hat{A}H^{(l)}W^{(l)}\right) \tag{9}$$

with

$$\hat{A} = D^{-\frac{1}{2}}\bar{A}D^{\frac{1}{2}} \tag{10}$$

where $D_{ii} = \sum_j \bar{A}_{ij}$ and $\bar{A} = A + I_n \cdot H^{(l+1)}$ and $H^{(l)}$ denotes the output and input feature matrices of the lth GNN layer, and $W^{(l)}$ denotes the weight matrix that is layer-specific and trainable. σ denotes the activation function which we use is the parametric rectified linear unit (PReLU) function.

For the selection of GNN, we can adopt various GNN structure, such as standard GCN, GAT [32], and graph sample and aggregate (GraphSAGE) [33] for comparison. The difference between these method mainly lies in the way of neighbor information aggregation. GAT uses the multihead attention mechanism, which can allocate different weights to different neighbor nodes during the information aggregation, thus enhancing its expression ability and getting rid of the problem that GCN is completely dependent on the adjacency matrix with unlearnable coefficients. GraphSAGE proposes to first sample the neighbor nodes and then aggregate the information. Through random sampling, it reduces the dependence on the graph structure and enhances the efficiency compared with standard GCN.

The discriminator in (6) leverages a simple bilinear function to score the input–output feature pairs

$$D\left(h_i'; h_j\right) = \sigma\left(h_i'^T M h_j\right) \tag{11}$$

here M denotes a scoring matrix that is trainable and the nonlinear transformation σ is a sigmoid function, which aims at converting score to the probability that $(h_i'; h_j)$ is a positive example.

D. Overall Optimization Objective

We have introduced graphical MI optimization for communication learning to be efficient. Apart from the MI constraints on the feature representations in the communication learning module, all the parameters in other modules (feature process module and value decomposition module) are updated by minimizing the TD loss. To calculate the global TD loss, as shown in Fig. 2, in the value decomposition module, individual action values are fed into a mixing network, which output the estimation of global action-value Q_{tot}. The proposed communication learning module can be easily integrated with the existing value function decomposition method. In this article, MARGIN leverages the mixing network presented by QMIX [24], and it can be flexibly replaced by other value decomposition methods. Therefore, TD loss and overall optimization objective of MARGIN are presented in (12) and (13), respectively

$$L_{\text{TD}} = \left[r + \gamma \max_{a'} Q_{\text{tot}}\left(\tau', a'; \theta^-\right) - Q_{\text{tot}}(\tau, a; \theta)\right]^2 \tag{12}$$

where θ^- denotes the parameters of target network in DQN, and θ represents all parameters in the model

$$L = L_{\text{TD}} + \lambda L_{\text{MI}} \tag{13}$$

where λ represents the adjustable hyperparameter to accomplish a tradeoff between the TD loss and the sum of MI loss of all agents.

E. Algorithmic Summary

The pseudocode of the MARGIN is presented in Algorithm 1. Lines 5–15 illustrate the decentralized execution phase, with lines 9–11 describing the communication learning module. During the decentralized execution phase, the agents can obtain information from neighbor agents through GNN in communication module and select actions in a decentralized manner. The trained GNN contains a set of learnable weights for agents. Due to the messaging nature of GNN updates, they can be distributed to individual agents during the execution phase.

Algorithm 1 (Pseudo-Code of MARGIN)

Input: The local observation $o_i \in O_i$ of agent i, environment state s, and the action-observation history τ_i of agent i.
Output: The global action-value function Q_{tot}.
1: Initial the parameters of networks, the frequency of network updating, the maximum size of replay buffer $\Re$, the time step t, the maximum length of an episode.
2: **For** each episode **do**
3: **For** time step $t = 1$ *to* n **do**
4: **For** each agent $i \in N$ **do**
5: % During Decentralized execution phase
6: Receive local observation o_i
7: Obtain feature h_i through MLP and GRU
8: Input feature h_i to communication module
9: % During Communication phase
10: Aggregate features of neighbors via GNN
11: Optimize the feature representation h_i' via MI
12: Obtain action-value Q_i based on the h_i' and the action-observation history τ_i
13: Select action $a_i = \pi(Q_i)(\varepsilon - greed)$
14: Store episode history τ_i, a_i in replay buffer
15: Output individual value function $Q_i(\tau_i, a_i, h_i')$
16: % During Centralized training phase.
17: Input $Q_i(\tau_i, a_i, h_i')$ to mixing network
18: Sample histories from the replay buffer.
19: Minimize loss function based on Eq.(12)
20: Update parameters of networks
21: Output global action-value Q_{tot}
22: **End for**
23: **End for**
24: **End for**

Next, Lines 16–21 illustrate the centralized training phase. During the centralized training procedure, we assume that the MARGIN can receive the individual local observation-action histories from replay buffer. The proposed method can be easily fused with the existing value function decomposition method. The value decomposition module is the same as that of QMIX, which takes the individual action values as the input to perform monotone mixing and generates the global action-value Q_{tot}. As shown in Fig. 3, the blue part

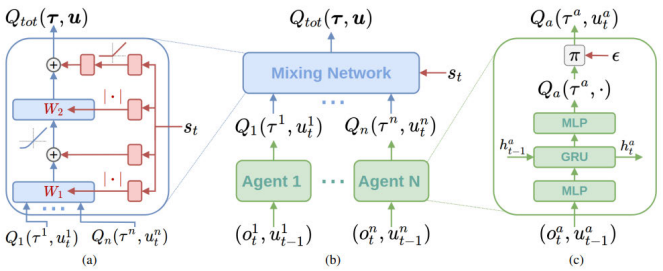

Fig. 3. Framework of QMIX, reproduced from the paper [24].

is the framework of the mixing network, whose weights and biases are produced by hypernetworks (the red) to satisfy the monotonicity constraints. Besides, we leverage the loss function defined in QMIX to train the mixing networks and update the parameters.

IV. Experiments

In this section, the performance of MARGIN is evaluated in several environments, including many-agent reinforcement learning (MAgent) [34] and starcraft multiagent challenge (SMAC) [35]. All experiments are conducted on GPU Nvidia RTX 2080 with Pytorch. We utilize two-layer GNN and the number of neighbor agents depend on scenarios. We employ QMIX [24], graph two-stage attention network (G2ANET) [17], deep graph convolutional reinforcement learning (DGN) [11], and multiagent graph-attention communication (MAGIC) [18] as baselines. These methods leverage the value decomposition or GNN and have the state-of-the-art performance on MAgent and SMAC. Baselines are introduced as follows.

1) QMIX [24] designs a mixing network, in which the joint action value is estimated by the nonlinear combination of individual action values.
2) G2ANET [17] uses a complete graph and two-stage attention mechanism to model the relationship between agents.
3) DGN [11] adopts GNNs to model the multiagent scenario for communication among agents to learn cooperation.
4) MAGIC [18] presents an attentional communication mechanism by GATs to tackle with the issue of how to process messages and when to communicate.

A. MAgent

1) Battle: Battle scenario consists of K agents that are trained to learn how to defeat L more capable enemies. Each agent is able to take action *attack* or *move*, and the objective of the allied agent is to defeat the enemy and obtain more rewards. The range of *move* or *attack* is four adjacent grids. Nevertheless, the enemy agent is able to select to attack 1 of the eight adjacent grids or to move to the 12 most adjacent grids. It is easy for the environment to lose balance after the death of an allied agent or enemy, so we add a new agent or enemy randomly to the environment to keep the balance.

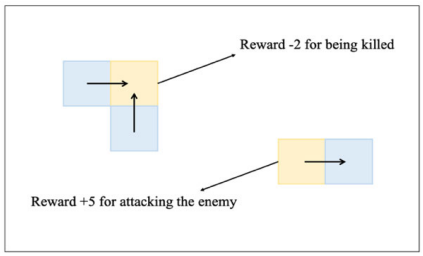

Fig. 4. Illustration of the battle scenario.

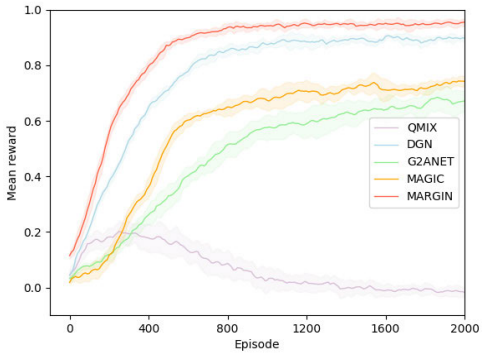

Fig. 5. Learning curves of different methods in battle scenario.

Since a single enemy is more capable than a single agent, thus the agent must formulate strategies to cooperate to fight with other agents. Besides, due to enemy health being six (killed by six hits), agents must continuously cooperate to defeat enemy. As shown in Fig. 4, an agent attacking the enemy can obtain the positive reward of $+5$. An agent gets a negative reward of -2 and -0.01 when it is killed or hit a blank grid, respectively.

We select the pretrained DQN model built in the MAgent benchmark to play the role of enemy agents and let the different MARL methods trained agents battle against DQN agents. Therefore, in addition to the reward of a given time step, we can fairly compare the kill-death rates of different MARL methods. All the MARL methods are trained under the same setup of $K = 40$ and $L = 24$ for 2k episodes. We evaluate MARGIN and the other baseline methods by running 20 test games with 200 timesteps each.

The learning curves of different methods are shown in Fig. 5. For all the methods, the shaded area is surrounded by the minimum and maximum values of the three training sessions, and the mean value is represented by the solid line in the middle. The performance of different methods in the battle scenario is shown in Table I. As shown in Table I and Fig. 5, MARGIN outperforms than all other baselines in terms of average reward, kill-death rates, and attack number among agents.

As can be observed from experiments shown in Fig. 6(left), other baselines learn suboptimal strategies at the beginning of training, such as gathering in the corner to avoid being attacked, as such strategies generate relatively high rewards. Nevertheless, due to the uneven distribution of rewards, agents outside the group are vulnerable to be attacked, and other baselines drew on the "low-reward experience" generated by

TABLE I
PERFORMANCE OF DIFFERENT METHODS IN BATTLE SCENARIO

	MARGIN	DGN	MAGIC	G2ANET	QMIX
Kills	**275**	218	175	136	5
Deaths	**82**	95	91	98	73
K/D ratio	**3.35**	2.29	1.92	1.38	0.07
Mean reward	**0.95**	0.90	0.72	0.63	-0.02

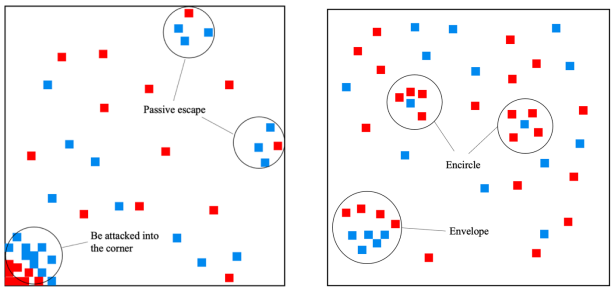

Fig. 6. Illustration of representative behaviors of MARGIN (right) and other baselines (left) in battle scenario.

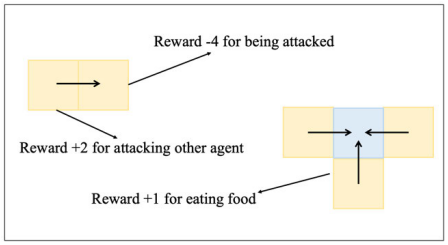

Fig. 7. Illustration of the jungle scenario.

suboptimal strategies to converge to more passive strategies, resulting in much lower rewards.

As shown in Fig. 6(right), MARGIN trained agents can learn a variety of tactical skills, including encircling and enveloping one side. For an individual enemy agent, MARGIN agents can successfully learn to cooperate to surround it and defeat it. For a group of enemies, MARGIN agents successfully learn to attack one flank of the enemy.

2) Jungle: Jungle scenario consists of K agents and F foods (stationary). There is a moral dilemma in the environment, where an agent can get a positive reward for catching food, but a larger reward for attacking other agents. Each agent is able to attack or move to one of the four adjacent grids at each time step. Agents that attack blank grids receive a small negative reward of -0.01 to discourage excessive attacks. As shown in Fig. 7, in the jungle scenario, an agent attacking the food can receive the positive reward $+2$. An agent obtains negative reward of -4 for being attacked. The experiment is designed to test whether agents could learn to cooperate to share resources rather than attack each other. We trained MARGIN and other baselines for 2k episodes with $K = 20$ and $F = 12$.

The learning curves of different methods are shown in Fig. 8. For all the methods, the shaded area is surrounded by the minimum and maximum values of the three training sessions, and the mean value is represented by the solid line in the middle. As shown in Fig. 8, MARGIN outperforms than

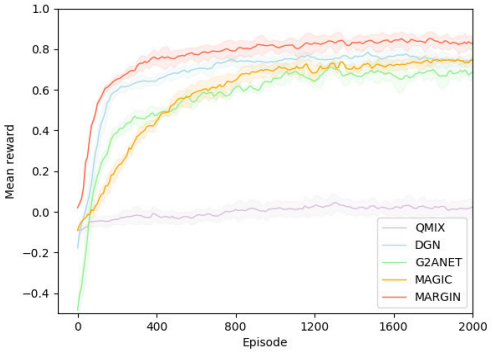

Fig. 8. Learning curves of different methods in jungle scenario.

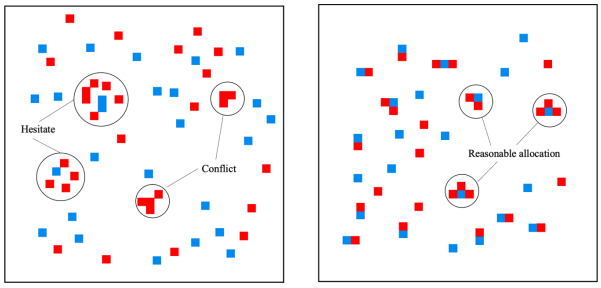

Fig. 9. Illustration of representative behaviors of MARGIN (right) and other baselines (left) in jungle scenario.

TABLE II
PERFORMANCE OF METHODS IN JUNGLE SCENARIO WITH DIFFERENT NUMBERS OF AGENTS

agents	MARGIN	DGN	MAGIC	G2ANET	QMIX
20	**0.75**	0.67	0.71	0.60	0.03
30	**0.86**	0.75	0.80	0.65	0.07
40	**0.90**	0.83	0.86	0.72	0.14
50	**0.93**	0.86	0.89	0.78	0.17
60	**0.95**	0.87	0.90	0.79	0.18

all the baselines. As shown in Fig. 9(right), MARGIN trained agents can allocate the food rationally and choose the food close correctly. Meanwhile, MARGIN trained agents learn to seldom hurt other agents.

Fig. 9 shows the representative behaviors of agents trained by diverse MARL methods. As shown in Fig. 9(left), other baselines trained agents sometimes occur conflicts between agents and sometimes become wary of others and hesitate when they approach the same food. MARGIN agents can learn to share food appropriately and seldom hurt each other. This may due to that the proposed method adopts graphical MI to learn high-quality feature representations, which promote the action coordination and policy learning of agents.

In addition, we compared MARGIN and baselines with the different numbers of agents (from 20 to 60). All settings are the same except that the number of agents and food increases proportionally. As shown in Table II, MARGIN can always perform best as the number of agents increases. The results demonstrate that MARGIN can be applied to large-scale multiagent setting.

TABLE III
SCENARIOS OF SMAC

Scenario	Ally	Enemy	Type
15m vs 17m	15Marines	16Marines	Homogeneous
27m_vs_30m	27Marines	30Marines	Homogeneous
6h vs 8z	6 Hydralisks	8 Zealots	Homogeneous
1c3s5z_vs_1c3s6z	1 Colossus, 3 Stalkers, 5 Zealots	1 Colossus, 3 Stalkers, 6 Zealots	Heterogeneous
MMM2	1 Medivac, 2 Marauders, 7 Marines	1 Medivac, 2 Marauders, 8 Marines	Heterogeneous
MMM3	1 Medivac, 2 Marauders, 7 Marines	1 Medivac, 2 Marauders, 9 Marines	Heterogeneous

TABLE IV
ABLATIONS STUDIES ON GNN TYPE ON BATTLE SCENARIO

GNN Type	Kills	Deaths	K/D ratio	reward
GCN	275	82	3.35	0.95
GAT	291	77	3.78	0.98
GraphSAGE-mean	247	90	2.74	0.93
GraphSAGE-LSTM	283	80	3.54	0.96
GraphSAGE-pool	259	86	3.01	0.94

B. SMAC

1) Setting: We evaluate the proposed method MARGIN and baselines on the SMAC benchmark to demonstrate the superiority of communication learning with graphical MI maximization. In particular, we make it more difficult to coordinate actions among agents. On the one hand, we narrow the field of vision of the agent from 9 to 5. On the other hand, we select the challenging scenarios. Table III shows the details of scenarios, and it is worth noting that all scenarios are super hard level. First, the setting of the SMAC environment is introduced in detail. In four challenging scenarios, ally units are all controlled by the MARL agents and enemy units are mastered by the built-in AI. Enemy units and ally units can be asymmetrical, and their initial positions are random.

The action space of the agent is of dimension four, which includes actions *move*, *noop*, *attack*, and *stop*. Under the control of these actions, agents attack and move in these maps of continuous space. At each time step, the agent receives a reward that is equivalent to the cumulative damage done to the enemy units. The agent can obtain an additional bonus of +10 for each enemy unit it kills and +200 for each battle it wins. We evaluate the proposed method on four challenging scenarios.

2) Performance: Our implementation is based on the PyMARL framework [35] and leverages its default structure of network and hyperparameter settings of the value decomposition module. The hyperparameter settings of other module (feature process module and communication module) are shown in Table IV. In the scenarios of SMAC, the training time of 2 million timesteps is approximately 15–24 h, which is ranged according to the agent number and scenario features of each map.

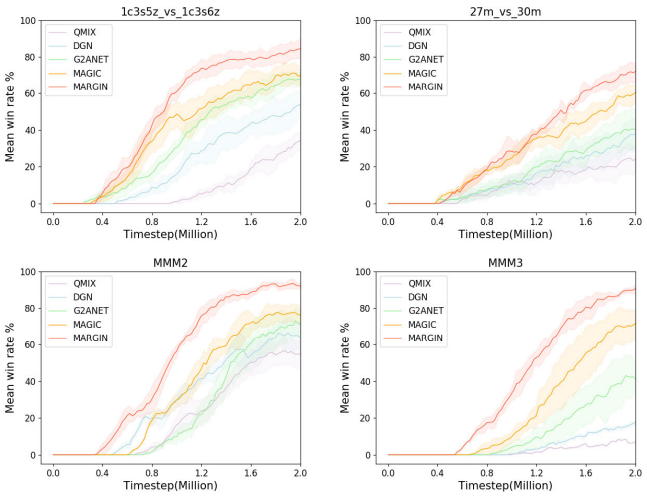

Fig. 10. Learning curves of different methods on SMAC scenarios.

The learning curves of different methods are shown in Fig. 10. For all the methods, the shaded area is surrounded by the minimum and maximum values of the five training sessions, and the mean value is represented by the solid line in the middle. As shown in Fig. 10, MARGIN outperforms other baselines in four super hard scenarios. For example, in the MMM3 scenario, MARGIN-trained agent successfully learned the strategy: the Medivacs first approach the enemy, absorb fire, and then withdraw to heal the appropriate allied agent.

3) Ablations: In this section, our ablation experiments are designed to answer the following questions: 1) whether each component of MARGIN is effective? 2) Does the superiority of MARGIN come from the increase in the number of parameters? 3) How does MARGIN perform with different GNN models? and 4) Can MARGIN fuse with various value decomposition methods?

To further evaluate the effectiveness of each module in MARGIN, we design the ablation studies on average test win rates on scenarios 15_versus_17 m and 6 h versus 8 z. We design two variants of MARGIN: 1) MARGIN-I is MARGIN without graphical MI maximization and 2) MARGIN-V is MARGIN without value decomposition module. By comparing MARGIN and MARGIN-I, we can see that the removal of the graphical MI maximization causes a drop in performance. Moreover, when comparing MARGIN and MARGIN-V, we can see that the removal of the value decomposition module results in a slight performance decline. These experimental results show that communication learning can indeed strengthen cooperation among the agents, that MI optimization can improve the quality of communication, and that value decomposition can further enhance the collaboration among agents.

We also design QMIX-LARGE with similar number of parameters as MARGIN to study whether the fact that MARGIN outperforms QMIX lies in the increase of the number of parameters. As shown in Fig. 11, the results show that QMIX with a larger network does not fundamentally enhance performance. The two variants show performance degradation

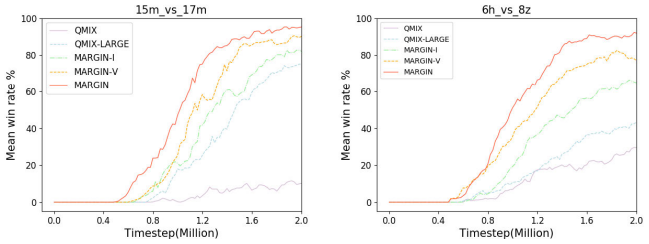

Fig. 11. Ablations studies on average test win rates on SMAC scenarios.

TABLE V
ABLATIONS STUDIES ON VALUE DECOMPOSITION METHOD TYPE ON SMAC SCENARIOS

Method	15m vs 17m	6h vs 8z	MMM2	MMM3
VDN	12	16	29	8
Ours (VDN)	91	87	85	78
QMIX	10	30	51	45
Ours (QMIX)	95	92	92	90
QPLEX	24	17	32	15
Ours (QPLEX)	96	90	89	87

TABLE VI
NOTATIONS AND EXPLANATIONS

Notation	Explanation
h_i	Input feature
h_i'	Output feature representation
C_i	Adjacency matrix of neighbor agents
D_w	Discriminator
$P=$	Empirical probability distribution
Q_i	Individual action-value
Q_{tot}	Global action-value
M	Trainable scoring matrix

compared to MARGIN, illustrating the effectiveness of each module.

In addition, we use different GNN structures in the MARGIN on the battle scenario to evaluate its performance. As shown in Table VI, we used three variants of GraphSAGE, of which GraphSAGE-LSTM has the best performance. Among all GNN models, GAT achieves the best effect. The possible reason is that by introducing the attention mechanism, GAT enables the agent to better assign weight to the information of the neighbor agents, and at the same time, it can handle different numbers of neighbor agents. Meanwhile, the graphical MI preserves the topology structure of the graph well.

MARGIN can be fused with any value decomposition methods. For example, we integrate it with popular value decomposition methods VDN, QMIX, and QPLEX. The fused methods, called MARGIN (VDN), MARGIN (QMIX), and MARGIN (QPLEX), respectively, are tested on various scenarios of SMAC. As shown in Table V, all fused methods perform better than the original value decomposition methods, which indicates that the communication learning module of MARGIN can significantly enhance the policy learning and action coordination of agents.

TABLE VII
FIXED HYPER-PARAMETERS OF MARGIN FOR ALL EXPERIMENTS

Hyper-parameter	Value
GNN layers	3
MLP layers	2
GNN activation	PReLU
Discriminator activation	Sigmod
Discount factor	0.99
Learning rate	1e-5
Optimizer	Adam
Mixing network	QMIX

TABLE VIII
HYPERPARAMETERS OF MARGIN FOR MAGENT AND SMAC

Hyper-parameter	MAgent	SMAC
Batch size	50	100
Random seeds	3	5
Action dim	2	4
Number of neighbors	5&3	3&2
Number of agents	20-60	14-57

C. Notations and Hyperparameters

The notations and explanations that we leverage throughout the method are summarized in Table VI.

Table VII describes the fixed parameters in all experiments. Table VIII gives the parameters of different benchmarks.

V. CONCLUSION

In this article, we introduce the graphical MI optimization concept for MARL. The propose method realizes efficient communication learning through maximizing the graphical MI between the input and output of the constructed graph in terms of agent features and agent relationships. Besides, the method can be flexibly fused with diverse MARL methods. Experimental results on various environments illustrate that the proposed method outperforms the existing MARL methods.

To the best of our knowledge, our work is the first attempt at learning communication via GNNs and graphical MI optimization in MARL domain. We believe that communication learning via GNN and MI optimization is a promising way in building efficient and versatile large-scale multiagent systems. Future work will focus on applying ideas from this work to heterogeneous multiagent environments.

REFERENCES

[1] T. T. Nguyen, N. D. Nguyen, and S. Nahavandi, "Deep reinforcement learning for multiagent systems: A review of challenges, solutions, and applications," *IEEE Trans. Cybern.*, vol. 50, no. 9, pp. 3826–3839, Sep. 2020.

[2] C. Sun, W. Liu, and L. Dong, "Reinforcement learning with task decomposition for cooperative multiagent systems," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 5, pp. 2054–2065, May 2021.

[3] Z. Zhang, D. Wang, and J. Gao, "Learning automata-based multiagent reinforcement learning for optimization of cooperative tasks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 10, pp. 4639–4652, Oct. 2020.

[4] S. Sukhbaatar et al., "Learning multiagent communication with backpropagation," in *Proc. Adv. neural Inf. Process. Syst.*, 2016, pp. 2244–2252.

[5] A. Singh, T. Jain, and S. Sukhbaatar, "Learning when to communicate at scale in multiagent cooperative and competitive tasks," in *Proc. Int. Conf. Learn. Represent.*, 2018, pp. 1–15.

[6] A. Das et al., "Tarmac: Targeted multi-agent communication," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 1538–1546.

[7] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 1263–1272.

[8] Z. Peng et al., "Graph representation learning via graphical mutual information maximization," in *Proc. Web Conf.*, 2020, pp. 259–270.

[9] G. Hu, Y. Zhu, D. Zhao, M. Zhao, and J. Hao, "Event-triggered communication network with limited-bandwidth constraint for multi-agent reinforcement learning," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Nov. 21, 2021, doi: 10.1109/TNNLS.2021.3121546.

[10] D. Xie, Z. Wang, C. Chen, and D. Dong, "Depthwise convolution for multi-agent communication with enhanced mean-field approximation," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Dec. 29, 2022, doi: 10.1109/TNNLS.2022.3230701.

[11] J. Jiang, C. Dun, T. Huang, and Z. Lu, "Graph convolutional reinforcement learning," 2018, *arXiv:1810.09202*.

[12] S. Iqbal and F. Sha, "Actor-attention-critic for multi-agent reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 2961–2970.

[13] A. Malysheva, D. Kudenko, and A. Shpilman, "MAGNet: Multi-agent graph network for deep multi-agent reinforcement learning," in *Proc. 16th Int. Symp. Problems Redundancy Inf. Control Syst. (REDUNDANCY)*, Oct. 2019, pp. 171–176.

[14] T. Wang, R. Liao, J. Ba, and S. Fidler, "NerveNet: Learning structured policy with graph neural networks," in *Proc. Int. Conf. Learn. Represent.*, 2018, pp. 1–54.

[15] J. Sheng et al., "Learning structured communication for multi-agent reinforcement learning," 2020, *arXiv:2002.04235*.

[16] H. Ryu, H. Shin, and J. Park, "Multi-agent actor-critic with hierarchical graph attention network," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 7236–7243.

[17] Y. Liu, W. Wang, Y. Hu, J. Hao, X. Chen, and Y. Gao, "Multi-agent game abstraction via graph attention neural network," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 7211–7218.

[18] Y. Niu, R. Paleja, and M. Gombolay, "Multi-agent graph-attention communication and teaming," in Proc. International Conference on Autonomous Agents and Multi Agent Systems, 2021.

[19] W. Du, S. Ding, C. Zhang, and Z. Shi, "Multiagent reinforcement learning with heterogeneous graph attention network," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Nov. 4, 2022, doi: 10.1109/TNNLS.2022.3215774.

[20] Y. Yang et al., "Q-value path decomposition for deep multiagent reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 10706–10715.

[21] W. J. K. D. E. Hostallero, K. Son, D. Kim, and Y. Qtran, "Learning to factorize with transformation for cooperative multi-agent reinforcement learning," in *Proc. 31st Int. Conf. Mach. Learn.*, 2019, pp. 5887–5896.

[22] J. H. Wang, Z. Z. Ren, T. Liu, Y. Yu, and C. J. Zhang, "QPLEX: Duplex dueling multi-agent Q-learning," in *Proc. 9th Int. Conf. Learn. Represent.*, 2021, pp. 1–27.

[23] P. Sunehag et al., "Value-decomposition networks for cooperative multiagent learning," in *Proc. AAMAS*, 2018, pp. 1–17.

[24] T. Rashid, M. Samvelyan, C. Schroeder, G. Farquhar, J. Foerster, and S. Whiteson, "Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 4295–4304.

[25] A. Mahajan, T. Rashid, M. Samvelyan, and S. Whiteson, "Maven: Multi-agent variational exploration," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 7613–7624.

[26] T. Wang, J. Wang, C. Zheng, and C. Zhang, "Learning nearly decomposable value functions via communication minimization," 2019, *arXiv:1910.05366*.

[27] J. Cao, L. Yuan, J. Wang, S. Zhang, C. Zhang, and Y. Yu, "LINDA: Multi-agent local information decomposition for awareness of teammates," 2021, *arXiv:2109.12508*.

[28] P. Li, H. Tang, and T. Yang, "PMIC: Improving multi-agent reinforcement learning with progressive mutual information collaboration," in *Proc. 9th Int. Conf. Learn. Represent.*, 2022, pp. 1–19.

[29] W. Du and S. Ding, "A survey on multi-agent deep reinforcement learning: From the perspective of challenges and applications," *Artif. Intell. Rev.*, vol. 54, pp. 1–24, Jun. 2020.

[30] M. I. Belghazi, A. Baratin, S. Rajeswar, and S. Ozair, "Mutual information neural estimation," in *Proc. Int. Conf. Mach. Learn.*, Stockholm, Sweden, Jul. 2018, pp. 531–540.

[31] S. Nowozin, B. Cseke, and R. Tomioka, "F-GAN: Training generative neural samplers using variational divergence minimization," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 271–279.

[32] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," 2017, *arXiv:1710.10903*.

[33] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 1–11.

[34] L. Zheng et al., "MAgent: A many-agent reinforcement learning platform for artificial collective intelligence," in *Proc. 32nd Conf. Artif. Intell.*, 2018, pp. 8222–8223.

[35] M. Samvelyan et al., "The StarCraft multi-agent challenge," in *Proc. 18th Int. Conf. Auto. Agents MultiAgent Syst.*, 2019, pp. 2186–2188.