Bridging Training and Execution via Dynamic Directed Graph-Based Communication in Cooperative Multi-Agent Systems

Zhuohui Zhang[1,2], **Bin He**[1,2], **Bin Cheng**[1,2*], **Gang Li**[1,2]

[1]Department of Control Science & Engineering, Tongji University, China
[2]National Key Laboratory of Autonomous Intelligent Unmanned Systems, Tongji University, China
{zhangzhuohui, hebin, bincheng, lig}@tongji.edu.cn

Abstract

Multi-agent systems must learn to communicate and understand interactions between agents to achieve cooperative goals in partially observed tasks. However, existing approaches lack a dynamic directed communication mechanism and rely on global states, thus diminishing the role of communication in centralized training. Thus, we propose the Transformer-based graph coarsening network (TGCNet), a novel multi-agent reinforcement learning (MARL) algorithm. TGCNet learns the topological structure of a dynamic directed graph to represent the communication policy and integrates graph coarsening networks to approximate the representation of global state during training. It also utilizes the Transformer decoder for feature extraction during execution. Experiments on multiple cooperative MARL benchmarks demonstrate state-of-the-art performance compared to popular MARL algorithms. Further ablation studies validate the effectiveness of our dynamic directed graph communication mechanism and graph coarsening networks.

Code — https://github.com/ZhuohuiZhang/TGCNet

1 Introduction

Cooperative multi-agent reinforcement learning (MARL) problems have emerged in the past decade as a vital framework for addressing intricate collaborative challenges in real-world scenarios (Wei et al. 2019; Vinyals et al. 2019; Cheng et al. 2024; Li et al. 2023), attracting considerable attention and showcasing substantial potential for practical applications and commercial viability. A natural approach to cooperative MARL regards the multi-agent system as a whole using centralized methods (Tan 1993), which face scalability challenges and limitations associated with centralized controllers (Foerster et al. 2018). Another approach is decentralized, where each agent learns its policy with single agent techniques (Tampuu et al. 2017), addressing the limitations of centralized controllers but introducing non-stationarity and credit assignment problems (Lowe et al. 2017). To mitigate these issues, decentralized policies can be learned using the paradigm of centralized training with decentralized execution (CTDE) (Sunehag et al. 2017; Yuan et al. 2022; Yu et al. 2022; Zhang, Zhang, and Lin 2019), which includes value-based (Rashid et al. 2020; Eccles et al. 2019; Wang et al. 2021) and policy-based (Yu et al. 2022; Lin et al. 2021) frameworks. CTDE-based MARL algorithms rely on access to the global state, which is often an idealized assumption.

Communication is essential in multi-agent systems for sharing information, learning, and collaborating toward common goals, especially in partially observable environments. It is crucial for complex tasks such as coordinating autonomous vehicles (Cao et al. 2012), sensor networks (Pipattanasomporn, Feroze, and Rahman 2009), and multi-robot control (Fox et al. 2000). Effective communication underpins cooperation. Recently, inspired by human cooperation, communication has been integrated into MARL to enhance information sharing among agents. Early work (Foerster et al. 2016; Sukhbaatar, Fergus et al. 2016) disseminated information through a broadcast format, but this approach led to high communication costs and information redundancy. Subsequent research has aimed to reduce communication overhead by selectively determining when to communicate (Singh, Jain, and Sukhbaatar 2018) and eliminating redundant information through targeted peer-to-peer communication (Jiang and Lu 2018; Jiang et al. 2018). However, there is a lack of a universal communication method that can address the following five issues simultaneously with (1) whom to communicate, (2) when to communicate, (3) which piece of information to communicate, (4) how to combine and integrate the information received, and (5) how to use communication to avoid reliance on global state are limited.

In this work, we focus on developing communication mechanisms in MARL to enhance team performance on collaborative tasks, while minimizing communication costs and avoiding reliance on global state information. We propose a novel MARL algorithm, the Transformer-based graph coarsening network (TGCNet), which models cooperative agents' communication as dynamic directed graphs, with each agent as a node. The adjacency trajectory matrix represents the structure of the dynamic directed graph. We connect the training and execution phases through dynamic directed graphs, allowing agents to communicate during both stages. TGCNet integrates a Transformer-based (Vaswani et al. 2017) multi-key gated communication mechanism with the Q-network, enabling end-to-end training without the need

*Corresponding author

for additional loss functions. The multi-key gated mechanism learns through a hard additive attention approach (Bahdanau, Cho, and Bengio 2014), using multi-keys for aggregation to determine the number of communications between agents. Based on the dynamic directed graph, we introduce a graph coarsening network that utilizes self-attention pooling and coarsening operations to approximate the global state. We conduct a comprehensive empirical study across diverse cooperative multi-agent benchmarks to evaluate the efficacy of our proposed methodology. Specifically, we evaluate our method on the Hallway scenario (Wang et al. 2019), the Level-Based Foraging (LBF) environment (Rangwala and Williams 2020), and eight distinct maps from the StarCraft Multi-Agent Challenge (SMAC) (Samvelyan et al. 2019). In addition, we perform component analyses to demonstrate the effectiveness of the multi-key gated communication network and the graph coarsening network. The contributions of this study are as follows:

- We formalize a novel general paradigm for communicative and cooperative MARL, which bridges training and execution through dynamic directed graphs. The communication policy is represented by the structure of dynamic directed graphs, which is shared during both training and execution.
- We design a multi-key gated communication network, which learns the structure of a dynamic directed graph. It can achieve multiple peer-to-peer directed communications between agents within the same time step, an efficient mechanism for information transmission.
- During centralized training, graph coarsening networks process the mix network, while in distributed execution, a Transformer architecture handles the Q-network; both phases leverage learned communication policies as inputs for global state coarsening aggregation and information communication with feature extraction, respectively.

2 Related Works

Communication is vital for MARL to capture the dependencies between agent actions. In addition, it has been demonstrated to effectively improve exploration and team rewards (Apicella et al. 2012). Our work builds on and relates to previous research on MARL and communication mechanisms. The existing MARL can be divided into two categories based on whether or not a communication mechanism is set up. MARL without communication uses the CTDE paradigm, which has been successfully implemented with value-based and policy-based algorithms. The focus of policy-based MARL is to stabilize training with centralized state value estimation. Representative works in this area include COMA (Foerster et al. 2018), MADDPG (Lowe et al. 2017), and MAPPO (Yu et al. 2022). The other category is value-based MARL, following the Individual-Global-Max (IGM) principle; it focuses on value function factorization. This approach includes VDN (Sunehag et al. 2017), QMIX (Rashid et al. 2020), QTRAN (Son et al. 2019), and QPlex (Wang et al. 2020). Although these algorithms have shown significant performance in many multi-agent cooperative tasks, their effectiveness relies heavily on the introduction of global state and the setup of centralized trainer. Unlike existing works, our algorithm facilitates communication among agents, thereby eliminating reliance on global state.

The second category focuses on the use of communication mechanisms to compensate for the limitations of local observations. Previous works enforce fixed and static broadcasting for communication. RIAL and DIAL (Foerster et al. 2016) are applied to improve communication skills by broadcasting messages across time steps. CommNet (Sukhbaatar, Fergus et al. 2016) broadcasts hidden states as messages and obtains fused information by averaging. IC3Net (Singh, Jain, and Sukhbaatar 2018) includes a gating mechanism to learn when to broadcast using CommNet. BiCNet (Peng et al. 2017) and ATOC (Jiang and Lu 2018) consider bidirectional communication and use bidirectional RNNs (Zaremba, Sutskever, and Vinyals 2014) or LSTMs (Greff et al. 2016) to achieve bidirectional broadcast. Recent studies aim to develop more sophisticated communication mechanisms to prevent indiscriminate message broadcasting. DGN (Jiang et al. 2018) leverages graph convolutional networks with relational kernels to capture dynamic agent interactions. TarMAC (Das et al. 2019) adopts an attention mechanism to determine whether two agents must communicate and distinguishes the importance of incoming messages. G2ANet (Liu et al. 2020) constructs a communication interaction graph using a two-stage attention mechanism. However, these algorithms only facilitate communication before agents take actions, neglecting communication during the centralized training process. We propose a new multi-key gated communication network for bidirectional, modeling communication as a dynamic directed graph, combined with Transformer for information fusion and retrieval, to address these concerns. Moreover, we integrate graph neural networks to convert the communication flow into a structured network of dynamic directed graphs that change dynamically at each time step, and leverage graph coarsening network to approximate the global state.

3 Preliminaries

Decentralized Partially Observable Markov Decision Processes (Dec-POMDP) We model cooperative multi-agent systems as Dec-POMDP (Oliehoek and Amato 2016), which imposes partially observable settings on multi-agent Markov decision processes. A Dec-POMDP can be described by a tuple $\langle \mathcal{N}, \mathcal{S}, \mathcal{A}, P, \Omega, O, R, \gamma, \mathcal{M} \rangle$, where $\mathcal{N} = \{1, \cdots, n\}$ indicates the set of n agents, $\mathcal{S}$ denotes the set of global states, $\mathcal{A}$ refers to the set of actions, P represents the state transition function, Ω denotes the set of observations, O refers to the observation function, R represents the reward function, $\gamma \in [0, 1)$ indicates the discount factor, and $\mathcal{M}$ is the set of messages that agents can communicate. At each time step t, each agent $i \in \mathcal{N}$ receives an observation $o_t^i \in \Omega$ from the observation function $O(s_t, i)$, where $s_t \in \mathcal{S}$. Each agent follows an individual policy $\pi^i(a_t^i|\tau_t^i, m_t^i)$, where $\tau_t^i = (o_1^i, a_1^i, \ldots, o_{t-1}^i, a_{t-1}^i, o_t^i)$ is the action-observation history of agent i, and $m_t^i \in \mathcal{M}$ is the message received by agent i at time t. The joint action $\boldsymbol{a}_t = \langle a_t^1, \ldots, a_t^n \rangle$ leads to the next state $s_{t+1} \sim P(s_{t+1}|s_t, \boldsymbol{a}_t)$,

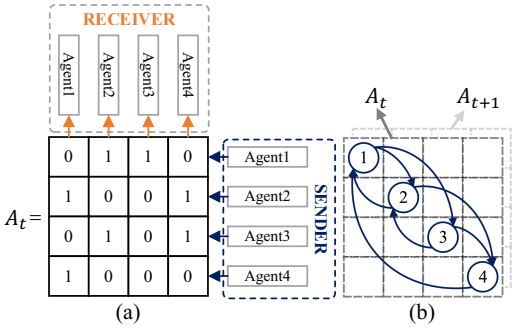


Figure 1: Dynamic directed graph. (a) Adjacency trajectory matrix at time t. (b) Dynamic directed graph. The dynamic directed graph can represent the associations and communication structures between nodes at a certain moment.

and the team receives a global reward $R(s_t, \boldsymbol{a}_t)$. The objective is to find a joint policy $\boldsymbol{\pi} = \langle \pi^1, \ldots, \pi^n \rangle$ that maximizes the global action-value function $Q^{\boldsymbol{\pi}}_{\text{tot}}(\boldsymbol{\tau}, \boldsymbol{a}) = \mathbb{E}_{s,\boldsymbol{a}} \left[\sum_{t=0}^{\infty} \gamma^t R(s, \boldsymbol{a}) \mid s_0 = s, \boldsymbol{a}_0 = \boldsymbol{a}, \boldsymbol{\pi} \right]$, with $\boldsymbol{\tau} = \langle \tau_1, \ldots, \tau_n \rangle$. Each agent serves as both sender and receiver.

Dec-POMDP with Dynamic Directed Graph Based on Dec-POMDP, we propose a communication topology using dynamic directed graphs. The structural properties of the graph are represented by the adjacency trajectory matrix $\mathbf{A}$, which is denoted as $\mathbf{A} \in \mathbb{B}^{l \times n \times n}$, where $\mathbb{B}$ indicates the Boolean matrix, l denotes the length of the trajectory and n indicates the number of agents. If $\lceil A_t^{ij} \rceil = 0$, then no communication occurred from agent i to agent j at time t. If $\lceil A_t^{ij} \rceil = 1$, then communication transpired from agent i to agent j at time t. As an example, the adjacency trajectory matrix A_t at time t is shown in Figure 1(a), and the corresponding dynamic directed graph is shown in Figure 1(b). After traversing the dynamic directed graph to agent i, the received message is computed as $\tilde{m}_t^i = A_t^i \odot m_t^i$, where $\odot$ means element-wise multiplication. For value-based MARL, which uses the action-value function to update policies, where in the distributed execution phase, each agent learns a Q-Network $Q^i(\tau, a, \tilde{m}; \theta)$ (Mnih et al. 2015) to approximate the action-value function $Q^i(s, a)$. In the centralized training phase, the group of agents learns a mix network $Q_{\text{tot}}(\boldsymbol{\tau}, \boldsymbol{a}, \tilde{\boldsymbol{m}}, s; \boldsymbol{\theta})$ to approximate the global action value function $Q_{\text{tot}}(s, \boldsymbol{a})$. The parameters $\boldsymbol{\theta}$ are learned by minimizing the expected temporal difference (TD) error:

$$\mathcal{L}(\boldsymbol{\theta}) = \sum_{i=1}^{b} \left[\left(y_i^{\text{tot}} - Q_{\text{tot}}(\boldsymbol{\tau}_t, \boldsymbol{a}_t, \tilde{\boldsymbol{m}}_t, s_t; \boldsymbol{\theta}) \right)^2 \right], \quad (1)$$

where b is the batch size of transitions sampled from the replay buffer $\mathcal{D}$ and $y^{\text{tot}} = R(s_t, \boldsymbol{a}_t) + \gamma \max_{\boldsymbol{a}_{t+1}} Q_{\text{tot}}(\boldsymbol{\tau}_{t+1}, \boldsymbol{a}_{t+1}, \tilde{\boldsymbol{m}}_{t+1}, s_{t+1}; \boldsymbol{\theta}^-)$, $\boldsymbol{\theta}^-$ are the parameters of a target network.

4 Method

In this section, we introduce the detailed structure and design of TGCNet. Fundamentally, TGCNet is an extension of value-based MARL. Our original intention is for TGCNet to simultaneously solve the five communication problems for MARL, specifically in Section 1. Our motivation stems from the objective of communication in MARL, which is to mitigate the negative effects of limited observations in the Dec-POMDP. Through communication, agents can obtain an approximation of the global state. Our aim is to learn a state function $S(o_t^i, m_t^i, A_t^i)$. Moreover, if an agent's information significantly affects the global state for agent i during centralized training, it should also be crucial during decentralized execution. Communication plays two roles in TGCNet. In the Q-network, the agent uses communication to acquire information beyond its immediate observations to assist in decision making, and the message m_t^i at time t defined as the action-observation history of all other agents except agent i, denoted as τ_t^{-i}. In the mix network, agents coarsen and aggregate individual local observations into global states through communication and the message m_t^i at time t defined as the local observations of all other agents except agent i, denoted as o_t^{-i}. In addition, the network is scalable to scenarios where agents may change and can be plugged into any CTDE structure.

4.1 Theoretical Analysis

We first theoretically analyze how the state function is constructed in the mix network. We construct a 3D dynamic directed which is defined as $\mathbf{G} = (T, V, E)$, where $t \in T$ is the set of the trajectory in the time dimension, $v^i \in V$ denotes the set of vertex, which is composed of agents and $e^{i,j} \in E$ denotes the set of edges connecting agent i and agent j. A dynamic directed graph can be regarded as a composition of T subgraphs $G_t = (V_t, E_t)$ organized along a temporal trajectory. We define the adjacency trajectory matrix to describe the connections (communication) associations between graph vertices. The adjacency trajectory matrices for the dynamic directed graph and subgraphs are denoted by $\mathbf{A}$ and A_t. When a node can access the observations of other nodes, it can approximate the global state. However, this information may be either independent or partially redundant. We define the relationship between the global state and local observation using the following equation:

$$\begin{aligned} s_t = &\text{Agg}\left(o_t^j, \forall \nu^j \in N(\nu^i) \parallel o_t^i\right) \\ &-\text{Overlap}\left(o_t^j, \forall \nu^j \in N(\nu^i) \parallel o_t^i\right), \end{aligned} \quad (2)$$

where $\text{Agg}(\cdot)$ indicates an aggregator function, $N(\nu^i)$ represents the set of neighbor nodes of node i, $\text{Overlap}(\cdot)$ denotes an overlap function and $\parallel$ represents the operator of parallel concatenation, which connects two vectors in parallel. It is evident that $o_t^j, \forall \nu^j \in N(\nu^i) \parallel o_t^i$ is essentially equivalent to $\tilde{m}_t^i$. When dealing with aggregation and overlap functions, it is crucial to maintain consistent output results regardless of the order of neighboring nodes. We design the summation aggregation functions Agg^{sum} and pooling overlap functions $\text{Overlap}^{\text{pool}}$, as follows:

$$\text{Agg}^{\text{sum}} = \sigma\left(\text{sum}\left\{W\left[\tilde{m}_t^i \parallel o_t^i\right] + b\right\}\right), \quad (3)$$

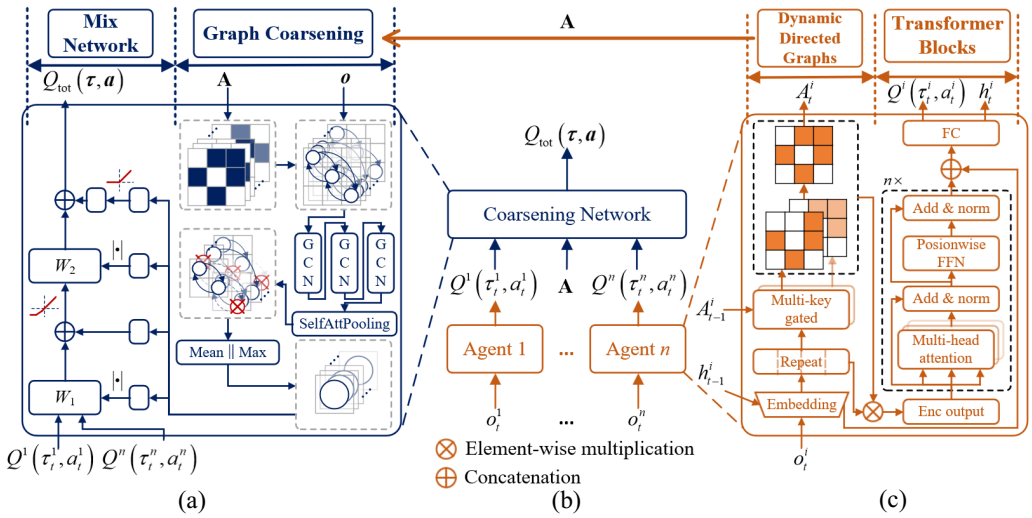


Figure 2: The network structure diagram of TGCNet. (a) Graph coarsening network and mix network. The inputs for the graph coarsening network include the local observations of the agents and the adjacency trajectory matrix. (b) Overall TGCNet architecture. (c) Transformer-Based multi-key gated communication mechanism. The communication mechanism outputs not only the individual state value function $Q^i(\tau^i, a^i)$ and hidden variables h_t^i but also the adjacency trajectory matrix A_t^i. Then, this matrix is passed into the graph coarsening network to complete end-to-end backward propagation updates.

$$\text{Overlap}^{\text{pool}} = \max\left\{\sigma\left(W\left[\tilde{m}_t^i \parallel o_t^i\right] + b\right)\right\}, \quad (4)$$

where σ denotes the activation function, W and b represent the weights and bias of a feed-forward neural network.

After constructing the state function, we only need to replace s from Section 3 to obtain the global action value function of TGCNet, as definded $Q_{\text{tot}}(\boldsymbol{\tau}, \boldsymbol{a}, \tilde{\boldsymbol{m}}, \text{Agg}(\tilde{\boldsymbol{m}} \parallel \boldsymbol{o}) - \text{Overlep}(\tilde{\boldsymbol{m}} \parallel \boldsymbol{o}); \boldsymbol{\theta})$. The expression of Q_{tot} shows that it excludes the global state as an input. The network design schematics are shown in Figure 2.

4.2 Graph Coarsening Network

We construct the graph coarsening network based on Equation (2) to address the last of the five communication problems outlined in Section 1. The schematics of the graph coarsening network are shown in Figure 2(a). The dynamic directed graph, composed of multiple subgraphs, can be considered a coarsened graph, where each subgraph is a super node to fit the global state. This approach enables hierarchical learning of global information. According to the theoretical analysis in Section 4.1, the input of the graph coarsening network consists of each agent's local observations o_t^i, the adjacency trajectory matrix A_t^i and the received message $\tilde{m}_t^i$. We normalize the adjacency trajectory matrix to prevent degree bias and gradient vanishing during training. We define $[\tilde{m}_t^i \parallel o_t^i]$ as X_t^i as feature inputs for each node in graph convolutional networks (GCN). The specific network structure is expressed as follows:

$$\tilde{X}_t^i = \sigma\left(\tilde{D}_t^{i\,-1/2} \tilde{A}_t^i \tilde{D}_t^{i\,-1/2} X_t^i\right), \quad (5)$$

where $\tilde{A}_t^i$ represents the adjacency matrix with self-loops $\tilde{A}_t^i = A_t^i + I$, I means identity matrix and $\tilde{D}_t^i$ represents its degree matrix. Following three iterations of graph convolution, the output is fed into self attention pooling (SelfAttPooling) (Lee, Lee, and Kang 2019). The structure is as follows:

$$\tilde{X}_t^{i\prime} = \tilde{X}_t^i \odot \tanh\left(\tilde{D}_t^{i\,-1/2} \tilde{A}_t^i \tilde{D}_t^{i\,-1/2} \tilde{X}_t^i\right). \quad (6)$$

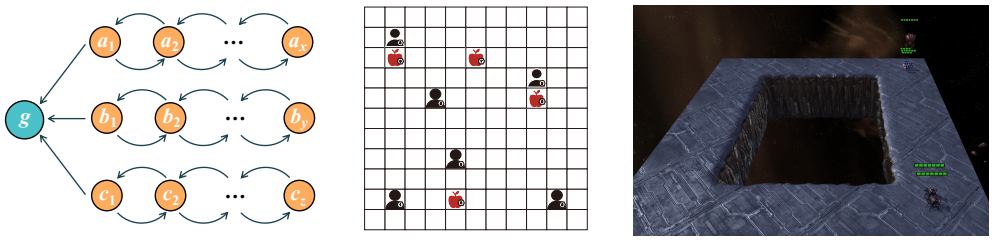
Based on Equations (3) and (4), we have developed a readout mechanism that performs a one-time aggregation operation on all nodes, resulting in a globally coarsened representation of the graph that approximates the global state function. This readout mechanism is similar to the global pooling operation commonly used after the last convolutional layer in CNN models (LeCun et al. 1998). Both approaches aggregate all inputs into a global representation in a single step, expressed as follows:

$$s_t = \sigma\left(W\left[\frac{1}{N}\sum_{i=1}^{n} \tilde{X}_t^{i\prime} \parallel \max_{i=1}^{n} \tilde{X}_t^{i\prime}\right] + b\right). \quad (7)$$

After approximating the global state using graph coarsening networks, we implement the output of Q_{tot} in Section 4.1 using a similar approach to QMIX (Rashid et al. 2020).

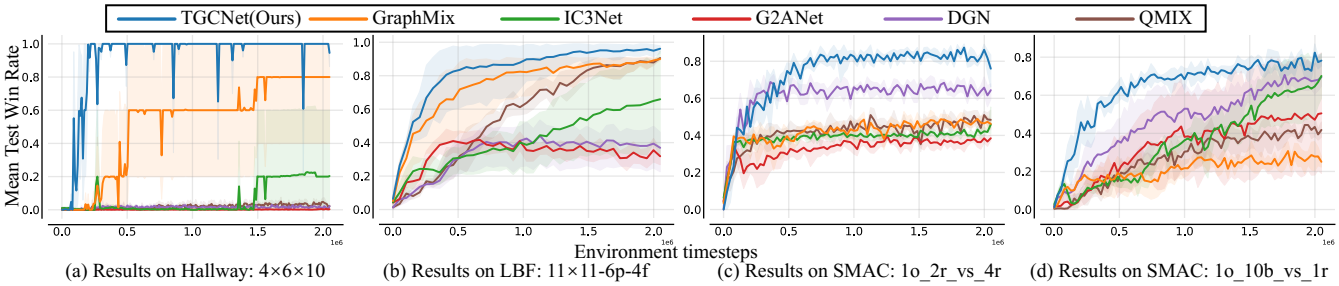
4.3 Transformer-Based Multi-Key Gated Communication Mechanism

After constructing $S(o_t^i, m_t^i, A_t^i)$, we design a multi-key gated mechanism to learn A_t^i in the Q-network. Combined with the Transformer decoder, it addresses the first four of the five communication problems discussed in Section 1. The multi-key gated communication network is the encoder, and the Transformer is the decoder. The schematics of the communication mechanism are shown in Figure 2(c).



(a) Hallway: $x \times y \times z$ (b) Level-Based Foraging: 11×11 (c) SMAC: 1o_2r_vs_4r

Figure 3: Multiple benchmarks used in our experiments.



(a) Results on Hallway: 4×6×10 (b) Results on LBF: 11×11-6p-4f (c) Results on SMAC: 1o_2r_vs_4r (d) Results on SMAC: 1o_10b_vs_1r

Figure 4: Performance comparison with baselines on multiple benchmarks.

Historical information is initialized, and the observations of each agent are pre-processed by embedding their local observations o_t^i and historical information h_{t-1}^i. Then, the adjacency trajectory matrix A_{t-1}^i is initialized and all its elements are set to 1, except the agent i itself, which is masked off. We combine the initialized adjacency trajectory matrix and the information of each agent to perform a pre-communication, which indicates that each agent repeats the hidden variables of the other agents. The pre-communication message $\tilde{m}_{t-1}^i = A_{t-1}^i \odot m_t^i$ is regarded as input and processed by the multi-key gated communication network. The frequency of communication among the agents can be determined by the number of keys in the multi-key gated communication network. The output of each key k_t^i represents the updated adjacency trajectory matrix in this communication. The specific calculation equation is as follows:

$$k_t^i = \text{gumbel-softmax}\left(W_v^\top \tanh(W_q \tilde{m}_{t-1}^i + W_k \tilde{m}_{t-1}^i)\right), \tag{8}$$

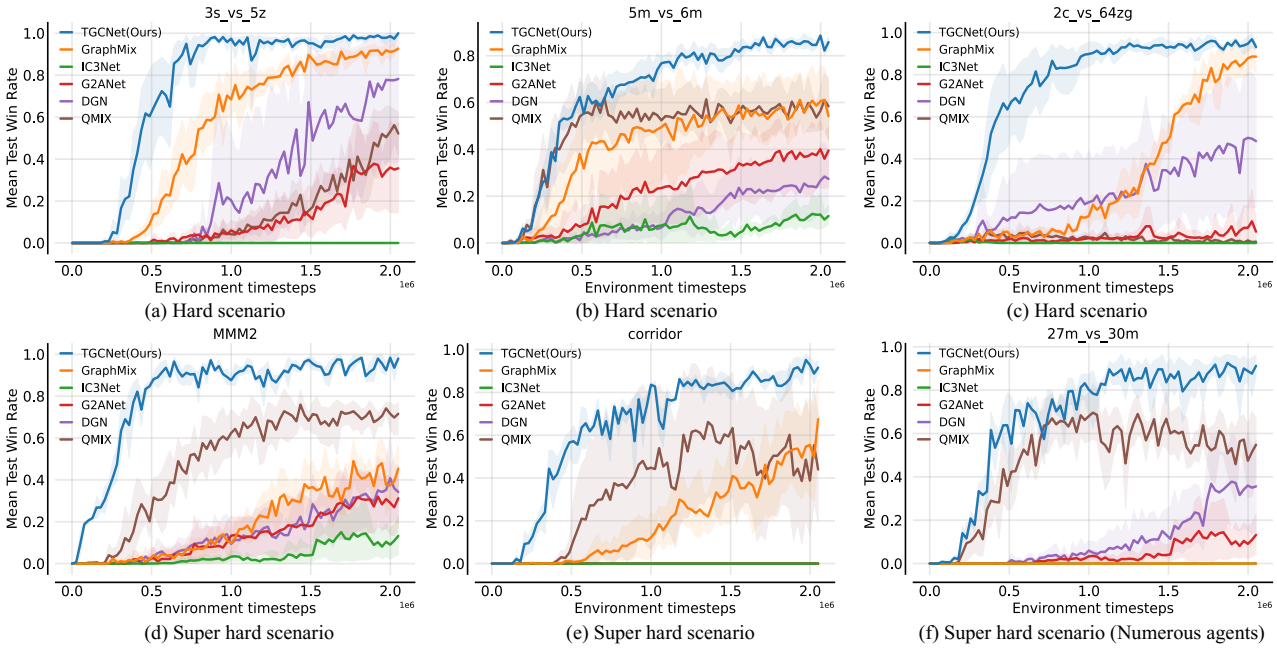
where gumbel-softmax (Jang, Gu, and Poole 2016) is an activation function for two output dimensions (communicate or not). After passing through communication network, the adjacency trajectory matrix is expressed as follows:

$$A_t^i = \lceil \max_j k_t^i|_j) \rceil, \tag{9}$$

where $k_t^i|_j$ represents the total of j keys obtained by agent i at time t and $\lceil \cdot \rceil$ represents rounding up. The transmission and reception of message are completed by using the updated adjacency trajectory matrix, combined with pre-communication information. The output of the encoder $\tilde{m}_t^i = A_t^i \odot m_t^i$ is used as input for the Transformer decoder. Each Transformer block is composed of self-attention mechanism, position-wise feed-forward networks, residual connection and layer normalization. Two Transformer blocks are commonly used to achieve an improved balance between performance and memory consumption. The output of the Transformer blocks and the previous history information concatenated together go through a fully connected layer to obtain the individual state value function Q^i and the hidden state at the next moment h_t^i.

5 Experiments

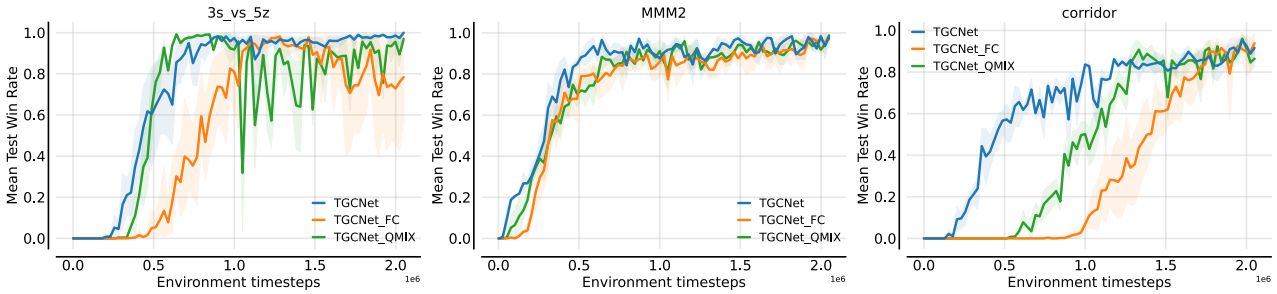
In this section, we evaluate our approach against five representative MARL baselines on various benchmarks with differing communication requirements. QMIX (Rashid et al. 2020) is a strong non-communication baseline, showing outstanding performance across multiple multi-agent benchmarks(Papoudakis et al. 2020). IC3Net (Singh, Jain, and Sukhbaatar 2018) includes a gating mechanism for learning when to broadcast messages, using CommNet (Sukhbaatar, Fergus et al. 2016). DGN (Jiang et al. 2018) uses graph convolutional networks with relational kernels to capture dynamic agent interactions. G2ANet (Liu et al. 2020) builds a sparse communication interaction graph with a two-stage attention mechanism. GraphMix (Naderializadeh et al. 2020) also constructs a communication model using graph structures and is a notably effective baseline. All algorithms use the EPyMARL (Papoudakis et al. 2020) implementation. We evaluate TGCNet with multiple state-of-the-art baselines on tasks including Hallway (Wang et al. 2019), LBF (Papoudakis et al. 2020), and the SMAC (Samvelyan et al. 2019). Figure 3(a) shows the Hallway task, where four agents start at different states (a_1 to a_x, b_1 to b_y, and c_1 to c_z, with $x, y, z = 4, 6, 10$) and must reach the goal g. Figure



(a) Hard scenario (b) Hard scenario (c) Hard scenario

(d) Super hard scenario (e) Super hard scenario (f) Super hard scenario (Numerous agents)

Figure 5: Performance comparison with baselines on hard (first row) and super hard (second row) scenarios in SMAC.



(a) Hard scenario (b) Super hard scenario (c) Super hard scenario

Figure 6: Ablation study results of TGCNet, TGCNet_FC, and TGCNet_QMIX on three hard and super hard scenarios.

3(b) depicts the LBF task, where six agents cooperate to collect four portions of food in an 11×11 grid world. In SMAC, the 1o2r_vs_4r and 1o10b_vs_1r maps (Wang et al. 2019) require collaboration and communication among agents to identify enemy positions, as shown in Figure 3(c). For fair evaluation, all experiments are conducted with five random seeds, and results are presented as means with a 95% confidence interval.

5.1 Overall Performance Comparison

We start by evaluating the overall performance of TGCNet against multiple baselines in communication-intensive benchmarks. As shown in Figure 4, TGCNet consistently outperforms the other methods with low variance across all benchmarks, demonstrating its robustness in scenarios of varying difficulty. In the Hallway task (Figure 4(a)), which requires frequent communication for success, methods lacking communication capabilities, such as QMIX, fail. Other communication-based methods also perform poorly or fail entirely in this environment. This finding indicates that inappropriate message generation or selection would injure the learning process. The performance of TGCNet exceeds GraphMix by approximately 20% given its effective communication modeling and powerful feature extraction capabilities. In the LBF task (Figure 4(b)), existing communication-based MARL methods, such as IC3Net, G2ANet, and DGN encounter challenges due to sparse rewards, especially when the food items are widely distributed. Different from its performance in Hallway, QMIX performs effectively in LBF because agents can observe nearby grids. Our method matches the performance of GraphMix, demonstrating strong coordination abilities even in scenarios with sparse rewards. For scenarios requiring communication in SMAC, QMIX performs the worst without a communication mechanism, whereas TGCNet can maintain high efficiency of learning and exhibit consistently competitive performance when converged, superior to other baselines.

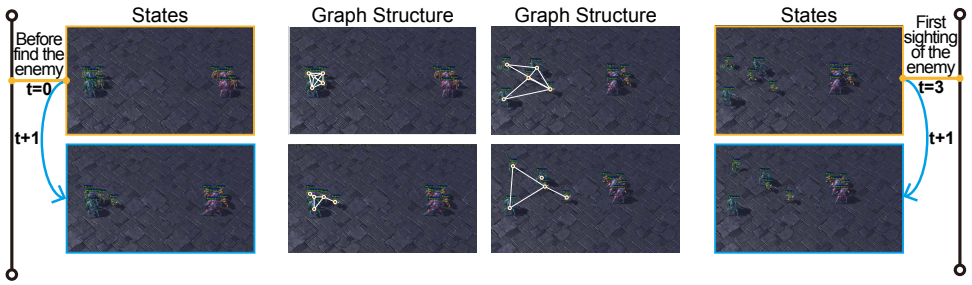


Figure 7: Information Completion visualization and analysis along the MARL trajectory.

To validate our algorithm's information extraction and scalability, we test it in SMAC scenarios without communication, including hard and super hard levels. As shown in Figure 5, TGCNet achieves the highest average test win rate across all scenarios upon convergence. GraphMix performs slightly worse than TGCNet but has a faster convergence speed and higher win rate than DGN and G2ANet due to its graph-based communication information aggregation. IC3Net, DGN, and G2ANet perform worse than QMIX in scenarios without communication because their inaccurate communication modeling fails to manage redundant information effectively, impacting the training of reinforcement learning policy networks. In hard scenarios, TGCNet's average test win rate is nearly 20% higher than those of algorithms. This finding is attributed to the Transformer-based decoder's superior representation of policy networks. In super hard scenarios with more agents, especially on the 27m_vs_30m map, TGCNet maintains a higher convergence speed due to its multi-key gated communication network. Overall, TGCNet demonstrates excellent performance in all scenarios, learning effective communication strategies through its Transformer-based multi-key gated communication mechanism and accurately adjusting the global state with the graph coarsening network.

5.2 Ablation Studies

To understand the superior performance of TGCNet, we conduct ablation studies to evaluate the contributions of its two main components, addressing the following questions: (1) Is the graph coarsening network effective for fitting the global state? How does it compare to using real global states? (2) How does the algorithm's effectiveness differ between full and partial communication? To address the first question, we create the TGCNet_QMix algorithm, replacing TGCNet's graph coarsening network with a hybrid network using QMIX, while retaining other network structures. For the second question, we create TGCNet_FC, which removes TGCNet's multi-key communication network and uses full communication (FC). We conduct ablation experiments on one hard scenario and two super hard scenarios. The results shown in Figure 6 illustrate that TGCNet performs similarly to TGCNet_QMIX in these scenarios, indicating that TGCNet's graph coarsening network effectively approximates the global state. TGCNet_FC converges more gradually than TGCNet in the corridor scenario but ultimately achieves the same performance level. This finding suggests that the multi-key communication network optimizes the communication structure without sacrificing performance, thereby avoiding communication waste.

5.3 Communication Performance

To analyze the communication strategies learned through the graph structure, we conduct a visualization analysis of the multi-agent communication network after training, focusing on a test scenario in SMAC. We illustrate the evolution of the graph structure over the trajectory, as shown in the Figure 7. Initially, at $t = 0$, no enemies are found within the allies' field of view, and the agents are unaware of each other's information. This condition result in a fully connected graph, indicating a state of complete communication. In the next timestep, if the agents' observations and states remain unchanged after their initial communication, the graph structure connections and communication volume decreased. At $t = 3$, communication increas again when enemies entered the agents' fields of view. Similarly, in subsequent timesteps, minimal information changes lead to a synchronous reduction in communication. In summary, the visualization results indicate that the multi-key communication network learned by TGCNet possesses a certain level of interpretability.

6 Conclusions and Future Work

In this work, we propose TGCNet, a novel MARL algorithm that jointly learns communication policies and dynamic directed graph topologies to maximize team rewards. Our key contributions include formalizing a new paradigm for communicative, cooperative MARL that bridges training and execution through dynamic directed graphs, designing a Transformer-based multi-key communication mechanism for selective information transfer, leveraging graph coarsening networks to aggregate local observations, and demonstrating state-of-the-art performance on various cooperative MARL benchmarks. To the best of our knowledge, TGCNet is the first approach to avoid the input of global states through graph coarsening networks and to apply dynamic directed graphs to MARL. In the future, we plan to evaluate TGCNet on more complex real-world multi-agent domains, investigate emergent communication protocols and coordination behavior, and incorporate model-based planning and exploration techniques.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under grants 62103302 and 62088101, by Shanghai Rising-Star Program under grant 24QA2709400, by the Shanghai Chenguang Program under grant 22CGA19, by the Shanghai Municipal Science and Technology Major Project under grant 2021SHZDZX0100, and by Fundamental Research Funds for the Central Universities under grant 22120240276.

References

Apicella, C. L.; Marlowe, F. W.; Fowler, J. H.; and Christakis, N. A. 2012. Social networks and cooperation in hunter-gatherers. *Nature*, 481(7382): 497–501.

Bahdanau, D.; Cho, K.; and Bengio, Y. 2014. Neural Machine Translation by Jointly Learning to Align and Translate. arXiv:1409.0473.

Cao, Y.; Yu, W.; Ren, W.; and Chen, G. 2012. An Overview of Recent Progress in the Study of Distributed Multi-Agent Coordination. *IEEE Transactions on Industrial Informatics*, 9(1): 427–438.

Cheng, B.; Lv, Y.; Li, Z.; and Duan, Z. 2024. Discrete Communication and Control Updating in Adaptive Dynamic Event-Triggered Consensus. *IEEE Transactions on Automatic Control*, 69(1): 347–354.

Das, A.; Gervet, T.; Romoff, J.; Batra, D.; Parikh, D.; Rabbat, M.; and Pineau, J. 2019. Tarmac: Targeted Multi-Agent Communication. In *Proceedings of the 36th International Conference on Machine Learning*, 1538–1546. Long Beach, Calif: PMLR.

Eccles, T.; Bachrach, Y.; Lever, G.; Lazaridou, A.; and Graepel, T. 2019. Biases for Emergent Communication in Multi-Agent Reinforcement Learning. *Advances in Neural Information Processing Systems*, 32: 13111–13122.

Foerster, J.; Assael, I. A.; De Freitas, N.; and Whiteson, S. 2016. Learning to Communicate with Deep Multi-Agent Reinforcement Learning. *Advances in Neural Information Processing Systems*, 29: 2137–2145.

Foerster, J.; Farquhar, G.; Afouras, T.; Nardelli, N.; and Whiteson, S. 2018. Counterfactual Multi-Agent Policy Gradients. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, 2974–2982. New Orleans, LA: AAAI Press.

Fox, D.; Burgard, W.; Kruppa, H.; and Thrun, S. 2000. A Probabilistic Approach to Collaborative Multi-Robot Localization. *Autonomous Robots*, 8(3): 325–344.

Greff, K.; Srivastava, R. K.; Koutník, J.; Steunebrink, B. R.; and Schmidhuber, J. 2016. LSTM: A Search Space Odyssey. *IEEE Transactions on Neural Networks and Learning Systems*, 28(10): 2222–2232.

Jang, E.; Gu, S.; and Poole, B. 2016. Categorical Reparameterization with Gumbel-Softmax. arXiv:1611.01144.

Jiang, J.; Dun, C.; Huang, T.; and Lu, Z. 2018. Graph Convolutional Reinforcement Learning. arXiv:1810.09202.

Jiang, J.; and Lu, Z. 2018. Learning Attentional Communication for Multi-Agent Cooperation. *Advances in Neural Information Processing Systems*, 31: 7254–7264.

LeCun, Y.; Bottou, L.; Bengio, Y.; and Haffner, P. 1998. Gradient-Based Learning Applied to Document Recognition. *Proceedings of the IEEE*, 86(11): 2278–2324.

Lee, J.; Lee, I.; and Kang, J. 2019. Self-Attention Graph Pooling. In *Proceedings of the 36th International Conference on Machine Learning*, 3734–3743. Long Beach, Calif: PMLR.

Li, Z.; Cheng, B.; Song, W.; and Zhang, S. 2023. *Distributed Event-triggered Control: Scalability and Robustness*. Berlin, Germany: Springer.

Lin, T.; Huh, J.; Stauffer, C.; Lim, S. N.; and Isola, P. 2021. Learning to Ground Multi-Agent Communication With Autoencoders. *Advances in Neural Information Processing Systems*, 34: 15230–15242.

Liu, Y.; Wang, W.; Hu, Y.; Hao, J.; Chen, X.; and Gao, Y. 2020. Multi-Agent Game Abstraction via Graph Attention Neural Network. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence*, volume 34, 7211–7218. New York, NY: AAAI Press.

Lowe, R.; Wu, Y. I.; Tamar, A.; Harb, J.; Pieter Abbeel, O.; and Mordatch, I. 2017. Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments. *Advances in Neural Information Processing Systems*, 30: 6379–6390.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A. A.; Veness, J.; Bellemare, M. G.; Graves, A.; Riedmiller, M.; Fidjeland, A. K.; Ostrovski, G.; et al. 2015. Human-level control through deep reinforcement learning. *Nature*, 518(7540): 529–533.

Naderializadeh, N.; Hung, F. H.; Soleyman, S.; and Khosla, D. 2020. Graph Convolutional Value Decomposition in Multi-Agent Reinforcement Learning. arXiv:2010.04740.

Oliehoek, F. A.; and Amato, C. 2016. *A Concise Introduction to Decentralized POMDPs*. Berlin, Germany: Springer.

Papoudakis, G.; Christianos, F.; Schäfer, L.; and Albrecht, S. V. 2020. Benchmarking Multi-Agent Deep Reinforcement Learning Algorithms in Cooperative Tasks. arXiv:2006.07869.

Peng, P.; Wen, Y.; Yang, Y.; Yuan, Q.; Tang, Z.; Long, H.; and Wang, J. 2017. Multiagent Bidirectionally-Coordinated Nets: Emergence of Human-level Coordination in Learning to Play Starcraft Combat Games. arXiv:1703.10069.

Pipattanasomporn, M.; Feroze, H.; and Rahman, S. 2009. Multi-agent systems in a distributed smart grid: Design and implementation. In *2009 IEEE/PES Power Systems Conference and Exposition*, 1–8. Seattle, WA: IEEE.

Rangwala, M.; and Williams, R. 2020. Learning Multi-Agent Communication through Structured Attentive Reasoning. *Advances in Neural Information Processing Systems*, 33: 10088–10098.

Rashid, T.; Samvelyan, M.; De Witt, C. S.; Farquhar, G.; Foerster, J.; and Whiteson, S. 2020. Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning. *The Journal of Machine Learning Research*, 21(1): 7234–7284.

Samvelyan, M.; Rashid, T.; De Witt, C. S.; Farquhar, G.; Nardelli, N.; Rudner, T. G.; Hung, C.-M.; Torr, P. H.; Foerster, J.; and Whiteson, S. 2019. The Starcraft Multi-Agent Challenge. arXiv:1902.04043.

Singh, A.; Jain, T.; and Sukhbaatar, S. 2018. Learning When to Communicate at Scale in Multiagent Cooperative and Competitive Tasks. arXiv:1812.09755.

Son, K.; Kim, D.; Kang, W. J.; Hostallero, D. E.; and Yi, Y. 2019. Qtran: Learning to Factorize with Transformation for Cooperative Multi-Agent Reinforcement Learning. In *Proceedings of the 36th International Conference on Machine Learning*, 5887–5896. Long Beach, Calif: PMLR.

Sukhbaatar, S.; Fergus, R.; et al. 2016. Learning Multiagent Communication with Backpropagation. *Advances in Neural Information Processing Systems*, 29: 2244–2252.

Sunehag, P.; Lever, G.; Gruslys, A.; Czarnecki, W. M.; Zambaldi, V.; Jaderberg, M.; Lanctot, M.; Sonnerat, N.; Leibo, J. Z.; Tuyls, K.; et al. 2017. Value-decomposition Networks for Cooperative Multi-Agent Learning. arXiv:1706.05296.

Tampuu, A.; Matiisen, T.; Kodelja, D.; Kuzovkin, I.; Korjus, K.; Aru, J.; Aru, J.; and Vicente, R. 2017. Multiagent cooperation and competition with deep reinforcement learning. *PloS One*, 12(4): 2395–2411.

Tan, M. 1993. Multi-Agent Reinforcement Learning: Independent vs. Cooperative Agents. In *Proceedings of the 10th International Conference on Machine Learning*, 330–337. New York, NY: PMLR.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is All you Need. *Advances in Neural Information Processing Systems*, 30: 5998–6008.

Vinyals, O.; Babuschkin, I.; Czarnecki, W. M.; Mathieu, M.; Dudzik, A.; Chung, J.; Choi, D. H.; Powell, R.; Ewalds, T.; Georgiev, P.; et al. 2019. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, 575(7782): 350–354.

Wang, J.; Ren, Z.; Han, B.; Ye, J.; and Zhang, C. 2021. Towards Understanding Cooperative Multi-Agent Q-Learning with Value Factorization. *Advances in Neural Information Processing Systems*, 34: 29142–29155.

Wang, J.; Ren, Z.; Liu, T.; Yu, Y.; and Zhang, C. 2020. Qplex: Duplex Dueling Multi-Agent Q-Learning. arXiv:2008.01062.

Wang, T.; Wang, J.; Zheng, C.; and Zhang, C. 2019. Learning Nearly Decomposable Value Functions via Communication Minimization. arXiv:1910.05366.

Wei, H.; Xu, N.; Zhang, H.; Zheng, G.; Zang, X.; Chen, C.; Zhang, W.; Zhu, Y.; Xu, K.; and Li, Z. 2019. CoLight: Learning Network-level Cooperation for Traffic Signal Control. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, 1913–1922. New York, NY: ACM Organization.

Yu, C.; Velu, A.; Vinitsky, E.; Gao, J.; Wang, Y.; Bayen, A.; and Wu, Y. 2022. The Surprising Effectiveness of PPO in Cooperative, Multi-Agent Games. *Advances in Neural Information Processing Systems*, 35: 24611–24624.

Yuan, L.; Wang, J.; Zhang, F.; Wang, C.; Zhang, Z.; Yu, Y.; and Zhang, C. 2022. Multi-Agent Incentive Communication via Decentralized Teammate Modeling. In *Proceedings of the 36th AAAI Conference on Artificial Intelligence*, 9466–9474. Virtual: AAAI Press.

Zaremba, W.; Sutskever, I.; and Vinyals, O. 2014. Recurrent Neural Network Regularization. arXiv:1409.2329.

Zhang, S. Q.; Zhang, Q.; and Lin, J. 2019. Efficient Communication in Multi-Agent Reinforcement Learning via Variance Based Control. *Advances in Neural Information Processing Systems*, 32: 3235–3244.