IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 30 June 2024, accepted 13 August 2024, date of publication 21 August 2024, date of current version 14 October 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3447056

APPLIED RESEARCH

Multi-Agent Transformer Networks With Graph Attention

WOOBEEN JIN AND HYUKJOON LEE
Department of Computer Engineering, Kwangwoon University, Seoul 01897, South Korea
Corresponding author: Hyukjoon Lee (hlee@kw.ac.kr)

This work was supported in part by the Ministry of Science and Information and Communication Technology (MSIT), under the National Program for Excellence in Software (SW) Supervised by the Institute for Information & Communication Technology Planning & Evaluation (IITP) under Grant 2017-0-00096; and in part by the Research Grant of Kwangwoon University in 2024.

ABSTRACT In addressing multi-agent reinforcement learning (MARL) challenges, Multi-Agent Transformer (MAT) has demonstrated a number of notable successes. In various benchmarks, MAT consistently showed a strong performance. A key observation in the latest MAT frameworks is MARL modeling with sequence modeling (SM) to represent the inter-agent relationships by self-attention mechanisms. This study applies graph-based modeling to represent the inter-agent relationships present in agent interactions to improve performance. To this end, we introduce the so-called MAT-GAT model, which leverages Graph Attention Networks (GAT) to allow for individualized consideration of interactions between agents. This enables MAT to pay more attention to information relative to inter-agent interactions within a cooperative MARL environment. To evaluate the performance of MAT-GAT, we conducted a series of benchmark tests across three different levels of StarCraft Multi-Agent Challenge (SMAC) tasks and the MuJoCo Half-cheetah task. The test results indicate that MAT-GAT outperforms both the original MAT and state-of-the-art baselines such as QMIX, particularly in complex environments. This demonstrates MAT-GAT's improved performance with respect to its representation capabilities and learning.

INDEX TERMS Multi-agent, reinforcement learning, transformer, GAT, SMAC, MuJoCo.

I. INTRODUCTION

Recently, the field of multi-agent reinforcement learning (MARL) has experienced rapid development and enormous progress especially in many complex real-world challenges such as robot swarms [1] and autonomous vehicles [2], because of the tremendous success of deep learning technology which allows highly complex action-value functions and policy functions to be closely approximated without explicit computation of all the possible values of states and actions. The ultimate goal of MARL is to carefully balance the policies of individual agents for the joint maximization of the team's rewards based on individual contributions. The design and implementation of cooperative MARL systems, in particular, require extensive consideration of scalability with respect to the joint action space, which can grow exponentially with the number of agents.

The associate editor coordinating the review of this manuscript and approving it for publication was Seifedine Kadry.

The main challenges of cooperative MARL include partial observability and communication constraints, which necessitate that agents determine actions based solely on local information. The centralized training with decentralized execution (CTDE) regime [3], [4] has gained wide acceptance in the cooperative MARL research community and has paved the way for the recent progress of many research works [5], [10], [11], [12], [13]. In the CTDE regime, agents decide their actions based only on their local information. This implies that the optimal joint actions are determined based on each agent's optimal action without global information, such as the interactions among individual agents. To account for direct interactions among agents, it is imperative to use global information, preferably somewhat limited, not only in training but also during execution while ensuring scalability.

Markov games [19], [20] are powerful frameworks for modeling cooperative MARL problems. Each agent selects actions based on its observations, which influence the state

and rewards of the entire system. Recent studies have applied Markov games to complex real-world problems, thereby highlighting their effectiveness. For example, reinforcement learning has been used within the Markov game framework to explore preferential cyber defense strategies for power grids, allowing agents to optimize their defense strategies in response to cyber threats [56]. Additionally, research on the efficiency of reinforcement learning in decentralized general-sum Markov games has provided theoretical guarantees for the convergence of optimal policies [57]. Furthermore, the potential of Markov games for critical infrastructure protection has been demonstrated by utilizing deep Q-learning to defend smart electrical power grids against cyberattacks [58]. In the field of UAV-assisted communication, a multi-agent reinforcement learning approach using policy clipping and average evaluation has been proposed to enhance the communication strategies [59]. Moreover, a Bayesian reinforcement learning approach in Markov games has been presented to achieve near-optimal policies through efficient exploration-exploitation trade-offs [60].

These studies underscore the versatility and power of Markov games in addressing a wide range of multi-agent reinforcement learning challenges, making them a foundational element in the development of sophisticated MARL systems. In [5], the authors propose the individual global max (IGM) condition, which computes the joint Q values for the optimal joint policy updates without information on other agents' actions or rewards in decentralized execution. It defines the optimal joint action space comprising the optimal actions of each agent. Several other algorithms based on the CTDE have been proposed to satisfy the IGM condition. QMIX is one of the best-known algorithms with excellent performance [6], [7], [8] in the StarCraft Multi-Agent Challenge (SMAC) [9], which is a suite of StarCraft II unit micromanagement [10]. It integrates the agent's Q functions nonlinearly into a joint Q function through a mixing network that adheres to the monotonicity constraints of the IGM condition. In addition, VDN [11] satisfies the IGM condition by linearly integrating the joint Q function with additive composition. However, satisfying the IGM condition has not been proven to induce cooperative movements among agents [12]. Although these algorithms exhibited strong performance in simple tasks such as controlling a small number of uniform units, the additivity or monotonicity conditions proposed to satisfy IGM limit the applicable tasks, leading to poor performance in complex tasks.

Recently, a multi-agent framework utilizing sequence models (SM) was proposed to incorporate interactions among agents [13]. In this work, a Multi-Agent Transformer (MAT) models the MARL problem as an SM, leveraging the transformer architecture based on the theorem of multi-agent advantage decomposition [14]. Specifically, each agent's local information based on local observations is disseminated such that high-level interrelationships representing agents' interactions are embedded within the MAT's encoder. This can help mitigate the difficulties of CTDE with respect to partial observability without compromising scalability.

MAT organizes agents as a sequence in an arbitrary decision order to be used as the input to a transformer [15]. However, it is controversial to use a sequence model to represent the interactions among agents, given that the agents are interdependent and the embedding of interrelationships is bi-directional [17]. Furthermore, inter-agent relationships that can be represented by a partially connected, asymmetrical, dynamically changing graph might not be efficiently modeled by the self-attention mechanism of the transformer architecture.

In military operations, for instance, various units, such as infantries, tanks, and drones, work together to perform a specific mission. Theoretically, this arrangement allows every unit to share all battlefield information, which seemingly enhances situational awareness. If agents are organized into a fully connected graph, the continuous information exchange required between all units may overload the entire system. This information overload can hinder the ability of each unit to identify crucial information and respond in a timely manner. More specifically, distant units not directly affected by a drone's observational data should process all the data unnecessarily, potentially slowing down decision-making processes and consequently degrading the overall efficiency. In contrast, GAT allows for a more precise and hierarchical representation of relationships by utilizing not only the observations of agents, but also the edges that directly represent the relationships between agents [18]. Consequently, graph-based modeling can provide more effective communication, resource allocation, and execution of operations.

In this study, we generalize the representation of a group of agents by using a graph instead of a sequence. We also propose to replacing the self-attention mechanism with a graph attention mechanism to allow for explicit graph representations within the transformer architecture. This approach offers substantial benefits for strategy and policy determination aimed at fostering advancements in the comprehension and enhancement of multi-agent reinforcement learning to optimize the performance of the overall system. We present the behavior and performance differences between traditional MAT and our MAT-GAT model by performing experiments and comparative analyses based on challenging StarCraft II maps [9].

II. BACKGROUNDS

A. MULTI-AGENT ADVANTAGE DECOMPOSITION

Multi-agent advantage decomposition is the primary theorem for modeling MARL problems as an SM model [14]. Within multi-agent environments, the direct application of traditional actor-critic methodologies encounters significant challenges [21]. Specifically, the actions of one agent invariably influence the rewards perceived by others, thus complicating the precise estimation of a policy's gradient. This intricacy

is intrinsically tied to the 'credit assignment' dilemma [22], [23], [44], which involves allocating rewards or penalties to specific agents and their corresponding actions. Two predominant strategies have been proposed to address this: the first considers only individual agents' local information, and the second employs a counterfactual baseline approach [14]. MAT integrates the significance of employing sequence models and adopts the counterfactual-based multi-agent advantage decomposition theorem to accurately represent the intricate interrelationships between agents.

The multi-agent state-action value function for agent $i_1, \ldots, i_k$ can be defined as

$$Q_\theta^{i_{1:k}}(o, \boldsymbol{a}^{(i_{1:k})}) \triangleq \mathbb{E}_{\boldsymbol{a}^{-i_{1:k}} \sim \pi_\theta^{-i_{1:k}}}[Q_\theta(s, \boldsymbol{a}^{i_{1:k}}, \boldsymbol{a}^{-i_{1:k}})], \quad (1)$$

where $-i_{1:k}$ is the set of all agents excluding $i_1, \ldots, i_k$, called a counterfactual. When $k = 0$, the multi-agent state-action value function is the same as the state-action value function and when $k = n$, multi-agent state-action value function is same as the original state-action value function. With the state-action value function of subset $i_{1:k}$ defined in Equation 3, the agents' contribution to the joint state-action value can be measured. Hence, the multi-agent advantage function is defined as:

$$A_\theta^{i_{1:k}}\left(o, \boldsymbol{a}^{j_{1:m}}, \boldsymbol{a}^{i_{1:k}}\right) \triangleq Q_\theta^{j_{1:m}, i_{1:k}}\left(o, \boldsymbol{a}^{j_{1:m}, i_{1:k}}\right) - Q_\theta^{j_{1:m}}(o, \boldsymbol{a}^{j_{1:m}}), \quad (2)$$

which is the advantage of agents $i_1, \ldots, i_k$ and their actions $a^{i_1}, \ldots, a^{i_k}$ when the actions $a^{j_1}, \ldots, a^{j_m}$ of agents $j_1, \ldots, j_m$ are given. Based on these equations, the multi-agent advantage decomposition is defined as:

$$A_\theta^{i_{1:n}}\left(\boldsymbol{o}, \boldsymbol{a}^{i_{1:n}}\right) = \sum_{m=1}^{n} A_\theta^{i_{1:m}}\left(\boldsymbol{o}, \boldsymbol{a}^{i_{1:m-1}}, a^{i_m}\right). \quad (3)$$

Addressing the MARL challenges through Equation 5 offers scalability analogous to, or potentially even superior to the CTDE approach, in which agents utilize their local information to linearly or non-linearly produce joint Q-values. Distinctively diverging from CTDE, incorporating the advantage function of agents from all prior sequences mitigates, the challenge of partial observability.

B. MULTI-AGENT TRANSFORMER (MAT)

Beyond addressing the previously mentioned challenges of scalability and partial observability, an SM offers flexibility in terms of the sequence length, allowing the integration of various agent types under a single solution [13]. Notably, the transformer, a state-of-the-art SM, has the advantage of concurrently updating multiple tokens, thus facilitating faster learning rates. The proposed MAT employs multi-agent advantage decomposition and adopts a transformer encoder-decoder architecture, to address the generic multi-agent joint policy update through a sequential policy search process. The encoder encodes high-level interrelationships that represent the interactions between agents $(i_1, \ldots, i_n)$. The objective function of the encoder that attempts to minimize the empirical Bellman error is

$$L_{Encoder}(\phi) = \frac{1}{Tn} \sum_{m=1}^{n} \sum_{t=0}^{T-1} [R(\mathbf{o}_t, \mathbf{a}_t) + \gamma V_{\bar{\phi}}\left(\hat{\mathbf{o}}_{t+1}^{i_m}\right)^2 - V_\phi(\hat{\mathbf{o}}_t^{i_m})], \quad (4)$$

where $\bar{\phi}$ is the parameter of the target network. The decoder computes attention through masked self-attention, so i_r^{th} and the i_j^{th} action heads, wherein $r < j$, only the latter utilizes the information from the actions of agents in the previous sequence for policy updates. The clipping PPO objective [24] of the decoder is

$$L_{Decoder}(\theta) = -\frac{1}{Tn} \sum_{m=1}^{n} \sum_{t=0}^{T-1} \min\left(\mathbf{r}_t^{i_m}(\theta) \hat{A}_t, \text{clip}\left(\mathbf{r}_t^{i_m}(\theta), 1 \pm \epsilon\right) \hat{A}_t\right), \quad (5)$$

and $\mathbf{r}_t^{i_m}(\theta)$ is defined as

$$\mathbf{r}_t^{i_m}(\theta) = \frac{\pi_\theta^{i_m}\left(\mathbf{a}_t^{i_m} \mid \hat{\mathbf{o}}_t^{i_{1:n}}, \hat{\mathbf{a}}_t^{i_{1:m-1}}\right)}{\pi_{\theta_{\text{old}}}^{i_m}\left(\mathbf{a}_t^{i_m} \mid \hat{\mathbf{o}}_t^{i_{1:n}}, \hat{\mathbf{a}}_t^{i_{1:m-1}}\right)}, \quad (6)$$

where $\hat{A}_t$ is an estimate of the joint advantage function computed with generalized advantage estimation (GAE) using the outputs $V_\phi\left(\hat{\mathbf{o}}_t^{i_1}\right), \ldots, V_\phi(\hat{\mathbf{o}}_t^{i_n})$ from the encoder [13]. At every time step, the actions of all agents were sequentially determined individually. The joint action space defined by the masked self-attention is $\sum_{i=1}^{n} |\mathcal{A}^i|$, while the conventional joint action space in MARL, which considers bidirectional interactions, is $\prod_{i=1}^{n} |\mathcal{A}^i|$. Consequently, the masked self-attention of the decoder plays a pivotal role in solving the scalability problem.

C. GRAPH ATTENTION NETWORKS (GAT)

Graph Neural Networks (GNN) [25], [26], [27], in their initial forms, provided a robust framework for incorporating neighborhood information into node representations. The common practice among these networks was to aggregate information from a node's neighbors. Most frequently, such aggregation was done using simple mean or sum pooling. While these pooling techniques are computationally efficient and straightforward, they come with a critical shortcoming: they do not discern the potential difference in significance among neighbors. Imagine a scenario where certain neighbors have a strong influence on a central node, while others have a minimal one. By merely summing or averaging, the nuances get lost, potentially leading to sub-optimal node representations.

Attention mechanisms [16], originally developed for and predominantly utilized in NLP tasks, have a unique selling proposition: they allow models to allocate varying importance weights to different inputs. This dynamism is starkly contrasted with the 'one-size-fits-all' approach of traditional GNNs. With attention, if certain words (or neighbors, in the

case of GNNs) are more pertinent to context (or node), they get higher importance, and vice versa.

GAT is an amalgamation of GNN's structure-sensitivity and attention's adaptive importance assignment. Rather than uniformly weighing neighbors, GAT lets each node decide the significance of its neighbors through learned attention weights. This is achieved using a shared self-attention mechanism across edges in the graph. So, for each node, GAT computes a coefficient for its neighbors, determining how much attention should be paid to each when updating its own features.

The essence of the GAT can be captured by its layer-wise propagation rule. Given a graph $G = (V, E)$, the feature update for a node in GAT is

$$h_i' = \sigma\left(\sum_{j\in\mathcal{N}(i)} \alpha_{ij} W h_j\right), \tag{7}$$

where h_j and h_i' are the features of node j and the updated feature of node i respectively; σ is the activation function; and $\mathcal{N}(i)$ is the neighbors of node i. The attention coefficient α_{ij} represents how much node i should attend to node j is computed as follows:

$$\alpha_{ij} = \frac{exp\left(LeakyReLU\left(\mathbf{a}^T\left[Wh_i||Wh_j\right]\right)\right)}{\sum_{k\in N_i} exp\left(LeakyReLU\left(\mathbf{a}^T\left[Wh_i||Wh_k\right]\right)\right)}, \tag{8}$$

where $\mathbf{a}$ is the shared weight vector of the attention mechanism.

III. MAT-GAT

In our work, we model fully cooperative multi-agents as a Markov game which is defined as a tuple $G =< \mathcal{N}, \boldsymbol{S}, \boldsymbol{\mathcal{A}}, P, R, \gamma >$. $\mathcal{N} = \{1, \ldots, n\}$ is the set of agents, $\boldsymbol{S} = \prod_{i=1}^{n} S^i$ and $\boldsymbol{\mathcal{A}} = \prod_{i=1}^{n} \mathcal{A}^i$ are the joint state space, which is the product of the agent's local state space and the product of the n agent's action space, respectively. $P : \boldsymbol{S} \times \boldsymbol{\mathcal{A}} \times \boldsymbol{S} \to \mathbb{R}$ is the transition probability function; $R : \boldsymbol{S} \times \boldsymbol{\mathcal{A}} \to \mathbb{R}$ is the joint reward function; and $\gamma \in [0, 1)$ is the discount factor. At time step $t \in \mathbb{T}$, agent $i \in \mathcal{N}$ is at state $s_t \in \boldsymbol{S}$. All agents take actions $a_t^1, \ldots, a_t^n$ simultaneously by using stochastic policy π_θ^i, where θ is the joint parameter. a_t^i and π_θ^i are the components of the joint action $\boldsymbol{a}_t = (a_t^1, \ldots, a_t^n) \in \boldsymbol{\mathcal{A}}$ and the joint policy $\boldsymbol{\pi_\theta}(\boldsymbol{a}_t \mid s_t) = \prod_{i=1}^{n} \pi_\theta^i\left(a_t^i \mid \mathrm{s}_t\right)$, respectively. The initial states are induced by the distribution ρ_π which is determined by the joint policy $\boldsymbol{\pi_\theta}$ and the transition probability function P. At the end of every time step, all agents receive a joint reward R and move to $\boldsymbol{S_{t+1}}$. They try to maximize their total expected cumulative return $R^\gamma \triangleq \sum_{t=0}^{\infty} \gamma^t R(\boldsymbol{o}_t, \boldsymbol{a}_t)$. The state value function V_θ and state-action function Q_θ are defined as follows:

$$V_\theta(s) \triangleq \mathbb{E}_{\boldsymbol{a}_0\sim\pi, \boldsymbol{a}_{1:\infty}\sim\pi_\theta, s_{1:\infty}\sim P}[R_0|s_0 = s], \tag{9}$$

$$Q_\theta(s, \boldsymbol{a}) \triangleq \mathbb{E}_{\boldsymbol{a}_{1:\infty}\sim\pi_\theta, s_{1:\infty}\sim P}, [R_0|s_0 = s, a_0 = a], \tag{10}$$

respectively.

As stated in [13], MAT addresses Markov game problems by applying sequence modeling. Similarly, the MAT-GAT architecture introduced in this paper is an enhanced version of MAT that takes a set of agents' observations in a predefined order as input and uses the encoder-decoder architecture of the transformer to convert it into latent representations, which are then used to generate a set of actions for the agents in the same order. Specifically, MAT-GAT utilizes the multi-agent advantage decomposition theorem to transform joint policy optimization into an ordered decision-making process. This allows each agent to understand how its actions impact the overall team's performance and to make optimal decisions while considering the actions of other agents. In what follows, we begin our discussion describing by graph-based modeling and GAT which motivated our study. Next, the architecture of the MAT-GAT model is presented.

A. GRAPH REPRESENTATION IN MARL

Decision-making in a multi-agent environment is a highly complex problem requiring precise methods to depict the relationships and interactions among agents for effective modeling and interpretation. In this section, we highlight the pivotal roles of graph-based modeling techniques and GAT in addressing these requirements.

In multi-agent settings, each agent's actions influence others and the system as a whole [48]. Graph structures serve as a powerful means of visually and mathematically representing complex interrelationships between agents, expressing various connections and strengths that are difficult to convey through sequential or flat data architectures [49].

A unidirectional relationship can be established if a predefined order exists between agents. Although an SM can be utilized for modeling an MARL problem as in [13], only a few multi-agent problems can be modeled based on an ordered sequence of agents. In a cooperative multi-agent reinforcement learning scenario, the actions of each agent can affect the behavior of other agents according to a predefined relationship. We propose using graph modeling as an alternative method to represent interdependencies and interactions better than sequential modeling [26], [54].

Although transformers can encode sequential data, they often require hierarchical encoding for tasks involving layered structures. Hierarchical encoding involves organizing data into multiple layers of abstraction, which can be complicated or unnecessary [55]. For example, in natural language processing, this might involve encoding data at the word, sentence, and paragraph levels successively. Traditional transformers apply self-attention mechanisms across these layers, processing all inputs simultaneously and attending to all parts of the data. The self-attention mechanism calculates the attention scores between every pair of tokens in the input sequence, resulting in an attention matrix that grows quadratically with sequence length. This comprehensive approach, although powerful, can be computationally expensive and sometimes less focused when dealing with structured data.

GAT can focus only on the relevant connections between nodes, which represent data points. Because graph structures

explicitly define the relationships between nodes, the specific organization of multiple agents, such as the hierarchies of team players, can be represented better. Moreover, GAT computes attention coefficients only for neighboring nodes, resulting in a computational complexity of $O(NE)$, in comparison to self-attention, which has a computational complexity of $O(N^2)$. Here, N is the number of agents and E is the number of edges, representing the number of connections each agent has with other agents. Because E is always less than or equal to N this significantly reduces computational overhead. Such a restricted attention mechanism allows for more efficient information propagation and representation learning, thereby effectively reducing the need for multiple processing layers. Therefore, graph modeling with GAT not only provides an explicit representation of relationships, but also simplifies the architecture, making it a more effective solution for multi-agent reinforcement learning [47].

We employed GAT to elucidate the high-level interrelations among graph-modelled agents [53]. This involves dynamically modulating the importance and direction of agent interactions through attention scores computed for adjacent agents, denoted as:

$$e_{ij} = \boldsymbol{a}(Wh_i, Wh_j). \tag{11}$$

GAT adeptly captures and interprets the intricate communication patterns and dependencies prevalent in multi-agent environments by using structural data of graphs, such that decisions are made based on information from interconnected agents [29], [47]. GAT' capability of GAT to assess the relevance of specific agents in a given context is widely accepted to surpass that of conventional neural networks [18], [50], [51], [52].

B. MAT-GAT STRUCTURE

After the initial success of the transformer model, researchers explored various modifications and extensions of the attention mechanism to optimize models for a wide range of problem domains and tasks. For instance, to adapt transformers not only for sequential data but also for diverse data types, such as images or graphs, modifications such as spatial attention and axial attention have been introduced [30], [31]. Such adaptations have allowed models to capture structural and spatial relationships better. Furthermore, domain-specific attention mechanisms have been proposed for tasks that require reflection of critical data features or structures. Examples include attention mechanisms that account for dependency relationships or syntactic information in natural language processing and those tailored for molecular structures [32], [33], [34]. There has also been a surge in studies focusing on reducing the computational complexity of self-attention for efficiency, as well as integrating multiple attention mechanisms for handling multi-modal data [35], [36], [37], [38]. Such advances in research validate the flexibility of transformers and showcase their potential for constructing more suitable and effective models for a multitude of problems and datasets.

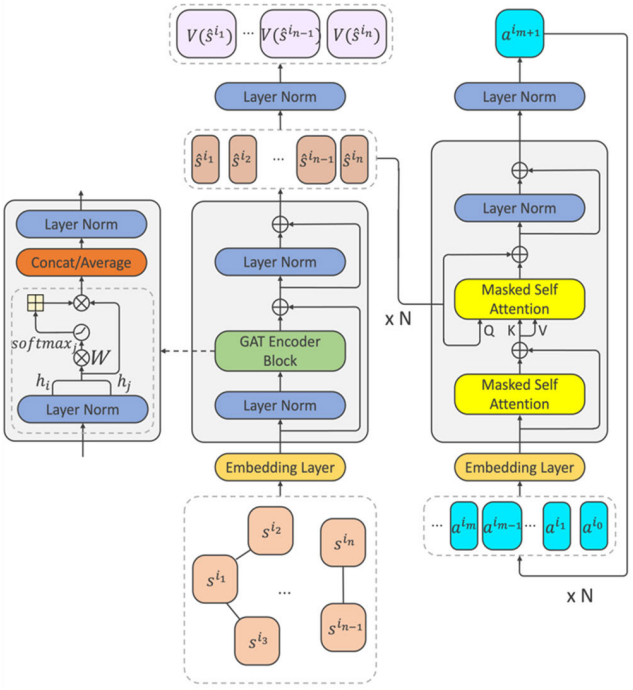

FIGURE 1. The MAT-GAT architecture.

As mentioned above, we believe that utilizing graph-based methodologies and GAT will enhance the accuracy and efficiency of modeling complex interdependencies and interactions among agents in multi-agent reinforcement learning challenges.

In our study, rather than altering the structural design of the transformer, we modified self-attention to adopt the GAT structure, as illustrated in Figure 1. In other words, GAT blocks are adopted to substitute the self-attention layers in the encoder of the basic transformer. We call these GAT encoder blocks. The GAT encoder block in Figure 1 calculates attention scores and updates features using Equations (9) and (10), where h_i and h_j are the state features of agent i and j respectively. The updated features h_i' are used to encode the high-level interrelationships of the agent's team in the states. In the transformer, self-attention grants distinct weights to each observation, ensuring that the output representations are discernible, even when a team is composed of heterogeneous agents. However, graph-based modeling and GATs differentiate diverse or complex relationships among heterogeneous agents, allowing each agent to assimilate the most pertinent information from its neighbors. This results in agents being embedded with complex interactions that are better represented by the encoder. The decoder, which receives state information embedded with detailed and differentiated interaction details facilitated by GATs, can make more informed auto-regressive action decisions through masked self-attention [13]. Consequently, the GAT encoder block ensures efficient task performance, even in scenarios requiring hierarchical neighbor information.

In our framework, agents are organized into graphs that encompass sequence information. While [13] arranges agents sequentially, the actual state information of the agents may not be inherently sequential, hence positional encoding is not applied. In contrast, the agents are treated sequentially in the MAT decoder. Therefore, the decoder's masked self-attention is retained in its original form, without transitioning to the GAT approach. Additionally, the objective functions of the encoder and decoder, which are Equations (6), (7) and (8), were consistently applied, mirroring the conventional MAT methodology.

IV. EXPERIMENTS

We conducted experiments using SMAC [9] and MuJoCo [45], [46] which are two of the most representative benchmarks in the MARL environment [13].

A. SMAC EXPERIMENTS RESULTS

Experiments were conducted using cooperative challenging StarCraft II maps within SMAC. The SMAC consists of a collection of StarCraft II micro-scenarios designed to evaluate methods for training independent agents to cooperate in solving complex tasks. Initial placements, numbers, and types of units vary across scenarios, as do the terrains, which may or may not be traversable. Allied units are controlled by trained allied agents, whereas enemy units are managed by the built-in game AI using untrained heuristics. Each episode begins with the game AI directing units to engage the allied agents using scripted strategies. An episode concludes when all units on either the allied or enemy side are eliminated or when a specified max time step is reached, with the latter resulting in a defeat for the allied agents. The goal of learning is to maximize the win rate. Maps are categorized into easy, hard, and super hard difficulty levels. In this study, performance analysis and comparative experiments between MAT and MAT-GAT were performed on easy and hard tasks. Additionally, for super hard tasks, we benchmark against the most representative and state-of-the-art algorithms, QMIX, to compare with the MAT, MAT-GAT, and CTDE approaches. Table 1 lists the units used for each task. All units are from StarCraft II, and the name of each task was assigned based on the composition of the ally and enemy units. The units are composed of various roles, such as ranged attack units like Marines, Stalkers, Colossi, Marauders, and Hydralisks; melee attack units such as Zealots and Zerglings; and healing units like Medivacs. Each task was categorized into different levels of difficulty according to how the units counter each other.

TABLE 1. Unit composition for SMAC challenges.

Difficulty	Name	Ally Units	Enemy Units
Easy	3m	3 Marines	3 Marines
	8m	8 Marines	8 Marines
	1c3s5z	1 Colossus, 3 Stalkers, 5 Zealots	1 Colossus, 3 Stalkers, 5 Zealots
	MMM	1 Medivac, 2 Marauders, 7 Marines	1 Medivac, 2 Marauders, 7 Marines
Hard	2c_vs_64zg	2 Colossi	64 Zerglings
	5m_vs_6m	5 Marines	6 Marines
	8m_vs_9m	8 Marines	9 Marines
	10m_vs_11m	10 Marines	11 Marines
Super hard	27m_vs_30m	27 Marines	30 Marines
	6h_vs_8z	6 Hydralisks	8 Zealots
	MMM2	1 Medivac, 2 Marauders, 7 Marines	1 Medivac, 3 Marauders, 8 Marines
	3s5z_vs_3s6z	3 Stalkers, 5 Zealots	3 Stalkers, 6 Zealots

In easy tasks, which involve combat between identical units, it is observable from Figure 2 that a peak win rate of 100% is achieved early in the learning process. Unlike the 3m and 8m scenarios shown in Figure 2, the 1c3s5z and MMM tasks feature a diverse unit composition, with both MAT and MAT-GAT exhibiting high performance. For hard tasks, which require more complex tactics, such as focus firing or utilizing terrain due to the allied units being fewer in number compared to the enemy, agents are not presented with varied unit combinations. Task 2c_vs_64zg in Figure 3 involves controlling two Colossi, which have greater mobility over terrain features such as hills compared to Zerglings. These two Colossi require a closer interrelationship due to their terrain navigation capabilities, and MAT-GAT demonstrates superior performance in the latter stages of training

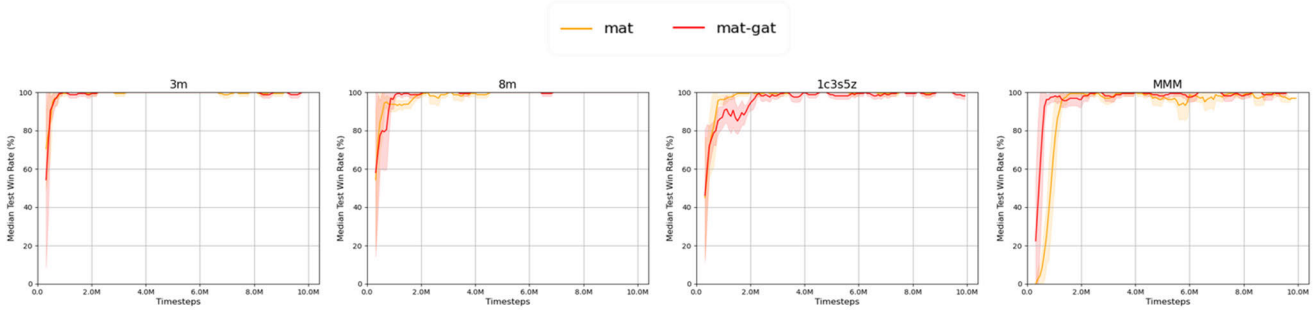

FIGURE 2. Experimental results of SMAC's easy tasks for performance comparison between MAT and MAT-GAT.

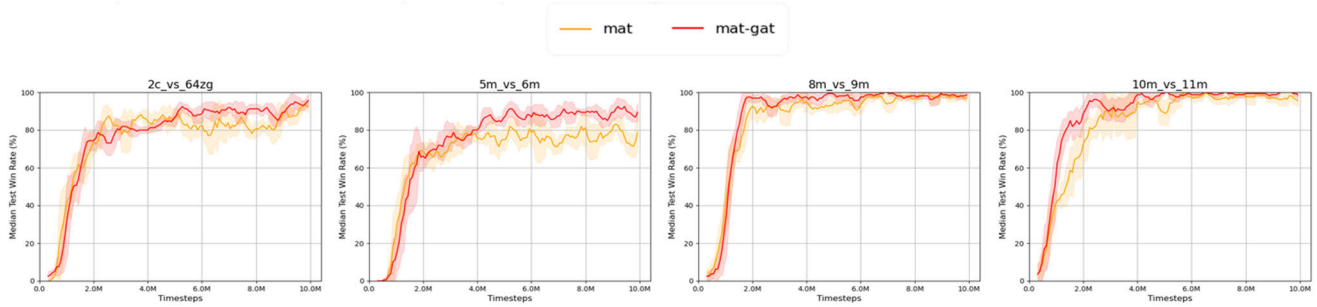

FIGURE 3. Experimental results of SMAC's hard tasks for performance comparison between MAT and MAT-GAT.

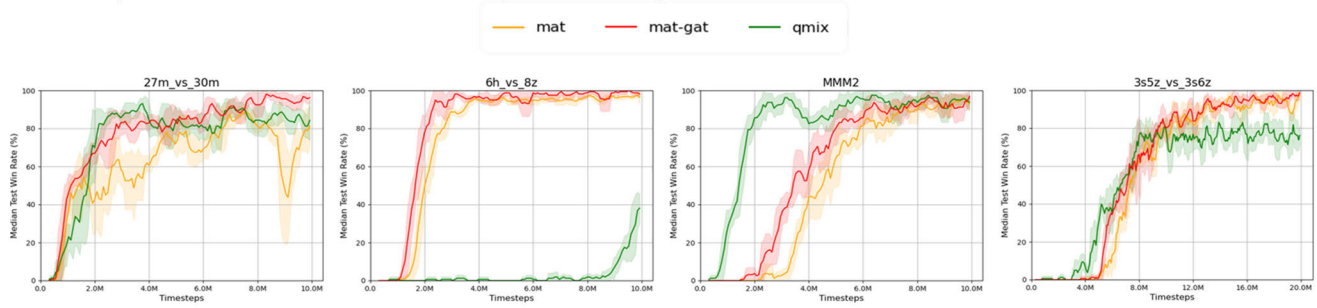

FIGURE 4. Experimental results of SMAC's super hard tasks for performance comparison among MAT, MAT-GAT, and QMIX.

for this task compared to MAT. The other three tasks consist of single-unit compositions where the allied units are outnumbered, necessitating focus firing and a keen consideration of allies' current states. MAT-GAT outperformed MAT with faster convergence and higher peak win rates, suggesting the advantages of GAT over self-attention in modeling interrelationships among agents. This is evidenced by the higher peak win rates for MAT-GAT observed in the 5m_vs_6m task in Table 2.

In the category designated as ''super hard'' tasks, the agents are given challenges that require controlling a larger number of units, facing off against enemy units with advantageous unit compositions and superior numbers, as well as more collaborative tactics and micro-management. It is hypothesized that in these more complex scenarios, the graph-based representation would offer a more advantageous portrayal of the interactions between agents. The results, as depicted in Figure 4, confirm this hypothesis. In the 27m_vs_30m task, MAT-GAT exhibits more stable learning behavior compared to MAT. This scenario, characterized by a larger number of units leading to the formation of small groups, benefits from the connectivity of graph-based modeling. This feature allows for the effective segmentation into smaller groups due to the explicit representation of graph connections, enabling focused detection of significant interactions within and between these groups and individual assessment of each unit's importance. In contrast, sequence modeling, due to its inability to effectively segregate into smaller groups because of its linear and

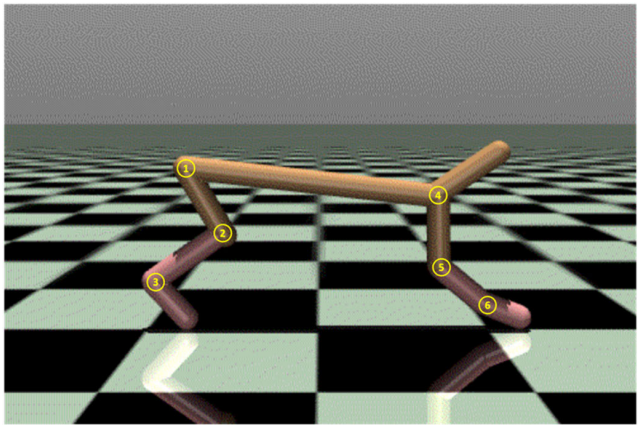

FIGURE 5. MuJoCo Half-cheetah joint structure; 1: back-thigh, 2: back-shin, 3: back-foot, 4: front-thigh, 5: front-shin, 6: front-foot.

interconnected approach, fails to distinctly allocate attention among the groups due to the large number of units, resulting in a less effective detection of significant interactions and, consequently, a relatively lower win rate. The outcomes for the 6h_vs_8z, MMM2, and 3s5z_vs_3s6z tasks exhibit similar trends, with MAT-GAT approaching convergence more rapidly than MAT, and higher final win rates for MAT-GAT as evidenced by Table 2. The 6h_vs_8z task, where kiting of units is crucial, shows that MAT-GAT tend to alternate attention promptly between attacking and retreating units. In the MMM2 and 3s5z_vs_3s6z tasks, which involve teams composed of multiple unit types with varying roles, GAT's

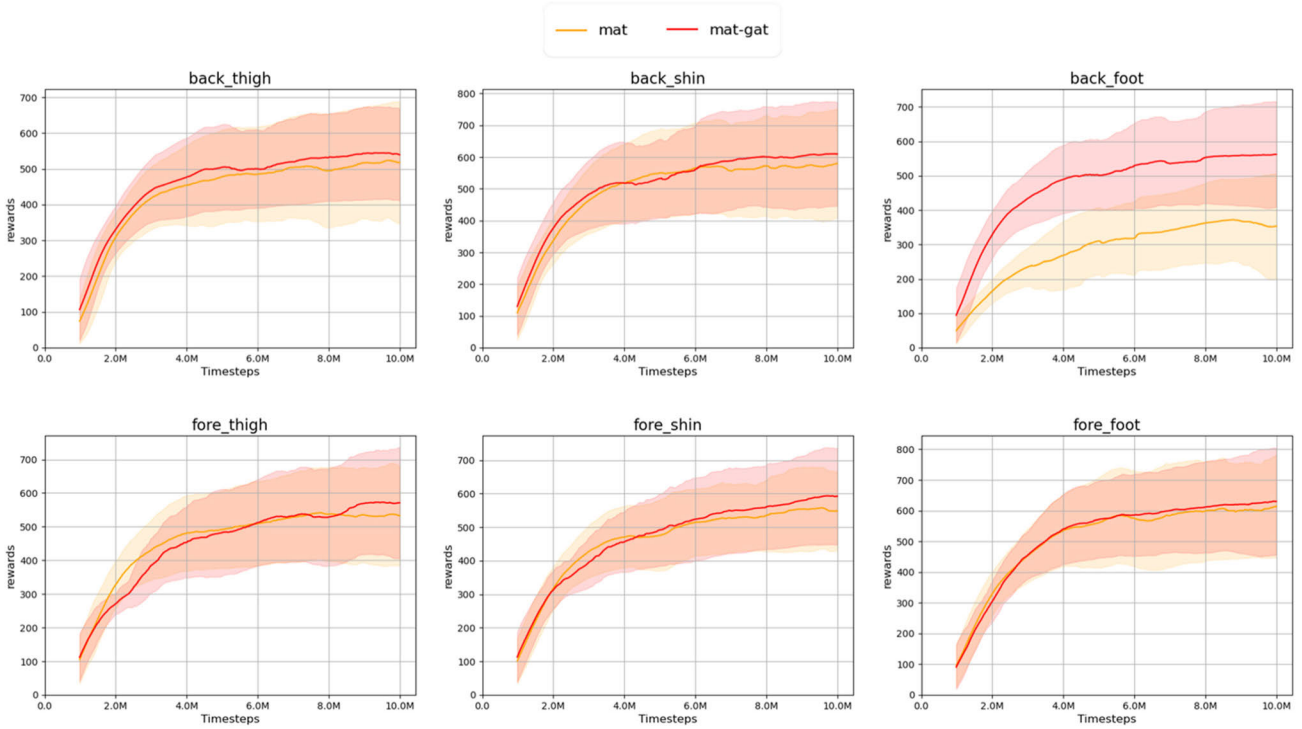

FIGURE 6. Inductive experiment results for verifying the adaptability of MAT and MAT-GAT in Half-cheetah; joint names specified above each graph are disabled during training.

dynamic allocation of importance based on each unit's role and the situational context enable a strategic advantage even in disadvantageous situations. In the MMM2 task, QMIX reaches the highest win rate in the shortest time thanks to its simple state-action value decomposition which offers greater training stability in complex tasks. However, QMIX does not converge stably, as indicated by a sharp drop-in win rate after reaching the peak. On the other hand, both MAT and MAT-GAT show a stable increase in win rate, with MAT-GAT reaching the highest win rate faster than MAT. MAT-GAT exhibits a consistently monotonic increase across the super hard tasks, in contrast to QMIX, which show varying patterns across different tasks. In the 27m_vs_30m and 3s5z_vs_3s6z tasks, MAT-GAT achieves peak win rates more slowly than QMIX, but ultimately recorded higher final win rates by the end of training. In the 6h_vs_8z task, QMIX exhibits almost no learning, whereas in the MMM2 task, it achieves the fastest convergence and the highest final win rate amongst the three algorithms.

B. MuJoCo EXPERIMENTS RESULTS

The Half-Cheetah task in the MuJoCo physics simulator serves as a critical benchmark for continuous control and reinforcement learning. This task is centered on a biomechanically inspired robotic cheetah designed to emulate the musculoskeletal structure of its biological counterpart. The objective was to propel the cheetah forward as quickly as possible without falling over, thus maximizing the distance covered over a fixed time horizon.

TABLE 2. SMAC experiments results.

Difficulty	Name	MAT-GAT	MAT	QMIX	Steps
Easy	3m	**100**	**100**	/	1e7
	8m	**100**	**100**	/	1e7
	1c3s5z	**100**	**100**	/	1e7
	MMM	**100**	**100**	/	1e7
Hard	2c_vs_64zg	**100**	98.8	/	1e7
	5m_vs_6m	**96.5**	90.6	/	1e7
	8m_vs_9m	**100**	**100**	/	1e7
	10m_vs_11m	**100**	**100**	/	1e7
Super hard	27m_vs_30m	**100**	96.5	96.5	1e7
	6h_vs_8z	**100**	**100**	46.9	1e7
	MMM2	**100**	93.8	**100**	1e7
	3s5z_vs_3s6z	**100**	96.5	90.6	2e7

In this task, the agents representing the half-cheetah must learn to coordinate the actions of their joints to produce effective locomotion. The state space includes the positions, velocities, and angular velocities of the robot body and joints, whereas the action space consists of torques applied to the joints. The complexity of this task arises from the high dimensionality of the state space and the need for precise real-time control to maintain balance and forward momentum.

GAT is employed to indirectly learn the global graph structure using only local neighborhood information [18]. This attribute makes GAT highly effective in modeling complex graph structures in inductive tasks [28]. To assess the adaptability and generalization capability of MAT-GAT, we conduct an experiment using the MuJoCo Half-Cheetah

task. In this experiment, each of the six joints, as seen in Figure 5, is disabled one at a time during training, followed by testing with all joints enabled.

As depicted in Figure 6, MAT-GAT's final reward slightly exceeds that of MAT in six out of six tasks. Notably, MAT-GAT demonstrate minimal variance in rewards across all tasks. The average final reward for MAT-GAT is 586, with a standard deviation of 26.82. In contrast, MAT's average final reward is 524, with a significantly higher standard deviation of 93.47. Specifically, in the back_foot task, final reward of MAT drops to 346, which is a 33.96% decrease compared to the average of other tasks. These results indicate that MAT-GAT is more adaptive than MAT, consistently delivering stable and superior performance in diverse environments. Additionally, Table 3 presents not only the average (Avg) and standard deviation (SD) for MAT and MAT-GAT but also their coefficient of variation (CV) and standard error (SE). MAT-GAT also exhibited superior outcomes in terms of the CV and SE.

TABLE 3. MuJoCo half-cheetah experiments results.

Name	Tasks	Final Rewards	Avg	SD	CV	SE
MAT-GAT	back_thigh	**548**	**586**	**26.82**	**0.046**	**10.95**
	back_shin	**605**				
	back_foot	**565**				
	fore_thigh	**581**				
	fore_shin	**599**				
	fore_foot	**620**				
MAT	back_thigh	522	524	93.47	0.178	38.16
	back_shin	589				
	back_foot	346				
	fore_thigh	530				
	fore_shin	550				
	fore_foot	608				

V. CONCLUSION

In this study, we explored the benefits of graph-based modeling to represent interactions among agents in MARL environments. To this end, we introduced MAT-GAT, a transformer-based model that transitions from the original MAT sequence model to graph-based modeling, replacing the conventional self-attention mechanism with GAT to enhance the representation of interactions. We conducted a theoretical analysis and empirical experiments on the application of graph-based modeling and GAT, effectively representing interactions among agents across a variety of MARL tasks. Empirical evaluations are performed using representative tasks of varying difficulty from the SMAC and MuJoCo Half-cheetah, comparing the outcomes of transformers with self-attention and GAT implementations. Future research will not only apply graph-based modeling without including sequence information, but will also focus on structuring agent teams as complete graphs, thereby robustly accounting for inter-agent relationships in MARL problems. Specifically, it is recommended to investigate the integration of advanced GAT variants and explore hybrid models that combine sequence and graph-based methods to address dynamic and heterogeneous MARL scenarios. Although there have been numerous attempts to solve MARL problems with graphs [39], [40], [41], [42], [43], the integration of a transformer-based model with graph-based modeling is anticipated to mark a new phase in this field.

REFERENCES

[1] M. Hüttenrauch, A. Sočić, and G. Neumann, "Guided deep reinforcement learning for swarm systems," 2017, *arXiv:1709.06011*.
[2] Y. Cao, W. Yu, W. Ren, and G. Chen, "An overview of recent progress in the study of distributed multi-agent coordination," *IEEE Trans. Ind. Informat.*, vol. 9, no. 1, pp. 427–438, Feb. 2013.
[3] F. A. Oliehoek, M. T. J. Spaan, and N. Vlassis, "Optimal and approximate Q-value functions for decentralized POMDPs," *J. Artif. Intell. Res.*, vol. 32, pp. 289–353, May 2008.
[4] L. Kraemer and B. Banerjee, "Multi-agent reinforcement learning as a rehearsal for decentralized planning," *Neurocomputing*, vol. 190, pp. 82–94, May 2016.
[5] K. Son, D. Kim, W. J. Kang, D. E. Hostallero, and Y. Yi, "QTRAN: Learning to factorize with transformation for cooperative multi-agent reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 5887–5896.
[6] T. Wang, J. Wang, C. Zheng, and C. Zhang, "Learning nearly decomposable value functions via communication minimization," 2019, *arXiv:1910.05366*.
[7] K. Zhang, Z. Yang, and T. Başar, "Multi-agent reinforcement learning: A selective overview of theories and algorithms," in *Handbook of Reinforcement Learning and Control*. Springer, 2021, pp. 321–384.
[8] J. Hu, S. Jiang, S. A. Harding, H. Wu, and S.-W. Liao, "Rethinking the implementation tricks and monotonicity constraint in cooperative multi-agent reinforcement learning," 2021, *arXiv:2102.03479*.
[9] M. Samvelyan, T. Rashid, C. S. de Witt, G. Farquhar, N. Nardelli, T. G. J. Rudner, C.-M. Hung, P. H. S. Torr, J. Foerster, and S. Whiteson, "The StarCraft multi-agent challenge," 2019, *arXiv:1902.04043*.
[10] T. Rashid, M. Samvelyan, C. S. De Witt, G. Farquhar, J. Foerster, and S. Whiteson, "Monotonic value function factorisation for deep multi-agent reinforcement learning," *J. Mach. Learn. Res.*, vol. 21, no. 1, pp. 7234–7284, 2020.
[11] P. Sunehag, G. Lever, A. Gruslys, W. M. Czarnecki, V. Zambaldi, M. Jaderberg, M. Lanctot, N. Sonnerat, J. Z. Leibo, K. Tuyls, and T. Graepel, "Value-decomposition networks for cooperative multi-agent learning," 2017, *arXiv:1706.05296*.
[12] A. Mahajan, T. Rashid, M. Samvelyan, and S. Whiteson, "Maven: Multi-agent variational exploration," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019, pp. 1–12.
[13] M. Wen, J. Kuba, R. Lin, W. Zhang, Y. Wen, J. Wang, and Y. Yang, "Multi-agent reinforcement learning is a sequence modeling problem," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 35, 2022, pp. 16509–16521.
[14] J. G. Kuba, M. Wen, L. Meng, H. Zhang, D. Mguni, J. Wang, and Y. Yang, "Settling the variance of multi-agent policy gradients," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, 2021, pp. 13458–13470.
[15] A. Vaswani, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1–11.
[16] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," 2014, *arXiv:1409.0473*.
[17] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," 2018, *arXiv:1810.04805*.
[18] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," 2017, *arXiv:1710.10903*.
[19] M. L. Littman, "Markov games as a framework for multi-agent reinforcement learning," in *Machine Learning Proceedings*. San Mateo, CA, USA: Morgan Kaufmann, 1994. 157-163.
[20] L. E. Zachrisson, *Markov Games*, vol. 52. Princeton, NJ, USA: Princeton Univ. Press, 1964.
[21] R. Lowe, Y. I. Wu, A. Tamar, J. Harb, O. A. I. P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1–12.

[22] J. Foerster, G. Farquhar, T. Afouras, N. Nardelli, and S. Whiteson, "Counterfactual multi-agent policy gradients," in *Proc. AAAI Conf. Artif. Intell.*, 2018, vol. 32, no. 1, pp. 1–9.

[23] A. Cohen, E. Teng, V.-P. Berges, R.-P. Dong, H. Henry, M. Mattar, A. Zook, and S. Ganguly, "On the use and misuse of absorbing states in multi-agent reinforcement learning," 2021, *arXiv:2111.05992*.

[24] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017, *arXiv:1707.06347*.

[25] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Trans. Neural Netw.*, vol. 20, no. 1, pp. 61–80, Jan. 2009.

[26] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun, "Graph neural networks: A review of methods and applications," *AI Open*, vol. 1, pp. 57–81, Jan. 2020.

[27] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?" 2018, *arXiv:1810.00826*.

[28] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1–11.

[29] J. B. Lee, R. A. Rossi, S. Kim, N. K. Ahmed, and E. Koh, "Attention models in graphs: A survey," *ACM Trans. Knowl. Discovery Data*, vol. 13, no. 6, pp. 1–25, Dec. 2019.

[30] N. Parmar, A. Vaswani, J. Uszkoreit, L. Kaiser, N. Shazeer, A. Ku, and D. Tran, "Image transformer," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 4055–4064.

[31] J. Ho, N. Kalchbrenner, D. Weissenborn, and T. Salimans, "Axial attention in multidimensional transformers," 2019, *arXiv:1912.12180*.

[32] S. Vashishth, N. Yadati, and P. Talukdar, "Graph-based deep learning in natural language processing," in *Proc. 7th ACM IKDD CoDS 25th COMAD*, Jan. 2020, pp. 371–372.

[33] A. Cohan, W. Ammar, M. van Zuylen, and F. Cady, "Structural scaffolds for citation intent classification in scientific publications," 2019, *arXiv:1904.01608*.

[34] Ł. Maziarka, T. Danel, S. Mucha, K. Rataj, J. Tabor, and S. Jastrzebski, "Molecule attention transformer," 2020, *arXiv:2002.08264*.

[35] I. Beltagy, M. E. Peters, and A. Cohan, "Longformer: The long-document transformer," 2020, *arXiv:2004.05150*.

[36] N. Kitaev, Ł. Kaiser, and A. Levskaya, "Reformer: The efficient transformer," 2020, *arXiv:2001.04451*.

[37] J. Lu, D. Batra, D. Parikh, and S. Lee, "ViLBERT: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019, pp. 1–11.

[38] L. Harold Li, M. Yatskar, D. Yin, C.-J. Hsieh, and K.-W. Chang, "VisualBERT: A simple and performant baseline for vision and language," 2019, *arXiv:1908.03557*.

[39] Y. Liu, W. Wang, Y. Hu, J. Hao, X. Chen, and Y. Gao, "Multi-agent game abstraction via graph attention neural network," in *Proc. Conf. Artif. Intell.*, 2020, vol. 34, no. 5, pp. 7211–7218.

[40] A. Malysheva, T. T. Sung, C.-B. Sohn, D. Kudenko, and A. Shpilman, "Deep multi-agent reinforcement learning with relevance graphs," 2018, *arXiv:1811.12557*.

[41] L. Kuyer, S. Whiteson, B. Bakker, and N. Vlassis, "Multiagent reinforcement learning for urban traffic control using coordination graphs," in *Machine Learning and Knowledge Discovery in Databases*. Berlin, Germany: Springer, Sep. 2008.

[42] P. Zheng, L. Xia, C. Li, X. Li, and B. Liu, "Towards self-X cognitive manufacturing network: An industrial knowledge graph-based multi-agent reinforcement learning approach," *J. Manuf. Syst.*, vol. 61, pp. 16–26, Oct. 2021.

[43] S. Chen, J. Dong, P. J. Ha, Y. Li, and S. Labi, "Graph neural network and reinforcement learning for multi-agent cooperative control of connected autonomous vehicles," *Computer-Aided Civil Infrastructure Eng.*, vol. 36, no. 7, pp. 838–857, Jul. 2021.

[44] J. G. Kuba, R. Chen, M. Wen, Y. Wen, F. Sun, J. Wang, and Y. Yang, "Trust region policy optimisation in multi-agent reinforcement learning," 2021, *arXiv:2109.11251*.

[45] B. Peng, T. Rashid, C. A. S. de Witt, P.-A. Kamienny, P. H. S. Torr, W. Böhmer, and S. Whiteson, "FACMAC: Factored multi-agent centralised policy gradients," 2020, *arXiv:2003.06709*.

[46] E. Todorov, T. Erez, and Y. Tassa, "MuJoCo: A physics engine for model-based control," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2012, pp. 5026–5033.

[47] Y. Shao, R. Li, B. Hu, Y. Wu, Z. Zhao, and H. Zhang, "Graph attention network-based multi-agent reinforcement learning for slicing resource management in dense cellular network," *IEEE Trans. Veh. Technol.*, vol. 70, no. 10, pp. 10792–10803, Oct. 2021.

[48] E. Khalil, H. Dai, Y. Zhang, B. Dilkina, and L. Song, "Learning combinatorial optimization algorithms over graphs," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp 1–11.

[49] 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, no. 5, pp. 3215–3238, Jun. 2021.

[50] S. Ma, J.-W. Liu, X. Zuo, and W.-M. Li, "Heterogeneous graph gated attention network," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2021, pp. 1–6.

[51] M. Wang, D. Zheng, Z. Ye, Q. Gan, M. Li, X. Song, J. Zhou, C. Ma, L. Yu, Y. Gai, T. Xiao, T. He, G. Karypis, J. Li, and Z. Zhang, "Deep graph library: A graph-centric, highly-performant package for graph neural networks," 2019, *arXiv:1909.01315*.

[52] T. Ohta, K. Tanaka, and R. Yamamoto, "Scene graph descriptors for visual place classification from noisy scene data," *ICT Exp.*, vol. 9, no. 6, pp. 995–1000, Dec. 2023.

[53] W. Du, S. Ding, C. Zhang, and Z. Shi, "Multiagent reinforcement learning with heterogeneous graph attention network," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 10, pp. 6851–6860, Oct. 2023.

[54] 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.

[55] W. Park, W. Chang, D. Lee, J. Kim, and S.-W. Hwang, "GRPE: Relative positional encoding for graph transformer," 2022, *arXiv:2201.12787*.

[56] M. Moradi, Y. Weng, J. Dirkman, and Y.-C. Lai, "Preferential cyber defense for power grids," *PRX Energy*, vol. 2, no. 4, Oct. 2023, Art. no. 043007.

[57] W. Mao and T. Basar, "Provably efficient reinforcement learning in decentralized general-sum Markov games," *Dyn. Games Appl.*, vol. 13, no. 1, pp. 165–186, Jan. 2022.

[58] M. Moradi, Y. Weng, and Y.-C. Lai, "Defending smart electrical power grids against cyberattacks with deep Q-learning," *PRX Energy*, vol. 1, no. 3, Nov. 2022, Art. no. 033005.

[59] Z. Feng, M. Huang, D. Wu, E. Q. Wu, and C. Yuen, "Multi-agent reinforcement learning with policy clipping and average evaluation for UAV-assisted communication Markov game," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 12, pp. 14281–14293, Dec. 2023.

[60] J. B. Clempner, "A Bayesian reinforcement learning approach in Markov games for computing near-optimal policies," *Ann. Math. Artif. Intell.*, vol. 91, no. 5, pp. 675–690, Oct. 2023.

WOOBEEN JIN was born in Jeju, South Korea, in 1996. He received the M.S. degree from the Department of Computer Engineering, Kwangwoon University, Seoul, South Korea, in 2024. His research interests include deep learning and multi-agent reinforcement learning.

HYUKJOON LEE was born in Seoul, South Korea, in 1963. He received the B.S. degree in computer science from the University of Michigan, Ann Arbor, MI, USA, in 1987, and the M.S. and Ph.D. degrees in computer and information science from Syracuse University, in 1989 and 1993, respectively. From 1994 to 1996, he worked for Samsung Electronics as a Senior Researcher. In 1996, he joined as a Faculty Member with the Department of Computer Engineering, Kwangwoon University. His research interests include machine learning, mobile and wireless networks, ITS, and smart healthcare systems.