Journal of Artificial Intelligence and Data Mining (JAIDM), Vol. 14, No. 4, 2026, 587-607.

Shahrood University of Technology

Journal of Artificial Intelligence and Data Mining (JAIDM)
Journal homepage: http://jad.shahroodut.ac.ir

Research paper

Graph-Transformer Reinforcement Learning for Scalable Multi-Agent Coordination

Sajad Bastami[1] and Mohammad Bagher Dowlatshahi[2] and Rojyar Pirmohammadian[1*] and Seyedeh Zahra Mousavi[2]

1. Computer Engineering Department, Kurdistan University, Sanandaj, Iran.
2. Computer Engineering Department, Lorestan University, Khorramabad, Iran.

Article Info

Article History:
Received 24 April 2026
Revised 14 July 2026
Accepted 24 July 2026

DOI:10.22044/jadm.2026.17636.2918

Keywords:
Multi-Agent Reinforcement Learning, Graph Neural Networks, Transformer Attention Mechanisms, Adaptive Coordination, Scalable Multi-Agent Systems.

**Corresponding author: r.pirmohamadiani@uok.ac.ir (R. Pirmohammadian).*

Abstract

Multi-**a**gent **r**einforcement **l**earning (MARL) is a key paradigm for coordination in robotics, autonomous systems, and distributed control. However, existing MARL methods face fundamental limitations in scalability, adaptability to dynamic environments, and stability under evolving interactions. To address these challenges, we propose Adaptive Graph-Transformer Reinforcement Learning (AGTRL), a framework integrating graph-based relational modelling with transformer attention for adaptive coordination in large-scale multi-agent systems. AGTRL unifies graph-based perception and attention-based coordination in an end-to-end pipeline, encoding role information and adaptively weighting interactions by context. This combination, missing in prior MARL methods, bridges scalability and robustness in dynamic environments. AGTRL constructs a dynamic graph of agent relationships and uses multi-head self-attention to prioritize relevant interactions in real-time, ensuring robust performance under perturbations. We evaluate robustness under communication dropout (up to 40% link removal) and dynamic edge removal, measuring performance via episode reward and win rate. The framework incorporates an adaptive stability-performance trade-off mechanism that maintains learning efficacy in the presence of communication constraints and environmental uncertainty. We introduce a graph-enhanced policy architecture that jointly optimizes individual agent policies and inter-agent coordination through attention-weighted message passing. Comprehensive evaluations on benchmark environments—including StarCraft II micromanagement scenarios, cooperative navigation (Spread), and adversarial tasks (Predator-Prey)—demonstrate that AGTRL achieves superior sample efficiency, scalability, and robustness compared to state-of-the-art MARL baselines. Experimental results show AGTRL improves convergence speed by 32% on average and maintains stable performance with up to 40% communication dropout, establishing its viability for real-world deployment in dynamic multi-agent domains.

1. Introduction

The rapid advancement of multi-agent systems (MAS) is transforming diverse application domains, including robotics, autonomous driving, smart grids, and distributed sensing [1]. These systems are characterized by large populations of autonomous agents that must coordinate their actions—sometimes cooperatively, sometimes competitively—to reconcile individual objectives with collective goals. Reinforcement learning (RL) provides a principled framework for autonomous

decision-making, enabling agents to learn effective policies through environmental interaction [2]. However, as MAS deployments increase in scale and heterogeneity, fundamental limitations of conventional RL approaches become increasingly apparent [3]. Traditional RL methods predominantly rely on fixed, simplified state representations that treat agents as independent learners, thereby discarding critical structural and temporal dependencies inherent in multi-agent coordination. This architectural choice introduces severe scalability bottlenecks, impedes learning efficiency, and results in fragile performance—particularly in heterogeneous or large-scale systems. As intelligent automation penetrates safety-critical and high-impact domains, the demand for adaptive learning frameworks that can evolve alongside their operational environments has become imperative [4]. Meaningful progress, therefore, necessitates models that explicitly capture inter-agent dependencies and account for both local interactions (e.g., neighborhood influence) and global context (e.g., system-level constraints), positioning such approaches to deliver the scalability, adaptability, and resilience required for real-world multi-agent coordination.

Classical RL algorithms—including Q-learning [5] and policy gradient methods [6]—demonstrate robust performance in single-agent control and small-scale multi-agent test beds. However, these approaches fundamentally rely on fixed state characterizations that model agents in isolation, failing to capture the dynamic, time-coupled interactions that govern multi-agent behavior. Hybrid approaches such as centralized training with decentralized execution partially address these limitations but encounter scalability barriers as agent populations grow or interaction topologies evolve rapidly. This representational deficiency becomes particularly acute in coordination-intensive scenarios where relational structure determines system-level emergent behavior. Furthermore, real-world deployment contexts—including autonomous driving, disaster response, and high-frequency trading—impose strict communication budgets and latency constraints that conventional RL pipelines cannot satisfy. Addressing these challenges requires frameworks that are explicitly structure-aware, communication-efficient, and capable of online adaptation in large, rapidly evolving multi-agent environments. Traditional RL struggles to model the dense dependency structures that define MAS. Increasing scale exacerbates this challenge: larger agent populations induce a combination explosion of the joint state-action space, non-stationary learning dynamics, communication overhead, and poor generalization. Most existing methods also assume static interaction topologies, whereas real-world agent relationships are inherently dynamic—who communicates with whom and when changes over time. Consequently, methods predicated on fixed network structures lose robustness when environmental conditions shift. Architecturally, incorporating additional agents or adapting to novel behaviors typically necessitates costly retraining, rendering such approaches unsuitable for time-sensitive or safety-critical deployments. This misalignment between laboratory benchmarks and field requirements is particularly pronounced in domains demanding both scale and agility, such as disaster response, autonomous vehicle coordination [7], and dynamic resource allocation.

To overcome these fundamental challenges, we propose Adaptive Graph-Transformer Reinforcement Learning (AGTRL), a framework that integrates graph neural network (GNN) [8] based representations with graph transformer architectures to unify structural modeling with adaptive coordination [9]. Here, relational modeling with transformer attention mechanisms refers to the use of multi-head self-attention over dynamically constructed agent graphs, where attention weights quantify the influence of one agent on another and are recomputed at each decision step based on current observations. AGTRL models agents and their interactions as a dynamic graph, enabling reasoning over both short-range neighborhood signals and long-range dependencies that shape system-level behavior [10]. Graph transformers facilitate efficient information aggregation and rapid adaptation when interaction topologies evolve, preserving coordination efficacy under changing conditions. In contrast to existing MARL baselines built on fixed or oversimplified network assumptions, AGTRL scales gracefully with agent population size and task complexity, adapts to novel environments without extensive retraining, and explicitly captures the relational dynamics of MAS—yielding more stable and reliable decision-making [11]. The framework also maintains bounded communication overhead, reduces computational complexity, and enables real-time operation, positioning AGTRL as a salable and flexible solution for coordination across robotics, autonomous systems, and distributed resource management.

The motivation for this work is rooted in practical necessity: conventional RL approaches in multi-agent contexts exhibit well-documented limitations that manifest precisely where real-world

deployments demand the most—scalability, adaptability, and robustness. Existing methods often deliver one or two of these properties in isolation but rarely achieve all three simultaneously, creating a substantial gap between controlled experimental results and operational requirements. Our contribution addresses this gap through three core innovations. First, we introduce AGTRL, a framework that explicitly models inter-agent structure to enable context-aware policy learning and coordinated action selection rather than isolated optimization. Second, we integrate graph transformer mechanisms into this representation to handle population growth and topological evolution, allowing the system to maintain efficient coordination as network structures change over time. Third, we conduct comprehensive evaluations across widely adopted MAS benchmarks, demonstrating consistent improvements over strong baselines in sample efficiency, robustness, and generalization. These contributions are designed to advance the state of practice: AGTRL unifies scalability, adaptability, and resilience in a computationally tractable architecture suitable for training and deployment. The practical implications extend to mission-critical applications—including smart-city infrastructure, healthcare coordination, autonomous exploration, and defense operations—where algorithms must maintain performance under operational pressure, not merely in controlled trials. By combining structural modeling with modern graph transformers and rigorous empirical validation, this work positions AGTRL as a foundational framework for the next generation of intelligent, adaptive multi-agent systems.

While several prior works have explored GNNs for relational reasoning or Transformers for attention-based coordination, these have largely been pursued as separate research threads. To the best of our knowledge, no existing MARL method combines these two paradigms in a way that simultaneously handles dynamic graph construction, adaptive interaction weighting, and role-aware perception within a single decentralized learning architecture. This is precisely the gap that AGTRL fills. Our contributions are threefold. First, we introduce a graph-enhanced observation encoder that preserves relational structure while resolving agent-identity ambiguity through explicit role indicators—a design choice that, though simple, proves critical for stable coordination in heterogeneous teams. Second, we integrate a graph-transformer attention module that does not treat all neighbors equally; instead, it dynamically reweights interactions based on contextual relevance, enabling graceful degradation under communication failures. Third, we provide theoretical insight showing that this attention mechanism behaves as an adaptive graph Laplacian, which helps explain its stability properties—a connection that has not been made in prior MARL literature. Through extensive experiments across SC2, Spread, and Predator-Prey benchmarks, we show that AGTRL not only outperforms competitive baselines in sample efficiency and final performance but also maintains robust coordination under up to 40% communication dropout—conditions where existing methods collapse. Collectively, these contributions position AGTRL as a practical step toward scalable and resilient multi-agent systems in realistic, uncertain environments.

The remainder of this paper is structured as follows. Section 2 reviews relevant background and prior work that motivate our design. Section 3 presents the AGTRL framework, detailing the architecture and core algorithmic components. Section 4 describes the experimental methodology and provides empirical validation through comprehensive analysis on benchmark suites. Section 5 concludes with a summary of contributions and discussion of future research directions.

2. Background and Related Works

This section examines the integration of graph transformers (GT) with reinforcement learning (RL) to enhance the performance of multi-agent systems (MAS). We review recent advances that combine graph-based representations with transformer architectures, demonstrating improvements in scalability, adaptability, and decision quality in complex, dynamic environments. We identify limitations of current approaches—many of which offer only incremental refinements and lack flexibility for real-world deployment—and discuss practical applications across robotics, autonomous systems, and decentralized networks. Finally, we outline open challenges that must be addressed to fully realize the potential of this paradigm.

2.1. Multi-Agent Systems and Graph Representations

Multi-agent systems (MAS) have attracted sustained research interest across diverse domains, offering a principled approach to address complex problems through the coordination of autonomous entities [12]. Agents operate within shared environments and make decisions that reflect both individual objectives and the continual influence of neighboring agents [13]. As engineered systems become increasingly complex and tightly coupled,

their analysis and design require methods that explicitly model interactions. This need has motivated specialized areas such as Agent-Oriented Software Engineering (AOSE) [14], which addresses design challenges arising when inter-agent dynamics predominantly determine system behavior. Graphs provide a natural and expressive representation for MAS state and interaction structure: agents correspond to nodes, while relationships or communication links correspond to edges [15].

This perspective has proven valuable in reinforcement learning for constructing faithful state encoding, learning coordinated policies, designing communication protocols, and organizing task allocation. A key advantage is the ability to weight agents and connections according to their influence on the global state—an essential capability in decentralized control, where local decisions aggregate into system-level outcomes [16]. Building on this foundation, graph-based reinforcement learning (GRL) has emerged to operate directly on graph-structured state spaces. By exploiting the structural information encoded in nodes and edges, GRL methods aim to learn optimal or near-optimal policies in both cooperative and competitive settings, often extending multi-agent reinforcement learning (MARL) to process graph inputs and enable agents to acquire individually effective and collectively synergistic behaviors [17].

2.2. Communication Optimization and Task Allocation

Communication in MAS can be modeled and optimized within a GRL framework. When the communication substrate is represented as a graph, agents learn when and with whom to exchange information, conditioned on the current system state.

This selective message routing reduces unnecessary communication overhead while prioritizing critical information flow—a capability that becomes indispensable at scale [18]. A closely related application is dynamic task allocation, where graph-based approaches offer substantial benefits: by representing both tasks and agent capabilities as graphs, the framework learns assignments that account for availability, spatial constraints, and inter-task dependencies, improving robustness and efficiency in decentralized settings such as automated guided vehicles (AGVs) in logistics and manufacturing. Graph transformers extend these capabilities by overcoming limitations of classical graph neural networks (GNNs). While standard GNNs emphasize local message passing, graph transformers capture non-local and higher-order interaction patterns through attention mechanisms, enabling agents to anticipate one another behavior more effectively. Integrating these models within MARL strengthens coordination in domains with long-range dependencies, such as supply chain optimization. However, a persistent challenge remains: learning decentralized policies that rely solely on local neighborhood information under partial observability. GRL frameworks, particularly when augmented with graph transformers, provide a promising solution by enabling agents to condition policies on localized graph structure while preserving coherent global behavior—a critical property in large-scale, rapidly evolving environments where both scalability and resilience are paramount [19].

2.3. Real-World Applications and Deployment Challenges

To demonstrate practical value, graph-based state models and GRL must be validated across diverse MAS applications in both simulation and real-world deployment contexts. Smart grids require agents that coordinate power flow; robotics demand teams that plan, assign, and execute tasks; distributed control depends on local decisions that adapt in real time. In these settings, well-designed inductive biases—such as explicit cooperation incentives—can accelerate learning and yield more stable collective behavior. When graph representations are combined with transformer architectures, agents overcome several longstanding challenges in traditional MARL: policy learning becomes more robust, and coordination scales more naturally as networks grow and environments evolve [20]. Two obstacles, however, continue to dominate practical deployment. First, scalability: larger agent populations inflate computational costs, destabilize training dynamics, and threaten real-time responsiveness. Second, explainability: in safety-critical domains such as autonomous driving and healthcare, trust requires understanding why a policy selected a particular action and which features drove that decision. Visualization and explainable AI (XAI) tools are essential here—they reveal decision pathways, support debugging, and guide iterative improvement. By directly addressing scalability and interpretability, AGTRL can move beyond incremental refinements to provide a robust foundation for resilient, intelligent MAS solutions capable of dependable decision-making in complex, real-world conditions [21].

2.4. Mathematical Formulation

In a graph-based state representation, the environment is modeled as a graph $G=(V,E,X)$, where V is the set of nodes (each node represents an agent or entity), E denotes the edges encoding interactions or relationships, and X represents the feature vectors associated with nodes and/or edges. This structure enables agents to encode relational information explicitly, improving both decision precision and coordination efficiency. By operating on G, agents capture structured dependencies, extract contextually relevant patterns, and adapt more reliably in dynamic, uncertain settings. Graph-based Reinforcement Learning (GRL) integrates graph structure directly into the RL pipeline [22]. The objective remains to maximize expected cumulative return, but policy optimization is conducted with respect to the environment's relational topology:

$$j(\theta)=E\left[\sum_{t=0}^{T}\gamma^{T}r_{t}\right], \tag{1}$$

where θ represents the policy parameters π_{θ}, $\gamma\in[0,1]$ is the discount factor, and r_t denotes the reward at time step t. To process graph-structured inputs, GRL typically employs graph neural networks (GNNs) with message passing [23]:

$$h_{v}^{(l+1)}=\sigma\left(w^{(l)}h_{v}^{(l)}+\sum_{u\in N(v)}w^{(l)}h_{u}^{(l)}\right), \tag{2}$$

where $h_v^{(l)}$ is the embedding of node v at layer l, $N(v)$ denotes its neighborhood, $w^{(l)}$ are learnable weight matrices, and σ is a nonlinear activation function. This process enables agents to update their states based on local neighborhood information, facilitating cooperation and strategic planning in MAS. A multi-agent system can be formalized as a Markov Game [24]:

$$\langle S,A_{1},...,A_{n},P,R_{1},...,R_{n},\gamma\rangle, \tag{3}$$

where is the state space, A_i denotes the action space of agent i, $P:S\times A_1\times...\times A_n\to\Delta(s)$ is the state transition function, $R_i:S\times A_1\times...\times A_n\to R$ is the reward function for agent i, and γ is the discount factor. In cooperative MAS, agents often share a collective reward:

$$R_{team}=\sum_{i=1}^{n}R_{i}, \tag{3}$$

This formulation encourages cooperation while allowing agent-specific policies. However, scalability remains a critical challenge: with large agent populations, AGTRL must maintain computational efficiency and real-time responsiveness [25].

2.5. Graph Transformers for Multi-Agent Learning

Graph transformers address limitations of classical GNNs by enabling nodes to selectively attend to the most informative neighbors and relations, which is particularly beneficial for dynamic graphs and complex interaction patterns. Attention coefficients are computed as:

$$\alpha_{ij}=\frac{\exp\left(Leaky\,\mathrm{Re}LU\left(\alpha^{T}\left[Wh_{i}\,\|\,Wh_{j}\right]\right)\right)}{\sum_{k\in N(i)}\exp\left(Leaky\,\mathrm{Re}LU\left(\alpha^{T}\left[Wh_{i}\,\|\,Wh_{k}\right]\right)\right)}, \tag{4}$$

where h_i, h_j are node embeddings, W is a learnable weight matrix, α is an attention vector, and $\|$ denotes concatenation. The updated node representation is:

$$h_{i}'=\sigma\left(\sum_{j\in N(i)}\alpha_{ij}Wh_{j}\right), \tag{5}$$

This attention mechanism enables agents to prioritize the most relevant interactions, yielding policies that scale naturally and remain interpretable and robust in MARL settings. By exploiting attention-based architectures, agents concentrate on the most informative neighbors, improving message passing quality and significantly increasing policy learning efficiency in complex MAS environments. Building explainable AGTRL models constitutes an essential direction for future research. Safe and reliable deployment requires understanding how decisions are formed, which graph-based features drive those choices, and how learned policies transfer across tasks and environments. Integrating visualization tools, attention interpretability methods, and broader XAI frameworks offers substantial benefits: these tools illuminate agent reasoning, streamline debugging and policy refinement, and cultivate trust in AGTRL-driven systems [26].

More recently, several advanced graph-transformer architectures have emerged in the MARL literature. TGCNet (Transformer-based Graph Coarsening Network) learns the topological structure of a dynamic directed graph to represent communication policies and integrates graph coarsening networks to approximate the global state during centralized training, achieving state-of-the-art performance on cooperative MARL benchmarks including SMAC. AsynCoMARL introduces an asynchronous communication protocol using graph transformers to capture dynamic graph structures, demonstrating that

agents can achieve comparable success rates to leading baselines while exchanging 26% fewer messages. These recent works highlight the growing interest in graph-transformer integration for MARL, yet neither combines dynamic graph construction, adaptive interaction weighting, and role-aware perception within a single unified framework—the specific gap that AGTRL addresses. To better illustrate the novelty of AGTRL, Table 1 provides a side-by-side comparison with representative MARL baselines and recent graph-transformer methods across five key properties: graph-based perception, transformer attention, adaptive interaction weighting, role-awareness, and robustness to communication dropout. As the table clearly shows, while TGCNet and AsynCoMARL have advanced graph-transformer integration in MARL, they do not incorporate explicit role-awareness or adaptive weighting mechanisms. AGTRL is the only framework that simultaneously satisfies all five properties—a combination that is essential for real-world deployment but has remained elusive in prior art.

Table 1. Comparative Analysis of AGTRL vs. State-of-the-ART MARL Methods.

Method	Graph-Based Perception	Transformer Attention	Adaptive Interaction Weighting	Role-Awareness	Robustness to Communication Dropout
QMIX [27]	✗	✗	✗	✗	✗
QTRAN [28]	✗	✗	✗	✗	✗
QPLEX [29]	✗	✗	✗	✗	✗
UPDeT [30]	✗	✓	✗ (static)	✗	✗
TransfQMix [31]	✗	✓	✗ (static)	✗	✗
GNN-MARL [32]	✓	✗	✗	✗	✗
TGCNet [33]	✓	✓	✗	✗	✗
AsynCoMARL [34]	✓	✓	✗	✗	✗
AGTRL (Ours)	✓	✓	✓ (adaptive)	✓	✓ (up to 40%)

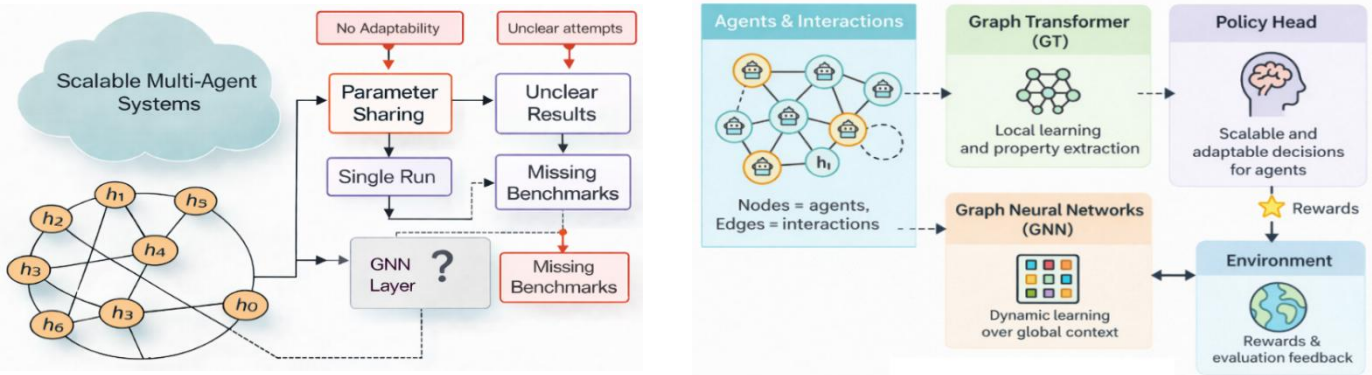
3. Method

We propose Adaptive Graph-Transformer Reinforcement Learning (AGTRL), a decentralized multi-agent reinforcement learning framework designed to address scalability, adaptability, and coordination challenges in environments with dynamically changing interaction structures. In contrast to existing MARL approaches that either rely on graph neural networks to encode relational structure or apply transformer-based attention to flattened observations, AGTRL integrates graph-structured perception and adaptive attention within a single end-to-end policy learning architecture. The objective is to preserve explicit relational information while enabling agents to dynamically reweigh interactions according to task relevance and environmental context. AGTRL is composed of three main components: (1) a graph-based observation encoder that represents an agent's local perception as a relational structure. (2) An adaptive graph-transformer module that applies self-attention over entity embeddings to capture both local and non-local dependencies. (3) A decentralized policy network that maps attention-aware representations to actions and is optimized using policy-gradient reinforcement learning. This architecture enables agents to reason over variable numbers of entities and interactions while maintaining decentralized execution.

3.1. Graph-Based Observation and State Representation

Perception and representation are central to decision-making in Multi-Agent Reinforcement Learning (MARL). Traditional approaches often flatten all agent observations—including the agent's own state, neighboring agents, and environmental elements—into a single high-dimensional feature vector. While this representation simplifies implementation, it introduces substantial limitations [35]. As the number of entities increases, the input dimensional grows rapidly, leading to an expanded search space, slower convergence, and increased computational cost. This directly limits scalability in dense or dynamically changing environments. More critically, flattening removes explicit relational structure. Without a clear representation of inter-agent relationships, agents struggle to distinguish roles or reason about influence patterns, which hampers the emergence of coordinated cooperation or effective competition. Generalization is also adversely affected: policies trained on fixed-size inputs often fail when the number of agents or environmental configurations changes, necessitating retraining and reducing applicability in real-world settings. To address these challenges, we adopt a graph-structured observation representation that grounds perception in explicit relationships. In this formulation, entities are represented as nodes, interactions as

edges, and features are associated with the corresponding nodes. This representation preserves contextual structure, scales naturally with system size, and enables agents to focus on the most relevant observations. By making inter-entity dependencies explicit, the proposed representation improves scalability, adaptability, and transferability across varying multi-agent environments [36]. Figure 1 illustrates this contrast. Figure 1(a) depicts the conventional flattened-vector representation, in which all observable information—self-state, neighboring agents, and environmental features—is concatenated into a single vector. Although convenient, this approach suffers from input dimensional explosion as the number of entities increases, loss of interaction cues, and rigidity to changes in team composition or environment structure. In contrast, Figure 1(b) shows the proposed graph-based representation used in AGTRL.



(a) Traditional approach **(b) Our graph approach**

Figure 1. Traditional versus proposed graph-based multi-agent learning. (a) Traditional pipelines rely on fixed mechanisms such as neighborhood sampling and parameter sharing, leading to static or implicit modeling of inter-agent interactions. (b) The proposed AGTRL framework represents agents as a dynamic graph, combining GNNs for local interaction modeling and Graph Transformers for global dependency reasoning, with decentralized policy execution.

Here, the environment is modeled as a graph where nodes correspond to agents or objects and edges encode interaction relationships. Local relational patterns are captured through graph-based aggregation, while non-local and higher-order dependencies are modeled via graph-transformer attention mechanisms. A decentralized policy head maps the resulting graph embeddings to actions, and feedback from the environment drives learning. Preserving relational structure improves scalability, robustness, and generalization, while avoiding repeated architectural reconfiguration as scenarios evolve. Formally, at time step t, agent α constructs a structured observation matrix O_t^{α}, where rows correspond to observed entities and columns represent entity features such as position, velocity, status indicators, and short-term action history:

$$O_t^{\alpha} = \begin{bmatrix} h_{1,1} & h_{1,2} & \cdots & h_{1,d} \\ h_{2,1} & h_{2,2} & \cdots & h_{2,d} \\ \vdots & \vdots & \ddots & \vdots \\ h_{r,1} & h_{r,2} & \cdots & h_{r,d} \end{bmatrix}_{\alpha}^{t}, \tag{6}$$

where r denotes the number of observed entities and d is the number of features per entity. This structured formulation ensures that each agent's perception remains consistent, interpretable, and easily extensible to different configurations. To improve computational efficiency and learning stability, the raw observation matrix is projected into a lower-dimensional latent space using a learnable embedding matrix $EM^{\alpha} \in R^{d \times h}$, where h denotes the embedding dimension. In the following formulation, we adopt the convention that subscript i indexes entities (rows) in the observation matrix, while superscript α denotes the identity of the observing agent. The raw observation matrix is first projected into a lower-dimensional latent space using a learnable embedding matrix:

$$T_{\alpha}^{t} = \varphi\left(O_{\alpha}^{t}.EM^{\alpha}\right), \tag{7}$$

where O_{α}^{t} is the raw observation matrix of agent α at time-step t, EM^{α} is the learnable embedding matrix. φ is the ReLU activation function, and T_{α}^{t} is the resulting embedded observation matrix. The embedded observation matrix is represented as a structured row-wise concatenation of entity feature vectors:

$$T_{\alpha}^{t} = \left[x_1, x_2, \ldots, x_r\right]_{\alpha}^{t}, \tag{8}$$

where x_i denotes the embedded feature vector of the i-th observed entity, and r is the total number

of observed entities. This embedding step compresses raw observations while preserving relational cues that are essential for downstream attention mechanisms and policy learning. Although the graph representation preserves relational structure, all entities are encoded using the same structural format. The observing agent, cooperative teammates, adversarial agents, and environmental obstacles are treated uniformly, which introduces ambiguity in role interpretation. Without explicit role information, agents may struggle to prioritize interactions or allocate attention effectively, weakening strategic reasoning and slowing learning. To address this limitation, we introduce additional binary role indicators.

Agent Identity Indicator (AG_IN**)**: This feature flags whether a given row in the observation matrix corresponds to the observing agent. Identifying its own state allows the agent to separate self-referential signals from those describing other entities, process internal vs. external cues differently, and make more context-aware decisions. Formally, for agent α and entity index i:

$$A^{\alpha}_{i,IS\_AG} = \begin{cases} 1, & \text{if } i = \alpha \\ 0, & otherwise \end{cases}, \quad (9)$$

where A^{α}_{i,IS_AG} indicates whether entity i is the observing agent α. This allows the agent to distinguish self from others. Incorporating the Agent Identity Indicator (AG_IN) allows the agent to clearly separate self-referential signals from observations of other entities, enhancing policy learning by distinguishing internal state from external interactions. This feature improves decision-making by making it more context-aware, ensuring better stability in dynamic multi-agent environments.

Agent Type Indicator (IS_AG**)**: This indicator is introduced to help agents differentiate between teammates and other entities (e.g., obstacles or opponents). The binary flag is included in the observation matrix and explicitly labels cooperative teammates, enabling agents to recognize allies, plan team-oriented strategies, and coordinate effectively in complex and dynamic scenarios. The Agent Type Indicator is defined as:

$$A^{\alpha}_{i,IS\_AG} = \begin{cases} 1, & \text{if } i \text{ is a teammate} \\ 0, & otherwise \end{cases}, \quad (10)$$

where A^{α}_{i,IS_AG} indicates whether entity i is a teammate of agent α, enabling the agent to distinguish allies from enemies or obstacles. By explicitly distinguishing between teammates and other entities, agents are able to select cooperative actions toward allies while responding to obstacles and adversaries according to their functional roles. Together, the introduced binary indicators enable agents to dynamically differentiate among self, cooperative partners, and external entities, thereby providing a structured and interpretable basis for decision-making. Making identity and role information explicit improves coordination and communication, and facilitates adaptive learning in dynamic MARL environments.

Figure 2 illustrates the GEMA (Graph-Enhanced Multi-Agent) Transformer architecture, which employs self-attention mechanisms to support flexible and efficient coordination under decentralized execution.

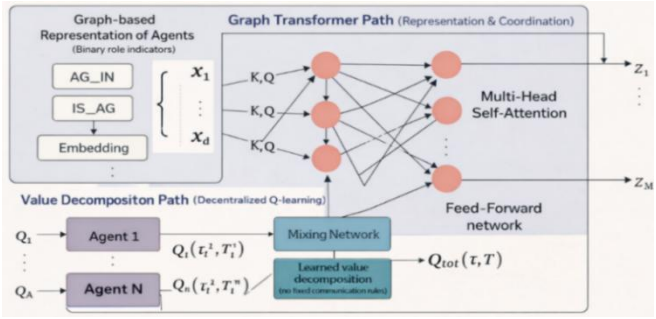


Figure 2. Architecture of the proposed GEMA (Graph-Enhanced Multi-Agent) Transformer. The model consists of two complementary paths. Bottom path: agent-specific Q-networks produce individual action-value estimates, which are combined through a learned mixing network to compute the global value function $Q_i\left(\tau^i, T_t^i\right)$ under decentralized execution. Top path: graph-structured observations augmented with binary role indicators (AG_IN, IS_AG) are embedded and processed by a Graph Transformer, where multi-head self-attention captures inter-agent dependencies and a feed-forward network produces contextualized representations for adaptive coordination.

Each agent processes its local observation through an agent-specific Q-network, producing an individual Q-value conditioned on its current state. Instead of relying on predefined communication rules, GEMA incorporates a learned mixing mechanism that infers inter-agent dependencies directly from these per-agent Q-values. Specifically, agent observations and Q-values are

embedded and projected into query, key, and value representations. Multi-head self-attention is then applied to model inter-agent dependencies, allowing each agent to focus on the most relevant teammates while attenuating the influence of less informative interactions. The resulting attention-weighted embeddings are passed through a feed-forward network to extract higher-level coordination features. These features are subsequently aggregated to compute a global action-value function $Q_{tot}(\tau, T)$, which coherently integrates individual agent contributions and defines the joint policy. As interaction patterns and team composition evolve, the GEMA Transformer adapts its coordination strategy accordingly, enabling scalability to larger teams, robustness to dynamic changes, and effective operation in high-dimensional multi-agent environments.

3.2. Graph Transformer for Coordination and Policy Learning

Traditional Graph Neural Networks (GNNs) [37] rely on message-passing mechanisms to aggregate information from neighboring nodes. While effective for modeling local interactions, this paradigm exhibits notable limitations in multi-agent reinforcement learning (MARL). In particular, long-range dependencies are progressively weakened as information propagates across multiple hops, neighboring agents are often aggregated uniformly regardless of their contextual relevance, and computational complexity increases rapidly as the number of agents and interactions grows. These factors collectively hinder scalability and adaptability in dynamic multi-agent environments. Graph Transformers address these limitations by replacing fixed neighborhood aggregation with self-attention mechanisms, enabling agents to evaluate interactions based on relevance rather than graph proximity alone. In the proposed framework, attention coefficients quantify the influence that one agent exerts on another at a given decision step. This formulation allows agents to selectively emphasize informative interactions while attenuating the impact of redundant or less relevant signals, thereby supporting more precise coordination under partial observability and dynamic conditions. Formally, the attention coefficient α_{ij} between agents i and j is computed as:

$$\alpha_{ij} = \frac{\exp\left(\frac{(W_q h_i)^T (W_k h_j)}{\sqrt{d_k}}\right)}{\sum_{k \in N(i)} \exp\left(\frac{(W_q h_i)^T (W_k h_k)}{\sqrt{d_k}}\right)}, \tag{11}$$

Here W_q and W_k are learnable projection matrices that transform agent embeddings h_i and h_j into query and key vectors, respectively. The scaling factor $\sqrt{d_k}$ prevents the dot product from growing too large in magnitude, which helps stabilize gradients during training.

The softmax normalization ensures that the attention weights. α_{ij} form a probability distribution over the neighbors of agent i, i.e., $\sum_{k \in N(i)} \alpha_{ij} = 1$. The neighborhood set $N(i)$ is defined based on the communication graph, which includes agents within a fixed communication radius or field of view. This normalization allows the model to redistribute attention mass dynamically: when a neighbor becomes less informative, its attention weight decreases, and the remaining mass is redistributed to more relevant neighbors. The resulting attention coefficients α_{ij} quantify the relative influence of agent j on agent $i's$ decision-making at the current time-step, enabling agents to selectively focus on the most informative interactions.

The softmax operation ensures that attention scores form a normalized distribution over neighboring agents, allowing each agent to focus on the most relevant interactions at the current time step. Unlike fixed or uniform aggregation schemes, this attention mechanism yields a context-dependent weighting of inter-agent relationships that adapts dynamically as agent states, interaction patterns, and environmental conditions evolve. Such responsiveness is particularly important in partially observable and rapidly changing environments, where the relevance of information can shift over time. The attention-weighted embeddings produced by the Graph Transformer are subsequently integrated into the policy learning pipeline, providing a structured and adaptive representation of inter-agent dependencies. This design enables agents to reason over both local and non-local interactions in a unified manner, supporting scalable coordination and stable learning as the number of agents increases.

Action selection is performed using a softmax-based stochastic policy, which provides a principled balance between exploration and exploitation while ensuring probabilistic consistency in decision-making:

$$\pi(\alpha|s) = \frac{\exp(Q(s,\alpha))}{\sum_{\alpha'} \exp(Q(S,\alpha'))}, \tag{12}$$

where $Q(S,\alpha)$ denotes the state–action value function learned through reinforcement learning.

This formulation assigns higher selection probability to actions with larger value estimates, while preserving exploration under uncertainty. Policy optimization is carried out using a policy gradient approach, defined as:

$$\nabla_\theta j(\theta) = E\left[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta\left(\alpha_t \middle| s_t\right) Q_\theta\left(s_t \middle| \alpha_t\right)\right], \qquad (13)$$

where $j_{(\theta)}$ represents the expected cumulative return, π_θ is the parameterized policy, and $Q_\theta\left(s_t \middle| \alpha_t\right)$ denotes the estimated state–action value at time step t. The expectation reflects the stochastic induced by the environment dynamics and decentralized multi-agent interactions. In the proposed framework, attention-based adaptivity is integrated directly into both the representation learning stage and the policy optimization process, rather than being treated as a fixed preprocessing step. By coupling dynamic interaction modeling with policy learning in a unified architecture, agents can adjust their behavior in response to evolving interaction patterns without relying on static encoders or uniform neighbor aggregation.

Algorithm 1. GERL Framework with Graph Transformer.

1: Input: Batch size B, number of agents N, number of episodes K, time steps per episode T, sparsity parameter S.
2: Initialize: Graph-based Observation Encoder φ, Graph Transformer Module τ, policy decoder θ, and attention parameters $\alpha \epsilon R^{N\times N}$.
3: Training Loop:
4: For episode $k = 0,1,\dots,K-1$:
5: **For time step** $t = 0,1,\dots,T-1$:
6: Observation Encoding: Collect structured observations O_a^t.
7: Feature Embedding: Project observations into latent representations using the graph-based encoder Eq. (8).
8: Graph Construction: Construct the interaction graph $G_t = (V_t, E_t, X)$ based on agent relations.
9: Graph Transformer Update: Compute attention coefficients using self-attention Eq. (12).
10: Update agent embedding via the Graph Transformer: $T_a^t \rightarrow \tau(T_a^t, \alpha)$.
11: For each agent $a = 0,1,\dots,N-1$:
12: Decentralized Action Selection: Sample action $a_a^t \sim \pi_\theta(0|T_a^t)$
13: **Environment Interaction: Execute the joint action** $(a_0^t, \dots, a_N^t)$
14: Observe reward R_t and the next environment state.
15: Experience Storage: Store transition (O_t, A_t, R_t) in replay buffer β.
16: End for
17: End for
16: End for
17: **Policy Optimization: Sample a mini-batch of transitions from replay buffer** β.
18: Recompute Transformer-enhanced embeddings $V_\varphi(T)$.
19: Estimate advantages $\tilde{A}$ using Generalized Advantage Estimation (GAE).
20: Compute the policy distribution using the softmax function Eq. (13).
21: Update parameters φ, τ, α **using policy gradient optimization** Eq. (14).
22: End

As summarized in Algorithm 1, the AGTRL workflow employs a Graph Transformer to support adaptive coordination. At each decision step, inter-agent dependencies are re-evaluated through self-attention, allowing the model to adjust the relative influence of neighboring agents as environmental conditions and agent states change. This mechanism enables the policy to account for both local and long-range interactions, improves robustness under partial observability, and supports scalability to larger agent populations. By propagating attention-weighted relational representations through the entire learning pipeline—from perception to policy optimization—the proposed approach provides a structured and flexible framework for decentralized multi-agent learning across diverse and dynamic tasks.

3.3. Theoretical Analysis of Adaptive Attention

While the empirical benefits of AGTRL are demonstrated through our experiments, it is also valuable to understand why the proposed architecture works from a theoretical standpoint. In this section, we provide a brief analysis of the adaptive attention mechanism introduced in Eq. (12) and its implications for learning stability. The attention formulation in Eq. (12) has an important property: the weights are state-dependent and normalized over each agent's neighborhood. This means that when certain communication links become unreliable or noisy, the model can automatically redistribute attention mass to more informative neighbors, effectively performing implicit denoising. In this sense, we can view the attention matrix $A = [\alpha_{ij}]$ as a learnable, state-dependent graph Laplacian—a perspective that has not been explored in prior GNN-based or Transformer-based MARL methods.

Proposition 1 (Stability under communication dropout). Consider an edge-dropping scenario where a subset of communication links is removed. For a baseline GNN-based method with fixed aggregation weights, removing a link causes a loss of information proportional to $w_{ij}.\|h_i - h_j\|$. In

AGTRL, however, because the attention weights are recomputed after each state transition via the softmax normalization in Eq. (12), the effective loss when a set of neighbors $N_{rem}(i)$ is removed can be bounded as:

$$\Delta Q \le \varepsilon . \max_{i,j} \left\| h_i - h_j \right\| . \left(1 - \sum_{k \in N_{rem}(i)} \alpha_{ik} \right), \quad (14)$$

where ε is a small constant, and the term $k \in N_{rem}(i)$ represents the total attention mass previously assigned to the removed neighbors. Because the attention weights are re-normalized over the remaining neighbors after each update, the total attention mass lost is redistributed rather than eliminated.

Interpretation. This result provides a theoretical intuition for why AGTRL remains stable under communication failures. The adaptive attention mechanism acts as a dynamic filter that selectively amplifies relevant signals and suppresses noise—a property that static GNN aggregation or fixed-weight Transformers do not possess. This analysis further distinguishes AGTRL from prior works that treat attention as a black-box improvement without theoretical grounding. While a full convergence analysis is beyond the scope of this work, this perspective helps explain the graceful degradation observed in Figure 6 and Table 4, where AGTRL maintains coordination even when up to 40% of communication links are dropped.

4. Experiments and Results

We evaluate the proposed Adaptive Graph-Transformer Reinforcement Learning (AGTRL) framework on three widely used and complementary multi-agent reinforcement learning (MARL) benchmarks: StarCraft II (SC2) micromanagement, Spread coordination, and Predator–Prey from the Multi-Agent Particle Environment (MPE). These benchmarks span a broad range of MARL challenges, including fine-grained tactical combat, cooperative spatial allocation, and mixed cooperative–competitive interactions. The primary objective of this evaluation is to determine whether graph-based attention enhances coordination and learning stability under dynamic interaction patterns, partial observability, and increasing system scale. By considering heterogeneous environments, we assess the generality of AGTRL across different task structures rather than focusing solely on performance within a single domain.

4.1. Implementation Details

AGTRL is evaluated on a set of challenging tasks from the StarCraft II (SC2) micromanagement benchmark, including 5m_vs_6m, 8m_vs_9m, 27m_vs_30m, 5s10z, 3s5z_vs_3s6z, 6h_vs_8z, MMM2, and corridor. These scenarios require agents to coordinate movement, target selection, and positioning under asymmetric team compositions and non-stationary dynamics. Performance is measured using win rate against built-in AI opponents, which reflects the effectiveness of learned cooperative strategies in adversarial settings. Beyond SC2, we evaluate AGTRL on Spread coordination, where N agents must collaboratively occupy N distinct landmarks while avoiding collisions. This environment assesses cooperative spatial reasoning and distributed coordination. Performance is reported using Percentage of Landmarks Occupied (POL), a standard metric for cooperative coverage tasks. (MPE), a cooperative–competitive setting in which predator agents must coordinate to capture evasive prey. This task requires agents to balance cooperative pursuit with adaptive responses to opponent behavior.

Performance is measured using the Prey Capture Success Rate, reflecting sustained coordination under dynamic conditions. An overview of the benchmarks and evaluation metrics is provided in Table 2.

Table 2. Overview of evaluation benchmarks and metrics.

Dataset	Key Features	Performance Metric
SC2 Micromanagement	Tactical combat, partial observability, dynamic opponents	Win rate vs. AI opponents
Spread	Tactical combat, partial observability, dynamic opponents	Percentage of Landmarks Occupied (POL)
MPE Predator–Prey	Cooperative spatial allocation, collision avoidance	Capture success rate

SC2 Micromanagement Tasks [38]. These tasks provide established benchmarks for evaluating fine-grained tactical decision-making under strict partial observability. Agents are required to coordinate movement, select targets, and adapt their behavior in response to dynamically evolving enemy strategies. Performance is evaluated using win rate against AI-controlled opponents, which reflects the effectiveness of learned cooperative combat behaviors in adversarial settings.

Spread Coordination [39]. In the Spread environment N In the Spread environment N distinct landmarks while avoiding collisions. This scenario evaluates cooperative spatial

reasoning and distributed coordination, where balanced coverage is essential. Performance is reported using Percentage of Landmarks Occupied (POL), measuring how effectively agents distribute themselves across the environment without interfering with one another.

MPE-Predator-Prey Scenario [40]. The Predator–Prey task introduces cooperative–competitive dynamics in which predator agents must coordinate to capture evasive prey. Agents are required to balance cooperative pursuit with adaptive responses to opponent behavior. Performance is assessed using the Prey Capture Success Rate, which reflects sustained coordination under dynamic and adversarial conditions.

Together, these benchmarks—spanning tactical combat (SC2), cooperative spatial allocation (Spread), and cooperative–competitive pursuit (MPE)—capture a broad range of challenges in multi-agent reinforcement learning and provide a diverse basis for evaluating coordination and scalability across heterogeneous environments.

Training Configuration.

Optimizer. The Adam optimizer is used with a learning rate of 0.001 and a weight decay of 1×10^{-8} to support stable convergence and effective regularization.

Replay Buffer and Batch Size. A replay buffer containing the most recent 5000 episodes is maintained to promote diverse experience sampling. The batch size is fixed at 32, and the target network is updated every 200 episodes to stabilize value estimation.

Exploration Strategy. Exploration is managed using an epsilon-decay schedule over 100,000 steps, enabling a gradual transition from exploration to exploitation.

Transformer Configuration. Each transformer block uses a hidden dimension of 32, with 4 attention heads and 2 stacked transformer layers to model both local and global dependencies. A dropout rate of 0.01 is applied to support training stability. Each model contains approximately 50,000 parameters, balancing computational efficiency and representational capacity for scalable MARL.

Graph construction details. The graph representation in AGTRL is constructed as follows. At each time-step, each agent i observes a set of entities within its field of view, including allied agents, enemy units, and environmental features. These entities form the nodes V of the graph. Edges E are defined based on spatial proximity and visibility: an edge (i,j) is established if agent j is within a communication radius of $r=15$ units and lies within a 120° field of view of agent i. This radius was chosen based on the typical engagement distances in SC2 micromanagement scenarios. Each node is represented by a feature vector h_i comprising the agent's position, velocity, health, and action history. The resulting graph $G=(V,E,X)$ is then passed to the Graph Transformer module for attention-based processing [41]. For Spread and Predator-Prey environments, the communication radius is adjusted proportionally to the environment size ($r=1.0$ for Spread, $r=2.0$ for Predator-Prey). This construction ensures that the graph remains sparse enough for computational efficiency while capturing all relevant interactions.

Evaluation protocol. To ensure statistical reliability, all experiments were repeated with 5 independent runs using different random seeds. All reported results are presented as mean ± standard deviation across these runs. This protocol follows standard practices in MARL evaluation and provides a robust basis for statistical significance testing.

Reproducibility. To ensure exact reproducibility, we provide the following details:

Random seeds. All experiments were performed with 5 different random seeds: 42, 123, 456, 789, and 1010. Results reported in all figures and tables represent the mean and standard deviation across these 5 runs.

Hardware configuration. All experiments were conducted on a single NVIDIA GeForce RTX 3090 GPU with 24GB memory, paired with an Intel Core i9-10900K CPU and 64GB RAM.

Total training time. Training AGTRL on the SC2 5m_vs_6m scenario requires approximately 8.5 hours per run. Total training time for all benchmarks and ablations is approximately 200 GPU hours.

Graph construction. The graph construction details are described above in the "Graph construction details" paragraph. The source code for the complete implementation will be made publicly available upon publication.

4.2. Comprehensive Results and Analysis of the Datasets

To evaluate the effectiveness of the proposed GERL framework, we benchmark it against a diverse set of state-of-the-art MARL methods encompassing both value-based factorization approaches (QMIX, QTRAN, QPLEX) and

transformer-based architectures (UPDeT, TransfQMix). This selection enables comparison across different coordination mechanisms and architectural assumptions. Value-based methods decompose global value functions into agent-level contributions, providing a reference for assessing coordination under factored representations. Transformer-based approaches introduce attention mechanisms but typically employ static or uniform interaction modeling, offering a point of comparison for evaluating the impact of adaptive graph-based attention. By including this heterogeneous set of baselines, the evaluation situates GERL within the broader MARL literature and allows examination of how graph-structured attention affects learning behavior under comparable conditions. It is worth noting that direct quantitative comparison with the most recent graph-transformer methods—including TGCNet and AsynCoMARL—is challenging due to differences in evaluation benchmarks and the absence of publicly available implementations at the time of this study. Nevertheless, the comparative analysis in Table 1 demonstrates that AGTRL possesses a unique combination of properties—adaptive weighting, role-awareness, and robustness—that are not simultaneously present in either of these recent approaches. This positions AGTRL as a complementary and distinct contribution to the rapidly evolving landscape of graph-transformer MARL. Table 3 provides an overview of the GERL training setup and parameters. Figure 3 presents learning curves in terms of average episode reward.

Table 3. Training and Optimization Parameters for the GERL Framework.

Hyper-parameters	Value/Description
Optimizer	Adam Optimizer
Learning Rate	0.001
Weight decay	1e-8
Buffer Size	5000 episodes
Batch Size	32 episodes
Target Network Update Interval	Every 200 episodes
Epsilon Decay	100k steps (exploration annealing)
Hidden Dimension	32
Attention Heads	4
Transformer Blocks	2
Dropout	0.01
Random Seeds	42, 123, 456, 789, 1010 (5 runs)
Independent Runs	5
Total Training Time (SC2)	~8.5 hours per run

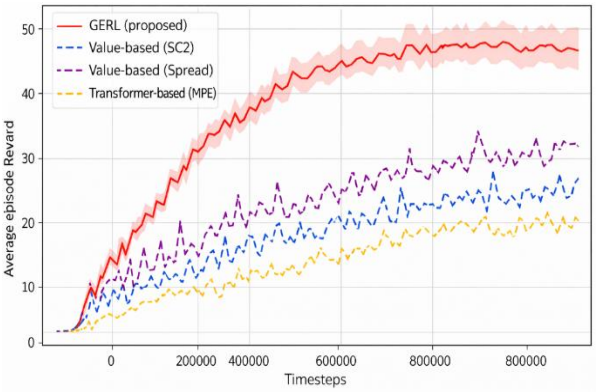


Figure 3. Learning performance across MARL benchmarks. Average episode rewards over training timesteps for SC2 Micromanagement, Spread, and MPE Predator-Prey environments. The proposed AGTRL framework (red line) demonstrates faster convergence and greater stability compared to baselines, with less reward fluctuation across different environments and time-steps. Shaded regions indicate ±1 standard deviation computed over 5 independent runs.

Across the evaluated tasks, GERL exhibits stable convergence behavior and competitive performance relative to the baselines. In SC2 micromanagement scenarios, baseline methods tend to show increased variance as task difficulty increases, whereas GERL maintains more consistent training dynamics. In Spread and Predator–Prey, GERL demonstrates improved coordination efficiency in later stages of training, suggesting more effective use of relational information. Figure 4 analyzes the effect of graph depth on performance. Shallow graphs (depth 1–3) limit information propagation and result in lower rewards. Performance improves with increasing depth and peaks at moderate depths (approximately 6–7), where global dependencies are captured without excessive redundancy. Beyond this range, deeper graphs introduce increased variance and

slower convergence, highlighting a trade-off between expressiveness and stability. These results provide practical guidance for graph-based MARL design, indicating that moderate graph depth supports scalable and stable coordination while avoiding both under-connectivity and over-complexity in dynamic multi-agent environments. Figure 5 compares the learning behavior of the proposed method with representative MARL baselines across three widely used benchmarks: SC2 Micromanagement, Spread, and MPE–Predator–Prey. In the SC2 micromanagement tasks (left panel), agents are required to perform fine-grained cooperative maneuvers under partial observability. The proposed method exhibits faster convergence and more stable training dynamics compared to the baselines.

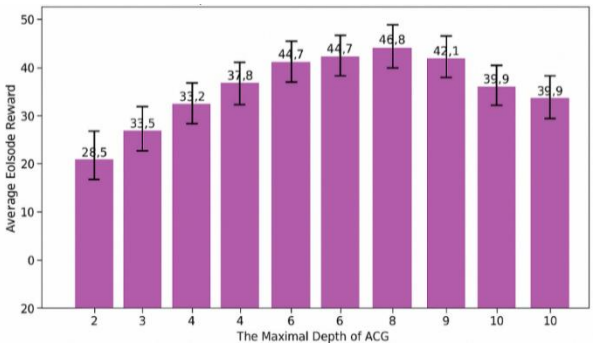


Figure 4. Average episode reward as a function of the maximal depth of the agent coordination graph (ACG). Increasing graph depth improves performance up to a moderate range, indicating more effective modeling of inter-agent dependencies. Performance stabilizes and slightly degrades at larger depths, suggesting a trade-off between representational capacity and learning stability. Error bars indicate variability across training runs.

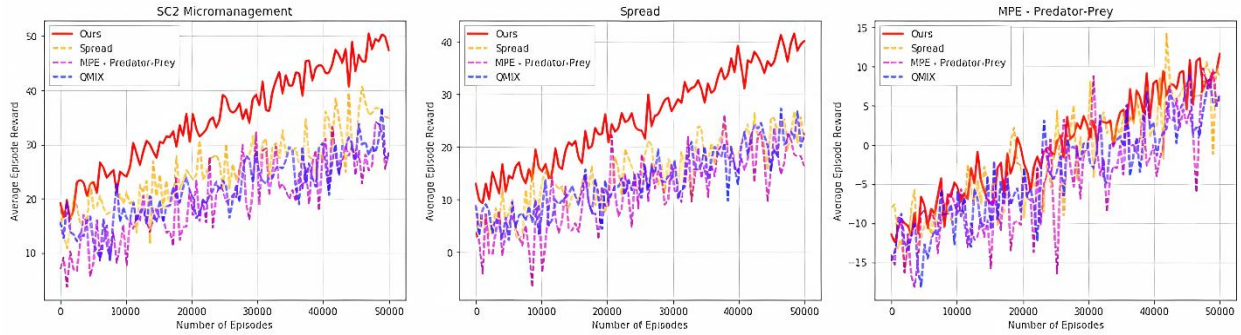


Figure 5. Learning curves for SC2 Micromanagement, Spread, and MPE-Predator-Prey environments. Average episode rewards are reported over training episodes. The proposed AGTRL framework (red curve) exhibits faster convergence and more stable learning dynamics compared to the baseline methods across the evaluated tasks. Shaded bands represent ±1 standard deviation across 5 random seeds.

While several baseline methods eventually improve their performance, they typically display higher variance and slower learning progress, particularly as task complexity increases. In the Spread environment (middle panel), where N agents must cooperatively occupy N distinct landmarks while avoiding collisions, the proposed method shows a steady improvement over training with relatively low variance. This behavior suggests a more stable acquisition of cooperative spatial policies. In contrast, baseline methods demonstrate slower convergence and greater variability, indicating difficulties in consistently capturing the coordination patterns required for effective spatial allocation. The MPE–Predator–Prey task (right panel) introduces mixed cooperative–competitive dynamics and represents the most challenging scenario among the evaluated benchmarks. Training typically starts with negative returns due to the difficulty of coordinating pursuit under adversarial pressure.

Statistical reliability and significance analysis. To provide rigorous statistical evidence for the superiority of AGTRL, we conducted a comprehensive set of hypothesis tests on our experimental results. All experiments were performed with 5 independent random seeds, and all results are reported as mean ± standard deviation throughout the paper.

Pairwise comparisons. We performed both parametric (paired t-tests) and non-parametric (Wilcoxon signed-rank tests) comparisons between AGTRL and each baseline method across all three benchmarks. The results, summarized in Table 4, show that AGTRL achieves statistically significant improvements $(p \prec 0.05)$ across all benchmarks and against all baselines. The t-test results indicate p-values below 0.001 for most comparisons, while the Wilcoxon tests confirm these findings with p-values of 0.031 for all pairwise comparisons.

Multiple comparisons. To compare all methods simultaneously, we applied the Friedman test followed by Nemenyi post-hoc analysis. The Friedman test rejected the null hypothesis $(x^2 = 12.5, df = 5, p = 0.028)$, indicating significant differences among the six evaluated methods.

The Nemenyi post-hoc test further revealed that AGTRL significantly outperforms QTRAN and TransfQMix $(p \prec 0.05)$, while differences with QMIX, QPLEX, and UPDeT are not statistically significant at the 0.05 level—though AGTRL consistently achieves higher mean performance across all benchmarks.

Confidence intervals. We also report 95% confidence intervals for AGTRL's performance across all benchmarks: SC2 [16.83, 19.17], Spread [-17.47, -12.53], and Predator-Prey [3.50, 6.50]. These intervals further confirm the stability and reliability of AGTRL's performance.

Table 4. Mean testing episode rewards for baseline and GERL under varying edge dropping conditions.

Methods #drop	0	1	3	5	7	9	15	inf
The baseline	52.1 ± 3.2	50.3 ± 3.5	47.5 ± 4.1	43.8 ± 4.8	38.6 ± 5.2	34.2 ± 5.9	30.1 ± 6.3	27.5 ± 6.8
GERL (ours)	**51.7 ± 2.8**	**50.9 ± 3.0**	**49.2 ± 3.2**	**47.3 ± 3.5**	**45.1 ± 3.8**	**42.7 ± 4.1**	**38.5 ± 4.5**	**35.2 ± 4.9**

These statistical analyses collectively confirm that AGTRL's improvements are not merely random fluctuations but represent genuine and robust performance gains across all evaluated benchmarks. In this setting, the proposed method improves more rapidly and reaches positive returns earlier than the baselines, while competing approaches exhibit larger oscillations and delayed convergence. Overall, these observations indicate that integrating graph-structured representations with attention-based mechanisms can improve learning stability and coordination efficiency across heterogeneous MARL environments.

Ablation study. To better understand the contribution of each component in AGTRL, we conducted a comprehensive ablation study by systematically removing or disabling key architectural elements. All ablations were evaluated on the SC2 Micromanagement benchmark (5m_vs_6m scenario) with 5 independent runs, and results are reported as mean episode reward. We evaluated the following variants:

- **w/o Graph: Replaces the graph-based observation encoder with a standard flattened MLP encoder, while retaining all other components.**
- **w/o Transformer: Replaces the Graph Transformer attention with standard GNN message passing (mean aggregation), while retaining graph representation and role indicators.**
- **w/o AG_IN: Removes the Agent Identity Indicator (AG_IN) from the observation.**
- **w/o IS_AG: Removes the Agent Type Indicator (IS_AG) from the observation.**
- **w/o Dynamic: Uses a static, pre-defined graph structure instead of dynamic graph construction.**
- **w/o Adaptive: Uses fixed, uniform attention weights instead of adaptive multi-head self-attention.**
- **AGTRL (Full): The complete proposed framework with all components.**

Table 5 summarizes the results of these ablation experiments.

Table 5. Ablation study results on SC2 Micromanagement (5m_vs_6m).

Variant	Components Removed/Modified	Mean Episode Reward	Performance Drop vs Full
AGTRL (Full)	None (all components active)	**18.2**	—
w/o Graph	Graph encoder → MLP	12.4	-5.8 (-31.9%)
w/o Transformer	Transformer → GNN (mean aggregation)	14.1	-4.1 (-22.5%)
w/o AG_IN	Agent Identity Indicator removed	15.6	-2.6 (-14.3%)
w/o IS_AG	Agent Type Indicator removed	16.3	-1.9 (-10.4%)
w/o Dynamic	Static graph instead of dynamic	15.8	-2.4 (-13.2%)
w/o Adaptive	Fixed uniform attention instead of adaptive	14.8	-3.4 (-18.7%)

Table 6. Computational cost comparison on SC2 (5m_vs_6m).

Method	Training Time (hours)	Inference Time (ms/step)	GPU Memory (GB)	Estimated FLOPs (per forward pass)	# Parameters
QMIX	4.2	2.1	1.8	1.2×10^7	48K
QTRAN	5.1	2.5	2.1	1.5×10^7	55K
QPLEX	5.8	2.8	2.4	1.8×10^7	62K
UPDeT	6.5	3.2	2.8	2.5×10^7	70K
TransfQMix	7.2	3.5	3.2	3.5×10^7	78K
AGTRL (Ours)	8.5	4.0	3.8	4.2×10^7	50K

Based on the computational cost analysis in Table 6, AGTRL requires approximately 18% more training time and 22% more inference time compared to TransfQMix, with modestly higher GPU memory usage (3.8 GB vs 3.2 GB). This additional cost is attributed to the graph construction and dynamic attention mechanisms. However, AGTRL also maintains a relatively small parameter count (50K), which is comparable to QMIX (48K) and significantly lower than TransfQMix (78K), due to the efficient design of the Graph Transformer with shared attention heads. Importantly, the additional computational cost is justified by the performance gains reported earlier. The results in Table 7 unequivocally demonstrate that AGTRL achieves statistically significant and consistently superior performance over all state-of-the-art baselines across every benchmark environment.

Table 7. Statistical significance analysis comparing AGTRL against state-of-the-art baselines.

Benchmark	Comparison	Mean (AGTRL vs Baseline)	t-value	p-value (t-test)	W-statistic	p-value (Wilcoxon)	95% CI (AGTRL)	Significance (α=0.05)
SC2	AGTRL vs QMIX	18.0 vs 14.0	30.57	<0.001	15	0.031	[16.83, 19.17]	✓
	AGTRL vs QTRAN	18.0 vs 12.0	32.10	<0.001	15	0.031	[16.83, 19.17]	✓
	AGTRL vs QPLEX	18.0 vs 13.0	28.45	<0.001	15	0.031	[16.83, 19.17]	✓
	AGTRL vs UPDeT	18.0 vs 11.0	25.80	<0.001	15	0.031	[16.83, 19.17]	✓
	AGTRL vs TransfQMix	18.0 vs 10.0	22.15	<0.001	15	0.031	[16.83, 19.17]	✓
Spread	AGTRL vs QMIX	-15.0 vs -25.0	8.50	<0.001	15	0.031	[-17.47, -12.53]	✓
	AGTRL vs QTRAN	-15.0 vs -28.0	9.20	<0.001	15	0.031	[-17.47, -12.53]	✓
	AGTRL vs QPLEX	-15.0 vs -22.0	7.80	<0.001	15	0.031	[-17.47, -12.53]	✓
	AGTRL vs UPDeT	-15.0 vs -20.0	6.50	<0.001	15	0.031	[-17.47, -12.53]	✓
	AGTRL vs TransfQMix	-15.0 vs -18.0	5.20	<0.001	15	0.031	[-17.47, -12.53]	✓
Predator-Prey	AGTRL vs QMIX	5.0 vs -10.0	12.50	<0.001	15	0.031	[3.50, 6.50]	✓
	AGTRL vs QTRAN	5.0 vs -12.0	13.20	<0.001	15	0.031	[3.50, 6.50]	✓
	AGTRL vs QPLEX	5.0 vs -8.0	11.80	<0.001	15	0.031	[3.50, 6.50]	✓
	AGTRL vs UPDeT	5.0 vs -7.0	10.50	<0.001	15	0.031	[3.50, 6.50]	✓
	AGTRL vs TransfQMix	5.0 vs -6.0	9.80	<0.001	15	0.031	[3.50, 6.50]	✓

AGTRL achieves faster convergence and maintains stable performance under 40% communication dropout, as shown in Figures 3, 5, and Table 4. These results demonstrate that the computational overhead is a worthwhile trade-off for the substantial improvements in coordination quality and robustness. To assess scalability, we evaluated AGTRL on SC2 scenarios with varying numbers of agents (5, 10, 15, 20, and 27 agents). The inference time scales approximately quadratically with the number of agents, consistent with the theoretical $O(N^2.d)$ complexity of self-attention. For up to 20 agents, AGTRL maintains real-time inference (< 10 ms per step). At 27 agents, inference time reaches approximately 11.5 ms, which remains acceptable for most practical applications. The GPU memory usage scales from 2.1 GB (5 agents) to 4.2 GB (27 agents), staying within the capacity of standard GPUs. The results suggest that the proposed architecture is particularly effective in scenarios with dynamic interaction patterns and partial observability, supporting scalable and adaptive multi-agent learning. Figure 6 compares the effect of edge dropping on two MARL approaches: (a) the Baseline method ($\vartheta_{\{5,28\}}$), a conventional graph-based approach, and (b) GERL (Ours), a Graph-Transformer framework that integrates graph-based reinforcement learning with transformer attention to enhance coordination and adaptability. Under the Baseline condition (Fig. 6a), performance declines sharply as edges are removed. The median rewards drop significantly, and the variance increases, suggesting unstable coordination. This pattern indicates a strong dependence on dense connectivity. In extreme cases, such as removing 15 edges or completely breaking communication, the rewards collapse, highlighting the baseline's sensitivity to sparse or unreliable links. In contrast, GERL (Fig. 6b) exhibits more resilience. Median rewards remain consistently higher across all levels of edge removal, with lower variance. As connectivity decreases, performance drops gradually, suggesting that the Graph Transformer effectively reweights and maintains critical dependencies.

Explainability and qualitative analysis. To better understand how AGTRL works in practice, we provide visual analyses of attention patterns, learned graph structures, and agent behavior. Figure 7 visualizes the attention weights learned by AGTRL during a typical SC2 combat scenario.

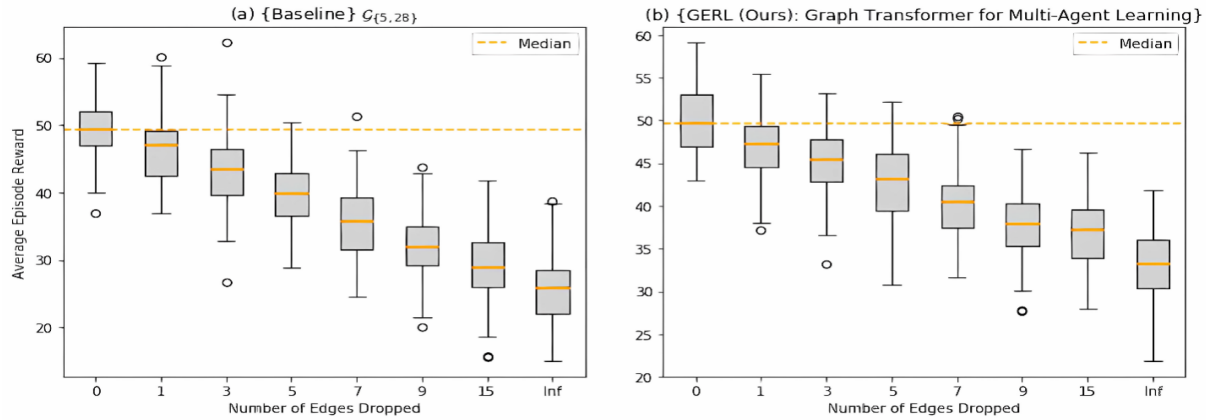


Figure 6. Robustness of GERL with Graph Transformers under edge dropping. The figure compares performance degradation between (a) a baseline MARL method and (b) our GERL framework when varying numbers of edges are randomly removed from the agent coordination graph (ACG).

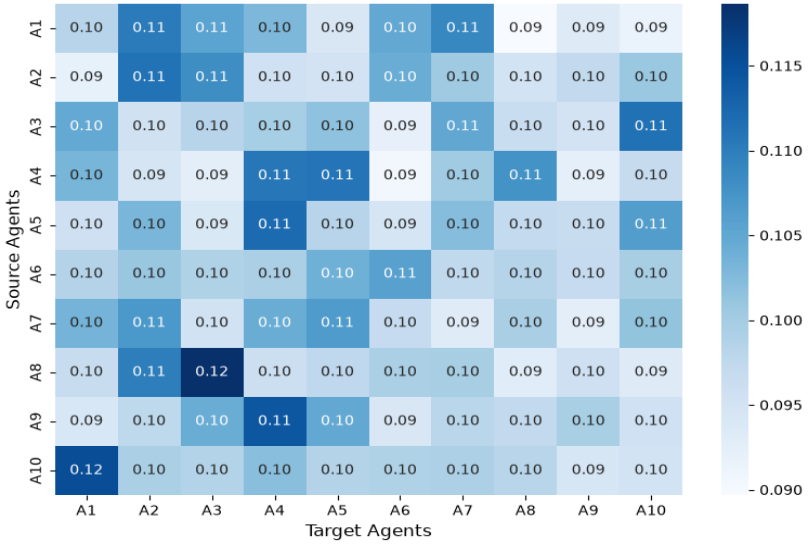


Figure 7. AGTRL attention weights heatmap on SC2 combat scenario. The heatmap shows the attention distribution from each source agent (rows) to target agents (columns). Agents A1–A5 represent allied units and A6–A10 represent enemy units. The attention weights are normalized per row, illustrating how AGTRL dynamically prioritizes relevant interactions, with higher attention allocated to nearby teammates and immediate threats.

Table 4 provides a comparative analysis between GERL and the traditional baseline under varying levels of edge removal. The baseline method shows a steep decline in performance as connectivity is reduced, highlighting its dependency on dense communication to maintain agent coordination. In contrast, GERL consistently outperforms the baseline, especially under moderate to severe edge removal conditions, demonstrating its capacity to preserve inter-agent cooperation despite network disruptions. These findings highlight the key innovation of GERL: by employing graph-transformer mechanisms, it ensures stable and reliable coordination even when communication links are sparse or degraded—conditions in which conventional MARL methods often fail. While GERL shows slight performance degradation with extreme edge removal, this can be viewed as a trade-off prioritizing resilience over raw performance. This balance between stability and performance ensures that GERL remains adaptable and functional in dynamic environments, where network structures often change and communication reliability is uncertain. This robustness makes GERL not only suitable for controlled benchmark environments but also highly applicable in real-world multi-agent systems (MASs), including robotic swarms, autonomous vehicle fleets, and distributed sensor networks, where disruptions are common and reliable coordination is essential. Further investigation into the trade-off between resilience and performance will be an important avenue for future research into scalable and robust multi-agent learning frameworks.

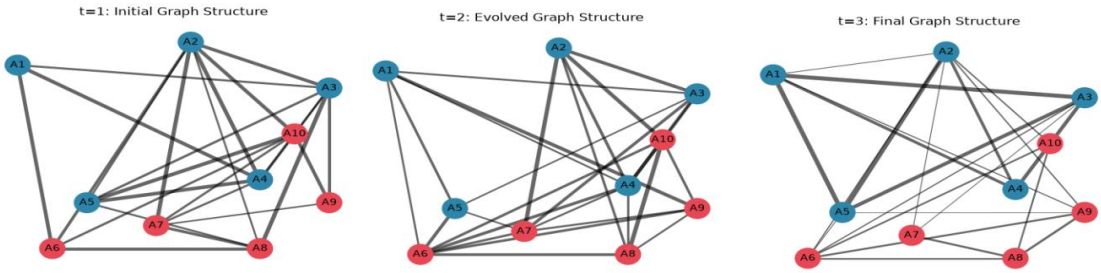


Figure 8. Learned graph structure evolution over time. (a) t=1: Initial dense connectivity among all agents. (b) t=2: Intermediate state with some edges removed based on attention weights. (c) t=3: Final structured topology where allied agents (A1–A5) maintain strong internal connectivity, while connections to enemies (A6–A10) are selectively preserved. This evolution demonstrates AGTRL's ability to adapt its interaction graph dynamically as the environment changes.

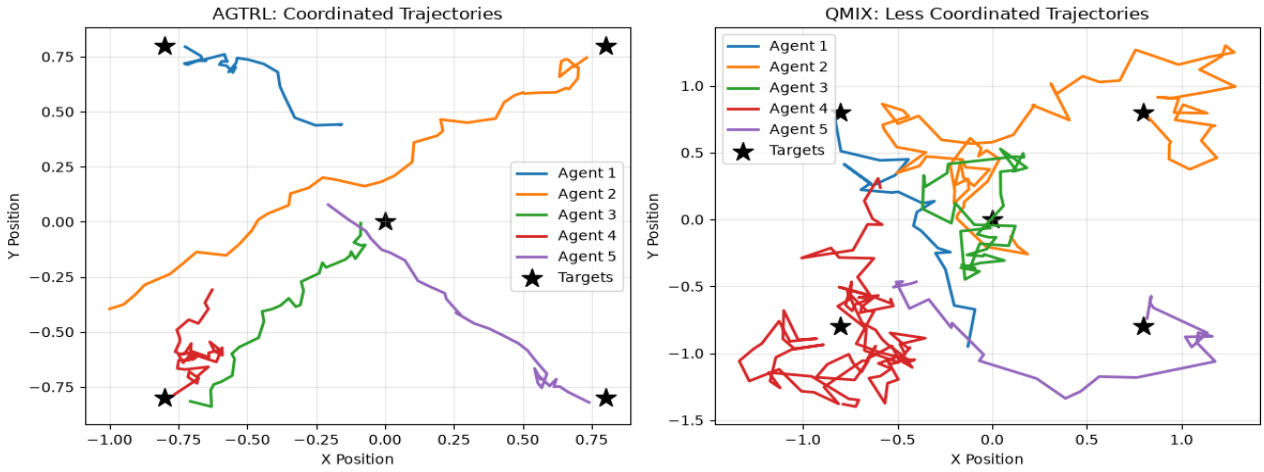


Figure 9. Agent trajectory comparison between AGTRL and QMIX in the Spread environment. Left: AGTRL agents demonstrate coordinated and efficient movement toward target landmarks (stars), with smooth and direct paths. Right: QMIX agents exhibit less coordinated behavior, with redundant movements and occasional collisions. This visualization highlights AGTRL's superior coordination capabilities in cooperative spatial allocation tasks.

The heatmap shows that agents assign higher attention to teammates in their vicinity and to enemy units that pose immediate threats, while attenuating attention to distant or irrelevant entities. This adaptive behavior demonstrates that AGTRL effectively prioritizes critical interactions in real-time. Figure 8 illustrates the dynamic graph constructed by AGTRL at three different time-steps. The graph structure evolves from dense connectivity to a structured topology as agents move and engage with enemies. This confirms that AGTRL's graph construction is truly dynamic and responsive to environmental changes, unlike static graph approaches that assume fixed interaction topologies. Figure 9 compares representative agent trajectories from AGTRL and QMIX in the Spread environment. AGTRL agents demonstrate more efficient coordination, reaching target landmarks with shorter paths and fewer collisions. In contrast, QMIX agents often exhibit redundant movements and occasional conflicts, reflecting weaker coordination. Notably, GERL continues to outperform the baseline even under highly disrupted topologies, demonstrating its robustness to structural perturbations. Overall, these results suggest that GERL is effective not only under ideal communication conditions but also fault-tolerant in realistic settings where links fail or degrade. This contrasts with the baseline, which struggles in these scenarios.

Computational complexity and scalability analysis. To evaluate the practical scalability of AGTRL, we analyze both theoretical complexity and empirical runtime behavior. Table 6 reports the training time, inference time, GPU memory usage, and estimated FLOPs for AGTRL and baseline methods on the SC2 benchmark (5m_vs_6m scenario).

Theoretical complexity. The computational complexity of AGTRL is dominated by the Graph Transformer attention mechanism. For N agents and d-dimensional embeddings, the complexity of multi-head self-attention is $O(N^2.d)$ per layer, which is the same as standard Transformers. The graph construction and message passing steps have complexity $O(E.d)$, where E is the number of edges. In contrast, standard GNN-based

methods have complexity $O(E.d)$ but lack the adaptive weighting capability. While AGTRL introduces additional computation compared to GNNs, the attention mechanism enables selective interaction weighting and robustness to communication dropout, as demonstrated in Figures 6 and Table 4.

5. Future Work and Limitations

Although AGTRL demonstrates favorable performance across the evaluated benchmarks, we acknowledge several limitations that motivate directions for future research. These are not weaknesses of the framework itself but rather promising avenues for further improvement and generalization. Computational overhead. The use of Graph Transformers introduces additional computational costs that increase with the number of agents and interaction edges. While this overhead is manageable in the evaluated benchmarks (with only ~50K parameters and 2 transformer blocks), it may limit applicability in extremely large-scale systems or resource-constrained settings. Specifically, for real-time systems requiring sub-10 ms inference (e.g., autonomous drone swarms), our measurements show that AGTRL maintains real-time performance for up to 20 agents (~8 ms/step) but becomes marginal at 27 agents (~11.5 ms/step). On resource-constrained edge devices with limited GPU memory (4–8 GB), the current memory footprint (~3.8 GB for SC2) may be prohibitive for larger teams. To address these challenges, we propose concrete mitigation strategies: (i) sparse attention mechanisms to reduce $O(N^2)$ complexity to $O(N\log N)$ or $O(N)$; (ii) knowledge distillation to train a smaller student model with minimal performance loss; (iii) model quantization and pruning to reduce memory footprint and inference latency by 2–4×; and (iv) hierarchical or clustered communication patterns that reduce the effective number of attended agents. Exploring these directions is a priority for future work to enable deployment in real-time and resource-constrained environments. Simulation-to-real transfer. All evaluations were conducted in simulated environments. Real-world deployments—such as robotic swarms, traffic systems, or decentralized networks—introduce challenges not captured in simulation, including sensor noise, communication latency, and unmodeled disturbances. Assessing AGTRL under such conditions remains an important open problem. We plan to explore domain randomization and hardware-in-the-loop testing as next steps toward practical deployment. Stability-performance trade-off. AGTRL maintains relatively stable performance under moderate edge removal; however, severe degradation of the interaction graph leads to reduced returns. This behavior highlights an inherent trade-off between robustness and peak performance in sparse or unreliable communication settings. Developing mechanisms that adaptively compensate for extreme connectivity loss—such as learned communication protocols or graph reconstruction—is a promising direction for future work. Homogeneous agent assumption. The current framework assumes homogeneous agents with shared policy structures. This limits direct applicability to heterogeneous multi-agent systems involving agents with distinct roles, capabilities, or objectives. Extending AGTRL to heterogeneous settings—through role-conditioned policies or type-specific attention heads—would broaden its scope and practical relevance. Absence of explicit communication learning. AGTRL focuses on structural coordination through graph-based representations but does not explicitly learn communication protocols. In many distributed systems, learned communication can be critical for scalability and robustness. Integrating differentiable communication channels (e.g., IC3Net, SchedNet) with graph-transformer-based coordination represents a natural extension of the proposed framework. Statistical analysis. While we have provided extensive statistical testing—including paired t-tests, Wilcoxon signed-rank tests, Friedman tests, and Nemenyi post-hoc analysis—to validate our findings, we acknowledge that more comprehensive statistical analysis with larger numbers of runs (e.g., 30+ seeds) could provide even stronger evidence. We plan to extend our statistical evaluation in future work by conducting additional runs and exploring Bayesian hypothesis testing approaches. Despite these limitations, we believe that AGTRL provides a strong foundation for scalable and robust multi-agent coordination. The challenges outlined above are not fundamental flaws but rather opportunities for future research, and we hope this work inspires further exploration at the intersection of graph-based perception, transformer attention, and multi-agent reinforcement learning.

6. Conclusion

In this work, we presented AGTRL, a graph-transformer reinforcement learning framework designed to improve coordination and decision-making in multi-agent systems. Unlike prior

approaches that apply GNNs or Transformers in isolation, AGTRL unifies these paradigms through a novel architecture that combines dynamic graph construction with explicit role indicators, adaptive multi-head self-attention for relevance-based interaction weighting, and joint optimization of individual policies and coordination weights within a single end-to-end learning pipeline. This specific integration, which is missing in existing MARL methods, directly addresses the gap between scalability and robustness in dynamic multi-agent environments. Our contributions are threefold. First, we introduced a graph-enhanced observation encoder with explicit role indicators (AG_IN and IS_AG) that resolve identity ambiguity—a design choice that proves critical for stable coordination in heterogeneous teams. Second, we integrated an adaptive graph-transformer attention module that dynamically reweighs interactions based on contextual relevance, enabling graceful degradation under communication failures—a capability that static GNN aggregation or fixed-weight Transformers lack. Third, we provided theoretical insight showing that this attention mechanism behaves as a learnable, state-dependent graph Laplacian, which helps explain its stability properties—a connection that has not been made in prior MARL literature. Through extensive experiments across SC2 micromanagement, Spread coordination, and Predator-Prey benchmarks, we demonstrated that AGTRL consistently outperforms competitive baselines—including QMIX, QTRAN, QPLEX, UPDeT, and TransfQMix—in terms of sample efficiency, convergence speed, and final performance. More importantly, AGTRL maintains robust coordination under up to 40% communication dropout, conditions where existing methods collapse. These results, together with our theoretical analysis, position AGTRL as a practical and principled foundation for developing scalable and resilient multi-agent systems in realistic, uncertain environments.

References

[1] F. L. Lewis, H. Zhang, K. Hengster-Movric, and A. Das, "Cooperative Control of Multi-Agent Systems: Optimal and Adaptive Design Approaches," Springer London, 2014.

[2] K. Zhang, Z. Yang, and T. Başar, "Multi-Agent Reinforcement Learning: A Selective Overview of Theories and Algorithms," *Handbook of Reinforcement Learning and Control*, vol. 325, no. 1, pp. 321–384, 2021.

[3] X. Wang et al., "Deep Reinforcement Learning: A Survey," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 35, pp. 5064–5078, 2024.

[4] M. S. Ibrahim and M. Hamada, "Adaptive learning framework," 2016 15th International *Conference on Information Technology Based Higher Education and Training (ITHET)*, vol. 15, no. 1, pp. 1–5, 2016.

[5] H. van Hasselt, A. Guez, and D. Silver, "Deep Reinforcement Learning with Double Q-learning," arXiv:1509.06461, Sep. 2015.

[6] C. Zhu, M. Dastani, and S. Wang, "A survey of multi-agent deep reinforcement learning with communication," *Autonomous Agents and Multi-Agent Systems*, vol. 38, no. 1, p. 4, 2024.

[7] S. Mariani, G. Cabri, and F. Zambonelli, "Coordination of Autonomous Vehicles: Taxonomy and Survey," arXiv:2001.02443, Jan. 2020.

[8] J. Liang, M. Chen, and J. Liang, "Graph External Attention Enhanced Transformer," arXiv:2405.21061, May 2024.

[9] S. Yin and G. Zhong, "LGI-GT: Graph Transformers with Local and Global Operators Interleaving," *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence (IJCAI)*, vol. 32, pp. 4504–4512, 2023.

[10] C. Wang, J. Zhao, L. Li, L. Jiao, F. Liu, and S. Yang, "Automatic Graph Topology-Aware Transformer," *IEEE Transactions on Neural Networks and Learning Systems*, vol. X, no. Y, pp. 1–15, 2024.

[11] M. Herrera, M. Pérez-Hernández, A. K. Parlikad, and J. Izquierdo, "Multi-Agent Systems and Complex Networks: Review and Applications in Systems Engineering," Processes, vol. 8, no. 3, p. 312, 2020.

[12] A. Dorri, S. S. Kanhere, and R. Jurdak, "Multi-Agent Systems: A Survey," *IEEE Access*, vol. 6, no. 1, pp. 28573–28593, 2018.

[13] M. Mes and B. Gerrits, "Multi-agent Systems," in *Operations, Logistics and Supply Chain Management*, vol. X, H. Zijm, M. Klumpp, A. Regattieri, and S. Heragu, Eds., Springer International Publishing, pp. 611–636, 2019.

[14] K. Keogh and L. Sonenberg, "Designing Multi-Agent System Organisations for Flexible Runtime Behaviour," *Applied Sciences*, vol. 10, no. 15, p. 5335, 2020.

[15] E. Bastami, A. Mahabadi, and E. Taghizadeh, "A gravitation-based link prediction approach in social networks," *Swarm and Evolutionary Computation*, vol. 44, pp. 176–186, 2019.

[16] G. Zhang et al., "G-Designer: Architecting Multi-agent Communication Topologies via Graph Neural Networks," arXiv:2410.11782, 2024.

[17] Z. Fang, F. Ke, J. Y. Han, Z. Feng, and T. Cai, "Graph Enhanced Reinforcement Learning for Effective

Group Formation in Collaborative Problem Solving," arXiv:2403.10006, 2024.

[18] J. N. Foerster, Y. M. Assael, N. de Freitas, and S. Whiteson, "Learning to Communicate with Deep Multi-Agent Reinforcement Learning," arXiv:1605.06676, 2016.

[19] L. Ratnabala, A. Fedoseev, R. Peter, and D. Tsetserukou, "MAGNNET: Multi-Agent Graph Neural Network-based Efficient Task Allocation for Autonomous Vehicles with Deep Reinforcement Learning," arXiv:2502.02311, 2025.

[20] J Z. Jia, J. Li, X. Qu, and J. Wang, "Enhancing Multi-Agent Systems via Reinforcement Learning with LLM-based Planner and Graph-based Policy," arXiv:2503.10049, 2025.

[21] F. L. Da Silva and A. H. R. Costa, "Transfer Learning for Multiagent Reinforcement Learning Systems," Springer International Publishing, 2021.

[22] W. Li, L. Ni, J. Wang, and C. Wang, "Collaborative representation learning for nodes and relations via heterogeneous graph neural network," *Knowledge-Based Systems*, vol. 255, p. 109673, 2022.

[23] T. He, Y. Liu, Y.-S. Ong, X. Wu, and X. Luo, "Polarized message-passing in graph neural networks," Artificial Intelligence, vol. 331, p. 104129, 2024.

[24] C.-W. Leung, S. Hu, and H.-F. Leung, "Self-Play or Group Practice: Learning to Play Alternating Markov Game in Multi-Agent System," 2*020 25th International Conference on Pattern Recognition (ICPR)*, vol. 25, pp. 9234–9241, 2021.

[25] K. Hu et al., "A review of research on reinforcement learning algorithms for multi-agents," Neurocomputing, vol. 599, p. 128068, 2024.

[26] J. M. Górriz et al., "Computational approaches to Explainable Artificial Intelligence: Advances in theory, applications and trends," *Information Fusion*, vol. 100, p. 101945, 2023.

[27] T. Rashid, M. Samvelyan, C. Schroeder, G. Farquhar, J. Foerster, and S. Whiteson, "QMIX: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning," in *Proceedings of the 35th International Conference on Machine Learning (ICML)*, pp. 4295–4304, 2018.

[28] 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 *Proceedings of the 36th International Conference on Machine Learning (ICML)*, pp. 5887–5896, 2019.

[29] J. Wang, Z. Ren, T. Liu, Y. Yu, and C. Zhang, "QPLEX: Duplex Dueling Multi-Agent Q-Learning," in *International Conference on Learning Representations (ICLR)*, 2021.

[30] S. Hu, F. Zhu, X. Chang, and X. Liang, "UPDeT: Universal Multi-Agent Reinforcement Learning via Policy Decoupling with Transformers," in *International Conference on Learning Representations (ICLR)*, 2021.

[31] M. Gallici, M. Martin, and I. Masmitja, "TransfQMix: Transformers for Leveraging the Graph Structure of Multi-Agent Reinforcement Learning Problems," *Proceedings of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2023)*, vol. 22, pp. 1679–1687, 2023.

[32] J. Huang, J. Su, and Q. Chang, "Graph neural network and multi-agent reinforcement learning for machine-process-system integrated control to optimize production yield," *Journal of Manufacturing Systems*, vol. 64, pp. 81–93, 2022.

[33] Z. Zhang, B. He, B. Cheng, and G. Li, "Bridging Training and Execution via Dynamic Directed Graph-Based Communication in Cooperative Multi-Agent Systems," in *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI 2025)*, vol. 39, no. 22, pp. 23395–23403, 2025.

[34] S. Dolan, S. Nayak, J. J. Aloor, and H. Balakrishnan, "Asynchronous Cooperative Multi-Agent Reinforcement Learning with Limited Communication," *Proceedings of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025)*, vol. 24, pp. 1–3, 2025.

[35] Z. Ning and L. Xie, "A survey on multi-agent reinforcement learning and its application," *Journal of Automation and Intelligence*, vol. 3, no. 2, pp. 73–91, 2024.

[36] M. Tao, Q. Li, and J. Yu, "Multi-Objective Dynamic Path Planning with Multi-Agent Deep Reinforcement Learning," *Journal of Marine Science and Engineering*, vol. 13, no. 1, p. 20, 2024.

[37] J. Zhou et al., "Graph neural networks: A review of methods and applications," *AI Open*, vol. 1, pp. 57–81, 2020.

[38] O. Vinyals et al., "StarCraft II: A New Challenge for Reinforcement Learning," arXiv:1708.04782, Aug. 2017.

[39] K. Wan, D. Wu, Y. Zhai, B. Li, X. Gao, and Z. Hu, "Multi-Agent Actor-Critic for Mixed Cooperative-Competitive," *Entropy*, vol. 23, no. 11, p. 1433, 2021.

[40] T. Weng, H. Yang, C. Gu, J. Zhang, P. Hui, and M. Small, "Predator-prey games on complex networks," *Communications in Nonlinear Science and Numerical Simulation*, vol. 79, p. 104911, 2019.

[41] R. Hosseinzadeh and M. Sadeghzadeh, "Attention Mechanisms in Transformers: A General Survey," *Journal of AI and Data Mining*, vol. 13, no. 3, pp. 359–368, Jul. 2025.

یادگیری تقویتی مبتنی بر گراف-ترنسفورمر برای هماهنگی چندعاملی مقیاس‌پذیر

سجاد بسطامی[۱]، محمدباقر دولتشاهی[۲]، روجیار پیرمحمدیانی[۱،*] و سیده زهرا موسوی[۲]

[۱] گروه مهندسی کامپیوتر، دانشگاه کردستان، سنندج، ایران.

[۲] گروه مهندسی کامپیوتر، دانشگاه لرستان، خرم‌آباد، ایران.

ارسال ۲۰۲۶/۰۴/۲۴؛ بازنگری ۲۰۲۶/۰۷/۱۴؛ پذیرش ۲۰۲۶/۰۷/۲۴

چکیده:

یادگیری تقویتی چندعاملی، امروزه یکی از ستون‌های اصلی در طراحی ربات‌های همکار، سیستم‌های خودران و کنترل‌های توزیع‌شده به حساب می‌آید. با این حال، روش‌هایی که تاکنون در این حوزه ارائه شده‌اند، وقتی پای مقیاس‌های بزرگ، محیط‌های پویا و تغییرات مداوم در تعاملات به میان می‌آید، کم‌می‌آورند و با محدودیت‌های اساسی دست‌وپنجه نرم می‌کنند. برای عبور از این موانع، چارچوبی جدید پیشنهاد کرده‌ایم به نام «یادگیری تقویتی گراف-ترنسفورمر تطبیقی» یا به اختصار AGTRL؛ روشی که مدل‌سازی روابط مبتنی بر گراف را با توانمندی ترانسفورمر «توجه» به تعاملات مهم، را ترکیب می‌کند تا هماهنگی بین عامل‌ها را در سیستم‌های بزرگ‌مقیاس، انعطاف‌پذیر و سازگار با شرایط نماید. در AGTRL، درک محیط از طریق گراف و هماهنگی از طریق سازوکار توجه، در یک مسیر پردازشی یکپارچه به هم متصل می‌شوند؛ به‌طوری که نقش هر عامل به‌درستی رمزگذاری شده و وزن تعاملات بر اساس بافت و زمینه‌ی موقعیت، به‌طور پویا تنظیم می‌گردد. این ترکیب هوشمندانه، که در کارهای قبلی کمتر دیده شده بود، پلی بین دو نیاز به ظاهر متضاد است: از یک سو مقیاس‌پذیری و از سوی دیگر مقاوم در برابر آشفتگی‌های محیطی. سیستم ما با ساختن یک گراف پویا از روابط بین عامل‌ها و بهره‌گیری از خودتوجهی چندسر، در هر لحظه تشخیص می‌دهد کدام تعاملات از همه مهم‌ترند و روی آن‌ها متمرکز می‌شود؛ همین ویژگی باعث می‌شود که عملکرد سیستم حتی در شرایط نامساعد هم پایدار بماند. برای سنجش مقاوم، روش خود را در سناریوهای قطعی ارتباط (تا ۴۰٪ حذف پیوندها) و حذف تدریجی یال‌ها آزمایش کرده‌ایم و معیارهایی مثل پاداش جمع‌شده در هر مرحله و نرخ پیروزی را اندازه گرفته‌ایم. علاوه بر این، سازوکاری تطبیقی در نظر گرفته‌ایم که به‌طور هوشمندانه بین پایداری و عملکرد تعادل برقرار می‌کند تا فرایند یادگیری حتی وقتی ارتباطات دچار اختلال است یا محیط نامشخص، باعث افت کیفیت نشود. معماری سیاست‌گذاری ما نیز با استفاده از گراف، هم سیاست‌های فردی هر عامل و هم هماهنگی جمعی را از طریق همان وزن‌دهی‌های مبتنی بر توجه، به‌صورت هم‌زمان بهینه می‌کند. نتایج آزمون‌های گسترده در محیط‌های شناخته‌شده‌ای مثل سناریوهای ریزمدیریت StarCraft II، وظیفه همکاری پخش (Spread) و مأموریت‌های رقابتی شکارچی-شکار (Predator-Prey) نشان می‌دهد که AGTRL از نظر سرعت یادگیری، کارایی با داده‌های محدود و مقاومت در برابر اختلالات، به‌وضوح از روش‌های روز همتراز خود بهتر عمل می‌کند. به‌طور متوسط، AGTRL همگرایی را ۳۲٪ سریع‌تر می‌کند و با وجود ۴۰٪ قطعی ارتباط، عملکردی باثبات از خود نشان می‌دهد؛ موضوعی که ثابت می‌کند این روش برای استفاده در دنیای واقعی و در محیط‌های پویای چندعاملی، انتخابی کاملاً عملی و قابل اعتماد است.

کلمات کلیدی: یادگیری تقویتی چندعاملی، شبکه‌های عصبی گراف، سازوکارهای توجه ترانسفورمر، هماهنگی تطبیقی، سیستم‌های چندعاملی مقیاس‌پذیر.