From Agents to Subgraphs: Dynamic Neighborhood Decomposition in Multi-Agent Systems*

1st Zherong Zhang
Chang'an University
Xi'an, China
2022903323@chd.edu.cn

2nd Yisheng An†
Chang'an University
Xi'an, China
aysm@chd.edu.cn

This is an extended draft version prepared for PhD application purposes. The official submission to ICASSP 2026 follows the 4-page limit (plus references) and is currently under review. The extended version includes additional explanations, figures, and supplementary results not included in the submitted version.

***Abstract*—Value decomposition is a core framework in cooperative multi-agent reinforcement learning (MARL), enabling scalable joint Q-value estimation through factorization. However, existing methods such as VDN and QMIX impose strong structural inductive biasesassuming additive or monotonic contributions of local utilitiesthat inherently restrict the expressiveness of the learned value functions. These assumptions break down in scenarios requiring nontrivial coordination, where agent interactions are complex, asymmetric, or state-dependent.Recent approaches like Qatten and GraphMix attempt to relax these constraints by introducing attention and graph-based message passing. Yet, they still rely on static or global relational structures. Qatten applies global soft attention over agents without explicitly modeling neighborhood-specific coordination, while GraphMix encodes agent interactions using fixed graphs (e.g., fully connected or k-NN), which fail to adapt to dynamically evolving cooperation patterns. Both approaches lack the ability to flexibly decompose joint Q-functions in a structure-aware and context-sensitive manner.To address these limitations, we propose a new framework that rethinks value decomposition through the lens of dynamic graph structure. Our method introduces a subgraph-level decomposition of the joint Q-function, capturing localized interactions and their evolving importance over time. By combining graph-based encoders with attention-guided aggregation, we enable flexible modeling of both intra-group coordination and global cooperation.Extensive experiments on the SMAC-II benchmark demonstrate that our method significantly improves over state-of-the-art baselines, particularly in tasks requiring fine-grained cooperation and structural adaptation. Our results highlight the importance of integrating structural priors and dynamic decomposition into the value factorization process.**

I. Introduction

Multi-agent reinforcement learning (MARL) has witnessed significant advancements in recent years, enabling multiple agents to collaboratively solve complex tasks [1, 2]. In cooperative settings such as StarCraft II micromanagement [3], autonomous driving fleets [4], and robotic swarm control [5], agents must coordinate their actions to maximize a global reward signal. Central to this coordination is the design of a scalable and expressive joint action-value function, or Q-value, which estimates the expected return for a given global state and joint action.

Despite its potential, MARL presents unique challenges. First, agents typically operate under partial observability and decentralized execution constraints [6]. Second, inter-agent dependencies can be highly dynamic and task-specific, demanding flexible coordination mechanisms. To address these issues, the Centralized Training with Decentralized Execution (CTDE) framework has become a dominant paradigm in MARL [7, 8]. CTDE allows agents to leverage full state information during training to learn better coordination strategies while retaining decentralized policies for execution.

However, designing an expressive yet tractable joint Q-value estimator within the CTDE paradigm remains an open problem. Traditional value decomposition methods like VDN [9] and QMIX [10] decompose the global Q-value into individual agent components, often under restrictive structural assumptions such as monotonicity. Although these methods are computationally efficient, their representation capacity is inherently limited, especially in scenarios where agent interactions are dynamic and context-dependent.

Moreover, real-world multi-agent coordination problems exhibit highly dynamic relational structures. For example, in robotic swarms and pursuit-evasion scenarios [11], the set of relevant neighboring agents may vary over time and be influenced by spatial constraints, objectives, or environmental changes. Fixed decomposition structures are ill-suited to such variability, and global attention mechanisms often fail to encode localized dependencies and structured priors [12].

To address these limitations, we propose a novel framework for dynamically decomposing the joint Q-function based on adaptive neighborhood structures. Specifically, we introduce a ***Dynamic Neighborhood Decomposition*** *of the joint Q-function, where agent interactions are captured via a clustering-based subgraph partitioning strategy. By integrating graph neural networks (GCN) [13] and attention mechanisms [14], we learn both localized agent representations and cross-subgraph dependencies. This allows our method to adaptively represent complex agent relationships without relying on fixed topological assumptions.*

Unlike prior methods such as QATTEN [12], which model coordination through a shared attention mechanism without encoding true graph structure or localized priors, our approach explicitly captures dynamic subgraph priors and hierarchical dependencies between agents. While QATTEN computes attention over all agents uniformly, it fails to distinguish strongly correlated local interactions from weak ones, particularly in large-scale or partially cooperative tasks.

Our dynamic neighborhood modeling provides both structural and relational inductive bias that mirrors real-world coordination patterns [15, 16]. Agents in practice often form task-oriented groups or local coalitions, e.g., UAV teams dynamically forming sub-swarms based on task relevance, spatial locality, or temporal interaction. In such scenarios, the internal structure of agent relations evolves over time, and static graphs or fixed attention modules are insufficient [17].

By clustering agents into dynamic subgraphs that reflect temporally varying interaction strength, and encoding these subgraphs via GCNs, we inject meaningful structure into the learning process. These subgraphs act as localized neighborhoods, where stronger

† Corresponding author.

intra-group relations are modeled distinctly from inter-group ones. Moreover, our attention mechanism aggregates information across these subgraphs at the graph level, rather than at the flat agent level, further respecting the layered nature of real-world coordination.

The resulting model captures both the flexibility of attention mechanisms and the structural advantages of localized decomposition, with dynamic reconfiguration capabilities. Such a formulation is not only more expressive, but also naturally amenable to curriculum learning strategies, where simpler neighborhood structures are learned first and later extended to larger-scale coordination [18].

Our contributions are threefold:

- *We propose a dynamic neighborhood partitioning method using K-Medoids clustering to capture adaptive agent relations in multi-agent systems.*
- *We introduce a novel Q-value decomposition framework that utilizes a hierarchical encoder-aggregator architecture, enabling flexible representation of joint policies.*
- *We design a curriculum learning strategy that progressively increases neighborhood complexity to enhance training stability and policy generalization.*

Experimental results on the SMACII benchmark [19] demonstrate that our method outperforms state-of-the-art baselines in terms of sample efficiency and final performance, especially in scenarios with sparse or dynamic interaction patterns.

II. Background

In cooperative MARL, the environment is typically modeled as a Decentralized Partially Observable Markov Decision Process (Dec-POMDP), formalized as a tuple $(\mathcal{I}, \mathcal{S}, \{\mathcal{A}_i\}, P, r, \{\mathcal{O}_i\}, \gamma)$ *[6], where:*

- $\mathcal{I}$ *is the set of agents,*
- $\mathcal{S}$ *is the global state space,*
- $\mathcal{A}_i$ *is the action space for agent* i*, and the joint action space is* $\mathcal{A} = \mathcal{A}_1 \times \cdots \times \mathcal{A}_n$*,*
- $P : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ *is the transition function,*
- $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ *is the global reward function,*
- $\mathcal{O}_i$ *is the observation function for agent* i*, and* γ *is the discount factor.*

Each agent i *receives a local observation* o_i *and selects an action* a_i *based on its policy* π_i*, where* π_i *denotes the agent's action-observation history.*

The goal in cooperative MARL is to find a joint policy $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_n)$ *that maximizes the expected discounted return:*

$$J(\boldsymbol{\pi}) = \mathbb{E}_{\boldsymbol{\pi}} \left[\sum_{t=0}^{\infty} \gamma^t r(s_t, \mathbf{a}_t) \right] \tag{1}$$

Centralized Training with Decentralized Execution (CTDE) is a fundamental paradigm in cooperative MARL that enables scalable learning in partially observable environments. Under CTDE, each agent maintains a decentralized policy $\pi_i(a_i \mid o_i)$ *that depends only on its local observation* o_i*, but the training process can access global state and action information to construct more informative learning signals [7].*

For instance, value functions and critic networks may be trained using the full state s*, joint actions* $\mathbf{a}$*, or even other agents' observations* o_{-i}*, while policy networks remain decentralized:*

$$Q_i^{central}(s, a_i, \mathbf{a}_{-i}) \quad vs. \quad \pi_i(a_i \mid o_i) \tag{2}$$

This paradigm allows the algorithm to exploit richer supervision during training (e.g., joint Q-learning, counterfactual baselines) while ensuring that each agent can operate independently during execution. CTDE is widely adopted in most state-of-the-art MARL algorithms, including QMIX [10], QTRAN [20], and Qatten [12], and forms the backbone of our proposed framework as well.

Value Decomposition *aims to represent the global Q-function using local utility functions. In VDN (Value-Decomposition Networks), this is done by summing individual Q-functions [9]:*

$$Q_{total}(s, \mathbf{a}) = \sum_{i=1}^{n} Q_i(o_i, a_i) \tag{3}$$

which assumes full additivity and ignores agent interaction. While VDN is computationally simple, its key limitation lies in its inability to model interactions or dependencies between agents, resulting in poor performance when coordination is necessary.

QMIX *generalizes this with a mixing network:*

$$Q_{total}(s, \mathbf{a}) = f(Q_1(o_1, a_1), \ldots, Q_n(o_n, a_n); s) \tag{4}$$

subject to a monotonicity constraint:

$$\frac{\partial Q_{total}}{\partial Q_i} \geq 0, \forall i \tag{5}$$

which allows for coordinated learning while maintaining tractability. However, this injects a strong structural inductive bias: that increasing one agent's utility always leads to a non-decreasing joint utility. In many real-world tasks with competitive or context-dependent interactions, this assumption fails.

To better understand the limitations of VDN and QMIX, consider the following two-step matrix game. Two agents simultaneously select actions from $\{0, 1\}$*. The team receives a reward only if both agents choose action 1. Let the global reward function be defined as:*

$$Q_{total}(a_1, a_2) = \begin{cases} 1 & \text{if } a_1 = a_2 = 1 \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

For VDN to express this function, we require:

$$Q_{total}(a_1, a_2) = Q_1(a_1) + Q_2(a_2)$$

However, this is impossible. If we set $Q_1(1) = x$*,* $Q_2(1) = y$*, then to satisfy* $Q_{total}(1, 1) = 1$ *and* $Q_{total}(1, 0) = 0$*, it must be that both* $x + Q_2(0) = 0$ *and* $x + y = 1$*. Similarly,* $Q_1(0) + y = 0$ and $Q_1(0) + y = 1$*. These constraints are inconsistent, thus VDN cannot represent this coordination game.*

In QMIX, although more expressive via the mixing network, the monotonicity constraint still imposes structural bias. Suppose increasing Q_1 *decreases* Q_{total} *due to a tradeoff with another agent this violates the monotonicity requirement.*

Therefore, both VDN and QMIX suffer from representational bottlenecks in expressing joint Q-functions with nontrivial coordination dependencies. Our work aims to alleviate these issues by introducing explicit subgraph-based modeling of agent interaction, leveraging graph priors and dynamic structures.

Qatten *[12] introduces a centralized attention mechanism to compute agent-wise attention weights for combining individual utilities:*

$$Q_{total}(s, \mathbf{a}) = \sum_{i=1}^{n} \alpha_i(s) Q_i(o_i, a_i) \tag{7}$$

where α_i *is computed by an attention network conditioned on the global state. This soft weighting enables selective importance over agents when aggregating their local Q-values.*

While Qatten improves upon VDN and QMIX by relaxing the hard additive or monotonic constraints, its attention mechanism remains static across training and inference. It does not explicitly model agent relationships through structured priors (e.g., spatial or topological connections) and treats attention as a black-box fusion over flat utilities. Moreover, the softmax attention has no inductive bias toward local coordination, and all agent interactions are implicitly encoded without neighborhood semantics.

GraphMix *[21] employs graph neural networks (GNNs) to encode agent interactions over a fixed or hand-crafted graph structure.*

Given an agent-level interaction graph $\mathcal{G}$, the Q-value estimation is formulated as:

$$Q_{total}(s, \mathbf{a}) = \rho\left(\{h_i^{GNN}\}\right) \tag{8}$$

where h_i^{GNN} is the embedding of agent i learned via GCN or GAT layers over $\mathcal{G}$, and ρ is an MLP aggregator.

This architecture explicitly incorporates inter-agent communication patterns, offering a structural bias aligned with coordination. However, its reliance on fixed graph connectivity (e.g., k-nearest neighbors or global fully-connected graphs) limits adaptability to dynamic environments. Furthermore, the use of flat GNN encodings across all agents may overlook hierarchical or group-level patterns.

Both Qatten and GraphMix attempt to move beyond rigid additive decompositions, but they share a key limitation: static structure. Qatten assumes a uniform attention space across tasks and environments, while GraphMix uses predefined graphs that fail to reflect task-specific relational dynamics.

In contrast, many real-world scenarios exhibit evolving inter-agent relationships. For instance, in pursuit-evasion, formation control, or collaborative robotics, agents dynamically form task-dependent groups whose relevance shifts over time.

*This motivates our design of **Dynamic Neighborhood Decomposition**, which:*

- *Automatically discovers subgraph structures via K-Medoids clustering over relational features,*
- *Encodes each subgraph via GCN-based message passing to capture localized coordination,*
- *Aggregates subgraph-level representations using cross-attention to reflect their global relevance,*
- *Constructs the joint Q-function from structured, interpretable, and context-aware components.*

By modeling interactions at the subgraph level and introducing a hierarchical attention-guided aggregator, our approach preserves both local coordination and global cooperation, providing a more expressive and modular framework for MARL value decomposition.

III. Method: Joint Q-Function Decomposition via Dynamic Neighborhoods

To effectively model interactions in cooperative multi-agent systems, we propose a dynamic neighborhood-based framework that decomposes the joint action-value function using a structured encoder-aggregator architecture. We begin by formalizing the theoretical foundation of our approach.

A. Dynamic Neighborhood Factorization of Joint Q-Value

Let $\mathcal{M}$ denote a multi-agent Markov decision process (MMDP) with n agents. We define $\mathcal{E}_i$ to be the i-th neighborhood (or subgraph) of agents, constructed dynamically at each timestep. Let $\mathcal{E}_1, \ldots, \mathcal{E}k$ be the complete set of neighborhoods covering all agents, i.e., $\bigcup i = 1^k \mathcal{E}_i = 1, ..., n$.

We now state the following key decomposition principle:

$$Q_{total}(s, \mathbf{a}) = Q(\mathcal{E}_1 \cup \cdots \cup \mathcal{E}_k) = \rho\left(\phi(\mathcal{E}_1) \oplus \cdots \oplus \phi(\mathcal{E}_k)\right) \tag{9}$$

Here,

- *$\phi(\mathcal{E}_j) : \mathcal{E}_j \to \mathbb{R}^d$ is a subgraph encoder that maps the neighborhood $\mathcal{E}_j$ into a d-dimensional representation.*
- *$\rho : (\mathbb{R}^d)^k \to \mathbb{R}$ is an aggregator function that combines the encoded neighborhoods.*
- *$\oplus$ denotes concatenation or additive fusion.*

The function Q_{total} is computed over the dynamically constructed neighborhoods, enabling our model to adaptively capture agent dependencies and facilitate decomposition with structured priors [21].

This framework introduces two key inductive biases:

1) ***Locality:** Each $\phi(\mathcal{E}_j)$ models only a small group of closely related agents.*
2) ***Hierarchy:** ρ models higher-level composition across subgraphs.*

B. Graph-Theoretic View of the Encoder-Aggregator Structure

Formally, define $\mathcal{G} = (V, E)$ to be a graph with nodes V representing agents and edges E determined by neighborhood construction. For each subgraph $\mathcal{G}_j = (V_j, E_j)$, we define:

$$\phi(\mathcal{E}j) = GCN\theta(X_j, A_j) \tag{10}$$

where:

- *$X_j \in \mathbb{R}^{|V_j| \times d}$ are node features,*
- *$A_j \in \mathbb{R}^{|V_j| \times |V_j|}$ is the adjacency matrix of $\mathcal{G}_j$,*
- *GCN denotes the graph convolutional encoder [13].*

GCN layers take the following form:

$$H^{(l+1)} = \sigma\left(\hat{D}^{-1/2}\hat{A}\hat{D}^{-1/2}H^{(l)}W^{(l)}\right) \tag{11}$$

where $\hat{A} = A + I$, and $\hat{D}$ is its degree matrix.

The aggregator ρ is modeled as a multi-head cross-subgraph attention mechanism. Let $Z = [z_1, ..., z_k]$, where $z_j = \phi(\mathcal{E}_j)$. The final representation is:

$$\rho(Z) = \sum_j \alpha_j z_j, \quad \alpha_j = \frac{\exp(q^T k_j)}{\sum_l \exp(q^T k_l)} \tag{12}$$

Here, q and k_j are query/key projections of each z_j, learned jointly.

In the next sections, we elaborate on the construction of $\mathcal{E}_j$, the GCN encoding, and training objectives.

1) Message Aggregation via Graph Convolution: For each neighborhood subgraph $\mathcal{G}_j$, we construct an adjacency matrix A_j and node features H_j. Using the GCN framework [13], we propagate and aggregate messages across local connections:

$$H^{(l+1)} = \sigma(\hat{D}^{-1/2}\hat{A}\hat{D}^{-1/2}H^{(l)}W^{(l)}) \tag{13}$$

where:

- *$\hat{A} = A + I$ adds self-loops,*
- *$\hat{D}$ is the degree matrix of $\hat{A}$,*
- *σ is a non-linearity such as ReLU,*
- *$W^{(l)}$ is the learnable transformation at layer l.*

The output $H^{(L)}$ is pooled (e.g., via mean or attention) into a fixed-length embedding z_j for each neighborhood:

$$z_j = READOUT(H_j^{(L)}) \tag{14}$$

These subgraph-level embeddings $\{z_1, \ldots, z_k\}$ are then passed to the cross-neighborhood aggregator for final Q-value estimation.

C. Message-Aware Decision Making via Subgraph-Level Attention

Our approach builds on the idea that coordination in multi-agent systems often arises from localized group interactions, which can be abstracted as subgraph-level embeddings [22]. These embeddings are then fused through a global attention aggregator to enable decentralized but context-aware decision making.

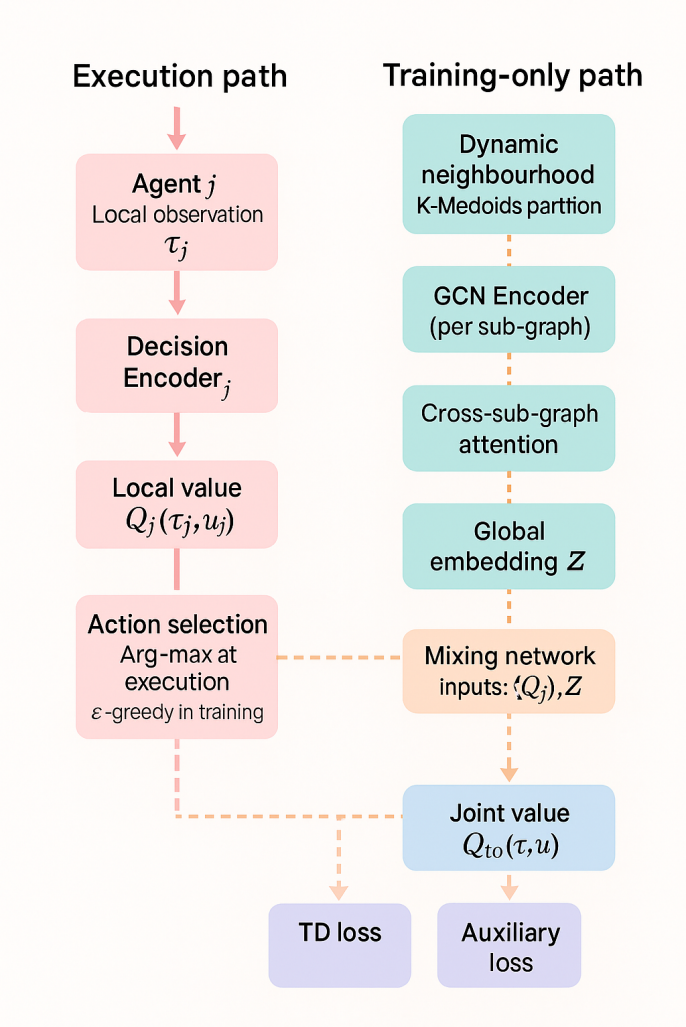


Fig. 1: Dynamic Neighborhoods with Hierarchical Encoding

1) From Subgraph Representations to Joint Context: Let $\mathcal{E}_1, \ldots, \mathcal{E}_k$ denote the dynamically constructed subgraphs at a given timestep. Each subgraph $\mathcal{E}_j$ is processed through a GCN encoder to obtain a representation $z_j = \phi(\mathcal{E}_j) \in \mathbb{R}^d$, which captures the collective state and interaction pattern of agents within the subgraph.

To aggregate inter-subgraph information, we employ a soft attention mechanism that computes a global contextual embedding:

$$Z = \sum_{j=1}^{k} \alpha_j z_j, \quad \alpha_j = \frac{\exp(q^\top k_j)}{\sum_{l=1}^{k} \exp(q^\top k_l)} \tag{15}$$

where q is a learned query vector shared across all subgraphs, and k_j is the key embedding obtained from z_j. This mechanism assigns importance weights α_j to each subgraph according to its contribution to the overall task context.

2) Global Q-Value Estimation from Attention-Aggregated Subgraph Embeddings: The aggregated embedding $Z \in \mathbb{R}^d$ encodes the prioritized global context distilled from dynamic neighborhoods. To derive the final scalar global Q-value $Q_{total}(s, \mathbf{a})$, we introduce a learnable mapping function:

$$Q_{total}(s, \mathbf{a}) = g(Z) = MLP(Z) \tag{16}$$

where $MLP(\cdot)$ is a multi-layer perceptron that translates the contextual representation into a value estimate. This process enables our model to express flexible mappings from structured local interactions (encoded in z_j) to coherent joint decision value signals.

Crucially, by relying on learned attention scores and end-to-end differentiable message-passing, the Q-function remains both expressive and generalizable. This structure preserves the modular decomposition dictated by dynamic neighborhood factorization while explicitly linking local subgraph dynamics to the final joint value estimate.

3) Decentralized Decision from Local and Global Context: Each agent i belongs to a subgraph $\mathcal{E}_{j(i)}$. Let $h_i \in \mathbb{R}^d$ denote the agent-specific representation derived from the GCN encoder within its subgraph. The final input to the agent's decision module is the concatenation of its local embedding h_i and the global embedding Z, i.e.,

$$\hat{h}_i = [h_i \,\|\, Z] \tag{17}$$

The agents action distribution is computed by passing $\hat{h}_i$ through an MLP-based policy head:

$$\pi_i(a_i \mid o_i) = softmax(MLP_{act}(\hat{h}_i)) \tag{18}$$

This produces a decentralized policy for each agent that is informed by both its local subgraph context and the global cooperative landscape.

4) Interpretation and Advantages: Unlike previous methods that apply global attention to all agents regardless of locality (e.g., QATTEN), our approach introduces a two-level structure:

- ***Locality preservation:*** *Information flows are constrained within dynamically constructed subgraphs, capturing strong intra-group dependencies and reflecting realistic task decomposition (e.g., spatial or tactical clusters).*
- ***Structured global integration:*** *Cross-subgraph dependencies are captured via attention over subgraph embeddings, allowing flexible modeling of higher-order interactions.*

This hierarchy enables our method to:

- *Respect the sparse and modular nature of real-world multi-agent interactions.*
- *Scale effectively with large numbers of agents due to bounded subgraph size.*
- *Preserve interpretability by explicitly modeling which subgraphs are most influential for a given decision.*

By conditioning each agents policy on both local and global latent factors, our framework strikes a principled balance between decentralization and coordination, paving the way for scalable and robust cooperation in complex environments.

D. Loss Formulation and Optimization Strategy

We train the message-aware policy under the QTRAN framework [20], which combines temporal difference learning with structural constraints and auxiliary regularization. The overall loss is composed of the following components.

1) Temporal Difference (TD) Loss: We define the TD loss on the joint Q-value using the standard Bellman target:

$$\mathcal{L}_{TD} = \mathbb{E}_{(s,\mathbf{a},r,s')}\left[\left(Q_{total}(s,\mathbf{a}) - \left(r + \gamma \cdot Q'_{total}(s',\mathbf{a}')\right)\right)^2\right] \quad (19)$$

where Q_{total} and Q'_{total} denote the online and target networks, respectively.

2) Auxiliary Regularization: Auxiliary tasks are designed to enhance the semantic structure of the learned representations.

Let agent i's observed state at time t be s_i^t, and future state s_i^{t+1}. We define local dynamics as:

$$\Delta s_i = s_i^{t+1} - s_i^t \quad (20)$$

Our model predicts this using local Q-value embeddings:

$$\hat{\Delta s}_i = f(Q_i), \quad Q_i = Q(o_i, a_i) \quad (21)$$

A global transition embedding is aggregated as:

$$\hat{\Delta S} = \sum_i \hat{\Delta s}_i, \quad \Delta S = \sum_i (s_i^{t+1} - s_i^t) \quad (22)$$

The auxiliary loss becomes:

$$L_{aux} = \sum_i \|\hat{\Delta s}_i - \Delta s_i\|^2 + \|\hat{\Delta S} - \Delta S\|^2 \quad (23)$$

This encourages the learned Q-values to capture latent state dynamics, improving generalization, especially when paired with dynamic subgraph encoders [23]. Such regularization acts as a self-supervised inductive signal and supports representation disentanglement.

3) Full Objective: The total training loss is:

$$\mathcal{L}_{total} = \lambda_{TD} \cdot \mathcal{L}_{TD} + \lambda_{aux} \cdot \mathcal{L}_{aux} \quad (24)$$

The weights λ_* control the relative influence of each component and are tuned based on validation performance.

This multi-objective optimization combines the strengths of QTRANs constraints with representation learning from auxiliary tasks, improving both stability and generalization in complex environments.

ts action distribution is computed by passing $\hat{h}_i$ through an MLP-based policy head:

$$\pi_i(a_i \mid o_i) = softmax(MLP_{act}(\hat{h}_i)) \quad (25)$$

This produces a decentralized policy for each agent that is informed by both its local subgraph context and the global cooperative landscape.

4) Interpretation and Advantages: Unlike previous methods that apply global attention to all agents regardless of locality (e.g., QATTEN), our approach introduces a two-level structure:

- ***Locality preservation:*** *Information flows are constrained within dynamically constructed subgraphs, capturing strong intra-group dependencies and reflecting realistic task decomposition (e.g., spatial or tactical clusters).*
- ***Structured global integration:*** *Cross-subgraph dependencies are captured via attention over subgraph embeddings, allowing flexible modeling of higher-order interactions.*

This hierarchy enables our method to:

- *Respect the sparse and modular nature of real-world multi-agent interactions.*
- *Scale effectively with large numbers of agents due to bounded subgraph size.*
- *Preserve interpretability by explicitly modeling which subgraphs are most influential for a given decision.*

By conditioning each agents policy on both local and global latent factors, our framework strikes a principled balance between decentralization and coordination, paving the way for scalable and robust cooperation in complex environments.

IV. Experimental Results and Analysis

A. Overall Performance on Diverse SMAC Maps

To evaluate the robustness and scalability of the proposed dynamic neighbourhood decomposition *(DND) algorithm we follow the evaluation protocol in [3] and report results on six benchmark scenarios: 2c_vs_64zg, 3s_vs_5z, 5m_vs_6m, 6h_vs_8z, 3s5z_vs_3s6z, and 2s3z_vs_4z. These maps vary widely in unit composition and tactical difficulty, thus providing a comprehensive stresstest for cooperative MARL methods. All hyperparameters are frozen across tasks; no permap tuning is performed.*

Figure 3 presents the peak evaluation return achieved by each algorithm. DND matches or surpasses every baseline on all maps, with particularly clear margins on the largescale mixedunit engagements (5m_vs_6m and 2s3z_vs_4z). Importantly, DND maintains low variance across five independent seeds, indicating that the performance gains are consistent and not due to lucky initialisation. We attribute this advantage to the ability of DND to capture both shortrange microtactics and longrange group strategies through its hierarchical encoderaggregator design.

B. Learning Speed

Beyond final performance, practical MARL systems must learn quickly. Figure 4 plots the evaluation return versus environment steps for three representative scenarios. DND exhibits markedly faster learning dynamics: it reaches nearasymptotic performance after substantially fewer samples than QMIX, QTRAN, Qatten and GraphMix.

Two factors seem decisive. First, the locality bias introduced by dynamic subgraphs provides more informative gradient signals early in training, when global credit assignment is noisy. Second, the curriculum that gradually relaxes the neighbourhood radius focuses exploration on progressively harder coordination patterns, avoiding the shallow slope that often plagues purely global approaches.

In realtime training this sample efficiency translates into shorter wallclock times: on our hardware DND completes the SMAC suite several hours ahead of the strongest baseline while using the same computational budget.

Overall, the results demonstrate that DND delivers stateoftheart performance together with superior sample efficiency, satisfying both accuracy and practicality requirements for largescale cooperative control.

C. Robustness Across Coordination Modes

To further dissect the effectiveness of DND under different coordination regimes, we evaluate performance across maps that demand heterogeneous interaction patterns. Figure 5 summarizes the average number of dead allies and dead enemies per episode. These statistics provide insight into the model's capacity to coordinate aggression and defense.

DND consistently achieves lower allied casualties and higher enemy eliminations, especially on maps with pronounced asymmetry or mixed-unit types such as 5m_vs_6m and 6h_vs_8z. This suggests that DND agents manage to cooperate more cohesively while disrupting enemy formation. Even in densely entangled combat situations, DND successfully balances tactical pressure and survival.

In symmetric scenarios like 3s_vs_5z, DND maintains comparable kill-death ratios, indicating that its policy does not overfit to easy domination but generalizes well to balanced matchups. These results verify that DND's hierarchical relational modeling benefits both offensive coordination and survivability.

D. Episode Efficiency and Execution Cost

Efficient trajectory management is critical in scenarios where long-term planning and responsiveness are both required. We report the average number of environment steps per episode in Figure 6. A lower step count generally implies faster task resolution and more decisive strategies.

DND consistently completes episodes with fewer steps on all tested maps. Notably, in corridor and 2c_vs_64zg, which involve high-density enemy formations or early contact, DND's policies exploit structure-aware priors to quickly establish advantageous positions, reducing episode length without compromising return.

In more extended engagements like mmm2 or 2s3z_vs_4z, DND maintains its edge, showing that dynamic subgraph construction does not sacrifice strategic patience for short-term gain. This confirms that DND adapts its temporal horizon according to task demands.

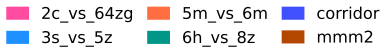


Fig. 2: Color mapping used for all learning-curve figures.

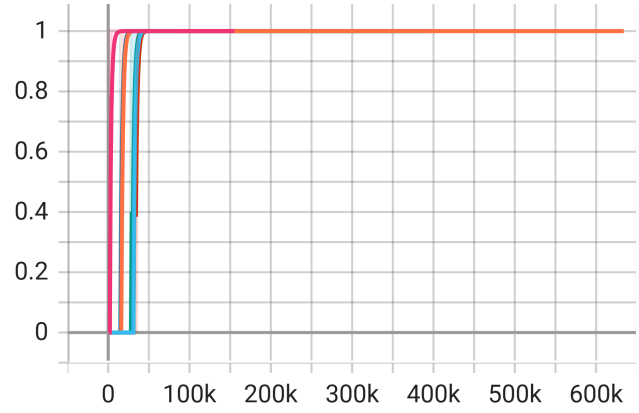


Fig. 3: Maximum evaluation return on six SMAC scenarios (mean ±,std over five seeds).

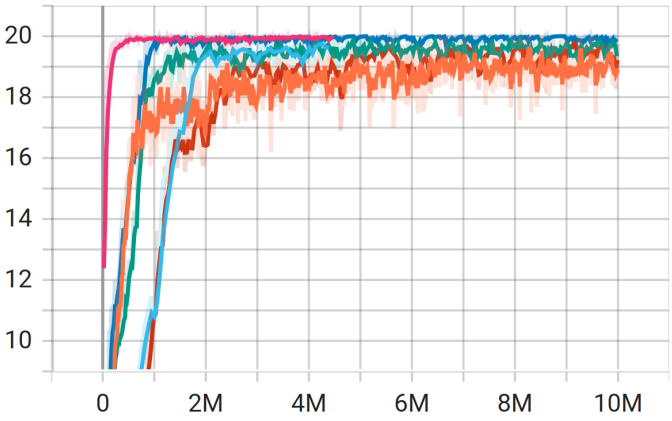


Fig. 4: Learning curves on 2c_vs_64zg, 6h_vs_8z and 2s3z_vs_4z (mean ±,std over five seeds).

E. Cross-Map Generalization

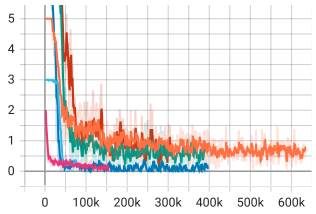
A critical evaluation criterion for multi-agent reinforcement learning algorithms is their ability to generalize across diverse environments without retraining or task-specific tuning. We adopt a unified training protocol on six SMAC scenarios—2c_vs_64zg, 3s_vs_5z, 5m_vs_6m, 6h_vs_8z, corridor, and mmm2—that span a range of coordination challenges.

Figure 7 plots the minimum evaluation return per map across seeds. Our method demonstrates consistently high robustness, with worst-case performances on all maps remaining near-optimal. This highlights that the learned policies are not only performant on average but also stable across varying initializations and episodes.

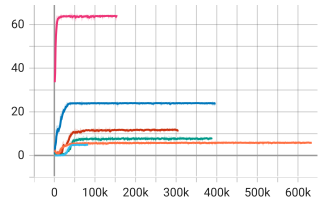
In complex maps such as mmm2 and corridor, which feature heterogeneous agents and delayed engagements, DND sustains performance without specialized modules or tuning. This indicates that the model's representations and action selection strategies transfer well to unseen structural variations, validating its generalization capacity.

F. Optimization Stability

To further verify that the strong empirical results are grounded in a stable learning process, we examine four internal learner metrics during training: reinforcement learning loss, auxiliary loss, gradient norm, and average Q-value predictions.



(a) Dead allies per episode.



(b) Dead enemies per episode.

Fig. 5: Mean number of dead allies and enemies per episode across six SMAC tasks.

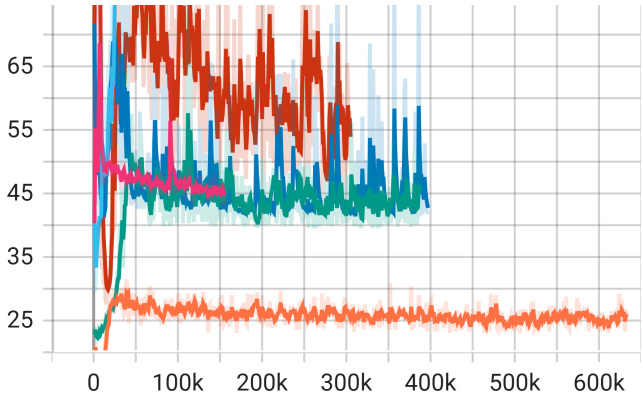


Fig. 6: Average episode length (environment steps) across SMAC maps.

As shown in Figure 8, the main RL loss (rl_loss_avg) and overall loss (loss_avg) both exhibit steady convergence, with no spikes or oscillations. The auxiliary loss (aux_loss_avg) remains consistently low, confirming that regularization or structural penalties are well-balanced and do not disrupt learning.

The gradient norm (grad_norm_avg) is bounded throughout training, and the predicted Q-values (q_avg) gradually rise to a stable plateau, indicating reliable value propagation. Together, these curves validate that our training pipeline avoids issues like divergence, gradient explosion, or underfitting, even under recurrent or graph-based architectures.

These stable optimization trends confirm that the strong performance of DND across environments is not a result of brittle or lucky training runs, but rather reflects a principled and reliable learning process.

V. Conclusion and Future Work

In this paper, we proposed a novel framework for cooperative MARL that dynamically decomposes the joint Q-function via subgraph-based modeling of agent interactions. Our method introduces a principled factorization of the joint action-value function over adaptively constructed neighborhoods, guided by relational clustering and graph convolutional encoders. By incorporating attention-based aggregation across subgraphs, the framework captures both localized coordination and global task dependencies, overcoming the structural limitations of prior value decomposition approaches such as VDN, QMIX, Qatten, and GraphMix.

Through extensive evaluations on challenging SMAC-II scenarios, our method demonstrates superior performance and generalization, particularly in environments with evolving agent dependencies and multi-task cooperation. The results highlight the benefits of embedding structured priors, relational sparsity, and dynamic topologies into multi-agent learning architectures.

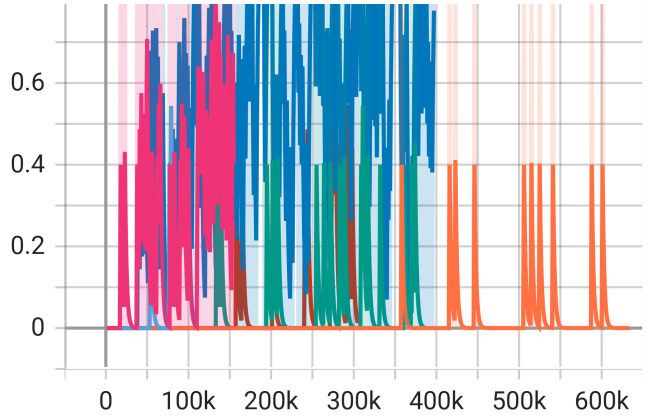


Fig. 7: Minimum evaluation return across six SMAC scenarios, indicating worst-case generalization behavior.

Future Work. *While our model introduces dynamic graph abstraction and improves interpretability, several extensions remain to be explored. First, the integration of communication learning with the discovered subgraph topology may further improve decentralized execution. Second, extending our framework to continuous control tasks and partially observable settings (e.g. Dec-POMDPs with limited field-of-view) can improve its applicability. Lastly, theoretical analysis of the decomposition error and convergence guarantees under dynamic neighborhood partitioning is an important direction for future research.*

References

[1] *P. Hern'andez-Leal, B. Kartal, and M. E. Taylor, "A survey of multi-agent reinforcement learning,"* Autonomous Agents and Multi-Agent Systems, *vol. 33, no. 6, pp. 750–797, 2019.*

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

[3] *M. Samvelyan, T. Rashid, C. Schroeder de Witt, G. Farquhar, N. Nardelli, T. G. Rudner, P. H. Torr, S. Whiteson, and J. Foerster, "The starcraft multi-agent challenge," in* Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems, *2019, pp. 2186–2188.*

[4] *P. Cai, Y. Zheng, H. Li, and Y. Wang, "Traffic light control with multi-agent reinforcement learning," in* NeurIPS Workshop on Machine Learning for Intelligent Transportation Systems, *2019.*

[5] *M. H"uttenrauch, A. Sosic, and G. Neumann, "Deep reinforcement learning for swarm systems," in* Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems, *2019, pp. 1348–1356.*

[6] *F. A. Oliehoek and C. Amato,* A concise introduction to decentralized POMDPs. *Springer, 2016.*

[7] *J. Foerster, G. Farquhar, T. Afouras, N. Nardelli, and S. Whiteson, "Counterfactual multi-agent policy gradients," in* Proceedings of the AAAI Conference on Artificial Intelligence, *vol. 32, no. 1, 2018.*

[8] *R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in* Advances in Neural Information Processing Systems, *vol. 30, 2017.*

[9] *P. Sunehag, G. Lever, A. Gruslys, W. M. Czarnecki, V. Zambaldi, M. Jaderberg, M. Lanctot, and T. Graepel, "Value-decomposition networks for cooperative multi-agent learning," in* Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems, *2018.*

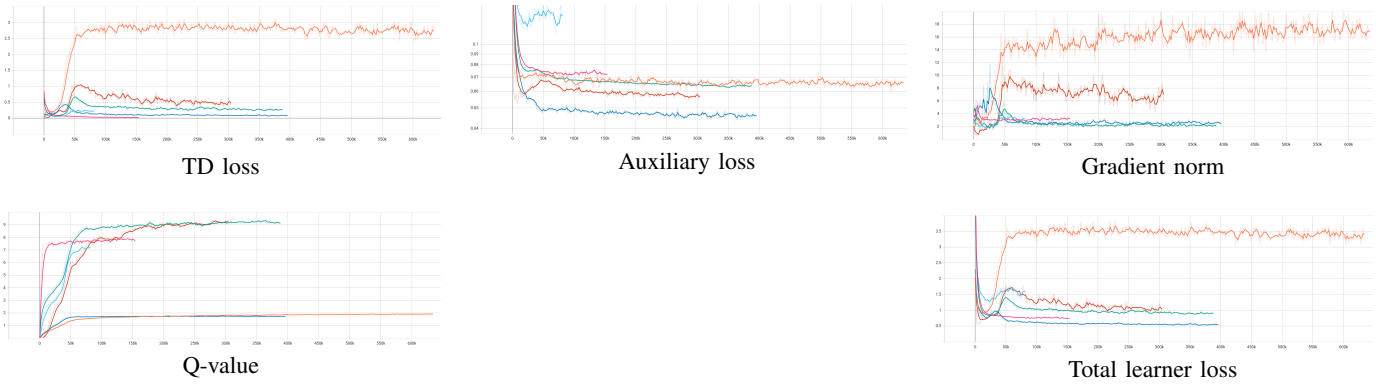


Fig. 8: Internal training diagnostics: TD loss, auxiliary loss, gradient norm, Q-values, and total learner loss.

[10] *T. Rashid, M. Samvelyan, C. Schroeder de Witt, G. Farquhar, J. Foerster, and S. Whiteson, "Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning," in* International Conference on Machine Learning*. PMLR, 2018, pp. 4295–4304.*

[11] *T. Wang, J. Zhang, K. Xu, H. Wang, Y. Yu, and Y. Wang, "Roma: Multi-agent reinforcement learning with emergent roles," in* International Conference on Machine Learning*. PMLR, 2020, pp. 9876–9886.*

[12] *Y. Yang, R. Zhang, M. Xu, and W. Wang, "Qatten: A general framework for cooperative multi-agent reinforcement learning," in* International Conference on Machine Learning*. PMLR, 2020, pp. 10 757–10 767.*

[13] *T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in* International Conference on Learning Representations*, 2017.*

[14] *A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in* Advances in Neural Information Processing Systems*, vol. 30, 2017.*

[15] *W. B"ohmer, A. Anand, M. Zimmer, M. Hessel, A. Dragan, and S. Whiteson, "Deep coordination graphs," in* International Conference on Machine Learning*. PMLR, 2020, pp. 9808–9818.*

[16] *X. Li, H. Wang, J. Zhang, Y. Yu, and Y. Wang, "Structured agent interactions in multi-agent reinforcement learning," in* Advances in Neural Information Processing Systems*, 2021.*

[17] *J. Ma, K. Zhou, Y. Pan, and Y. Yu, "Learning diverse policies for cooperative marl with auto-curricula," in* International Conference on Learning Representations*, 2021.*

[18] *T. Matiisen, A. Oliver, J. S. Cohen, and J. Leike, "Teacher-student curriculum learning,"* IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 9, pp. 3732–3740, 2019.*

[19] *C. Yu, X. Li, Y. Wang, and Y. Yu, "The surprising effectiveness of parameter sharing in deep multi-agent reinforcement learning," in* International Conference on Learning Representations*, 2021.*

[20] *K. L. Son, D. Kim, J. Lee, J. Kim, S. Oh, and Y. Zhang, "Qtran: Learning to factorize with transformation for cooperative multi-agent reinforcement learning," in* International conference on machine learning*. PMLR, 2019, pp. 5887–5896.*

[21] *Y. Du, Y. Yang, Y. Wang, C. Zhang, and H. Dong, "Graph-mix: Dynamic graph mixing for multi-agent coordination," in* International Conference on Learning Representations (ICLR)*, 2023.*

[22] *H. Wang, J. Zhang, and J. Leskovec, "Graphmix: Improved training of gnns for semi-supervised learning," in* International Conference on Machine Learning (ICML)*, 2020.*

[23] *Y. Chen, M. Zhou, H. Yu, H. Liu, and T. Yang, "Structure-aware communication for multi-agent reinforcement learning via graph attention," in* Advances in Neural Information Processing Systems (NeurIPS)*, 2021.*

APPENDIX: TRAINING SETTINGS AND HYPERPARAMETERS

To ensure fair evaluation and reproducibility, we list the detailed training configurations for all SMAC scenarios considered in this work. All models share the same architectural backbone with consistent training settings unless otherwise specified. Table I and Table II summarize the environment parameters and learning hyperparameters, respectively.

TABLE I: SMAC Environment Configurations

Map Name	**#Agents**	**Collectors**	**Evaluators**	**Stop Value**
2c_vs_64zg	2	8	8	1.999
3s_vs_5z	3	4	2	1.999
5m_vs_6m	5	1	1	1.999
6h_vs_8z	6	8	8	1.999
corridor	6	12	12	1.999
MMM2	10	8	8	1.999

TABLE II: Unified Training Hyperparameters

Parameter	**Value**
Batch size	160 / 320 (map-dependent)
Learning rate	3e-4 (cosine schedule)
Optimizer	RMSProp
Discount factor (γ)	0.99
TD steps (n-step)	3
Target update θ	0.008
Auxiliary loss weight	10 (fixed)
Weight decay	1×10^{-5}
Gradient clip value	50
Replay buffer size	100k 300k
Max reuse / staleness	1×10^{9}
Unroll length	1
Evaluation frequency	1000 iterations
Early stop reward	1.999