Understanding the Information Propagation Effects of Communication Topologies in LLM-based Multi-Agent Systems

Xu Shen[1]*, Yixin Liu[2]*, Yiwei Dai[1], Yili Wang[1], Rui Miao[1], Yue Tan[3], Shirui Pan[2], Xin Wang[1]†

[1]School of Artificial Intelligence, Jilin University, Changchun, China,
[2]School of Information and Communication Technology, Griffith University, Australia,
[3]School of Computer Science and Engineering, University of New South Wales, Australia
{shenxu23, daiyw23, ruimiao20}@mails.jlu.edu.cn, {yixin.liu, s.pan}@griffith.edu.au,
{yue.tan}@unsw.edu.au, {wangyili,xinwang}@jlu.edu.cn

Abstract

The communication topology in large language model-based multi-agent systems fundamentally governs inter-agent collaboration patterns, critically shaping both the efficiency and effectiveness of collective decision-making. While recent studies for communication topology automated design tend to construct sparse structures for efficiency, they often overlook why and when sparse and dense topologies help or hinder collaboration. In this paper, we present a causal framework to analyze how agent outputs, whether correct or erroneous, propagate under topologies with varying sparsity. Our empirical studies reveal that moderately sparse topologies, which effectively suppress error propagation while preserving beneficial information diffusion, typically achieve optimal task performance. Guided by this insight, we propose a novel topology design approach, EIB-LEARNER, that balances error suppression and beneficial information propagation by fusing connectivity patterns from both dense and sparse graphs. Extensive experiments show the superior effectiveness, communication cost, and robustness of EIB-LEARNER. The code is in: https://github.com/se7esx/EIB/.

1 Introduction

Agents powered by large language models (LLMs) (Li et al., 2023; Wang et al., 2024a; Xi et al., 2025; Du et al., 2025a,b), have demonstrated strong performance across tasks like reasoning (Yao et al., 2023; Du et al., 2025c), code generation (Zhang et al., 2024b), and complex decision-making (Guo et al., 2024b). A key advancement in this area is the development of collaborative **multi-agent systems (MAS)**, where multiple LLM agents interact to outperform single agent when tackling complex tasks (Talebirad and Nadiri, 2023; Liang et al., 2024; Guo et al., 2024a). While agent role specialization contributes to this improvement (Bo et al., 2024), the communication topology, which regulates how agents *propagate*, *exchange*, and *process* information, plays a more fundamental role in enabling effective multi-agent collaboration (Qian et al., 2025; Zhang et al., 2024a). From simple chain (Zhang et al., 2024c) and tree (Zhou et al., 2024), to fully connected graphs (Hu et al., 2024b) and LLM-generated designs (Zhuge et al., 2024), prior works have proposed various topologies to shape this interaction. These studies converge on a central insight: ***the topology of communication is critical to the effectiveness and efficiency of MAS***.

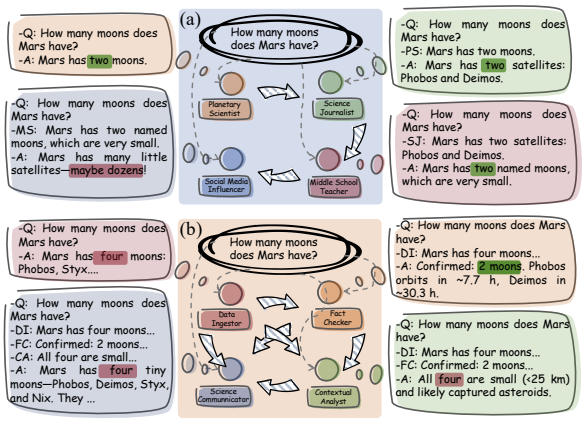


Figure 1: Illustration of (a) insight suppression caused by sparse chain and (b) error propagation induced by dense fully connected topologies.

Given its critical role in MAS performance, recent studies have increasingly focused on the automated design of communication topology using graph learning techniques (Li et al., 2024; Liu et al., 2025a; Zhou et al., 2025), advancing beyond using pre-defined topologies. Central to these approaches is the principle of ***sparsifying*** communication graphs, motivated by the insight that fewer but higher-quality interactions enhance system efficiency. For example, AgentPrune (Zhang

*Equal Contribution
†Corresponding Author

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 12359–12373
November 4-9, 2025

et al., 2025a) and AgentDrop (Wang et al., 2025c) learn a task-specific sparse graph to eliminate suboptimal connections and/or agents, while G-Designer (Zhang et al., 2025b) introduces a graph sparsification learning objective for the communication design model. Despite their advanced performance and efficiency, these methods often treat sparsification as an end goal, without considering a deeper question: *why and when do **sparse** topologies help MAS collaboration?* Conversely, despite enabling maximal inter-agent interaction, *why and when do **dense** topologies underperform*?

To answer the above questions, from a causal perspective, we conduct comprehensive analyses to investigate how the communication topologies with varying sparsity affect the information propagation among agents and thereby impact decision-making performance. Through empirical analysis, we find that sparser topologies exhibit greater robustness against the propagation of error information generated by an individual agent, but may also suppress beneficial insights that contribute to accurate collective decision-making (see Figure 1a). Conversely, denser topologies enable thorough insight dissemination, but also aggressively propagate individual errors (see Figure 1b). With task-oriented analysis, we derive a novel insight: ***an ideal communication topology should have a moderate sparsity, effectively suppressing error propagation while maintaining beneficial insight propagation.***

Based on the insight above, we propose **E**rror-**I**nsight **B**alanced **Learner** (EIB-LEARNER for short) for automated communication topology design of LLM-based MAS. Given a specific query, EIB-LEARNER can dynamically customize an optimal topology by balancing error suppression and insight propagation. Specifically, EIB-LEARNER utilizes dual-view graph neural networks (GNNs) to simultaneously simulate error suppression on sparse graphs and insight propagation on dense graphs, creating complementary topological representations that inform optimal connectivity. Then, EIB-LEARNER integrates the connectivity coefficients learned by both views with query-aware adaptive fusion, blending the error robustness of sparse topologies with the insight propagation capacity of dense topologies. Extensive experiments on 6 benchmark datasets demonstrate the effectiveness, communication efficiency, and robustness of EIB-LEARNER. To sum up, the main contributions of this paper are as follows:

- **Causal Analysis.** We conduct causal counterfactual analyses to assess the impact of an agent on collective decision making in communication topologies with varying sparsity, revealing how error and insight propagation affect the inter-agent collaboration.
- **Novel Method.** We propose EIB-LEARNER, a GNN-based communication topology design approach that simulates error and insight propagation in MAS and learn reliable and efficient topologies by balancing error-insight trade-off.
- **Comprehensive Experiments.** Experiments on 6 MAS decision-making benchmarks demonstrate that EIB-LEARNER achieves superior accuracy with reduced communication cost and better robustness compared to existing baselines.

2 Problem Definition

Communication Topology We formulate the communication topology of a multi-agent system (MAS) as a directed acyclic graph (DAG), defined as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. Here $\mathcal{V} = \{v_1, \cdots, v_N\}$ denotes the set of nodes, where each node v_i indicates a LLM-based agent. The edge set $\mathcal{E}$ specifies directed communication links, where each edge $e_{ij} = (v_i, v_j) \in \mathcal{E}$ indicates that agent v_i can receive messages from agent v_j. The connectivity of $\mathcal{G}$ can be represented by an adjacency matrix $\mathbf{A} \in \{0,1\}^{N \times N}$, where $\mathbf{A}_{i,j} = 1$ if and only if $(v_i, v_j) \in \mathcal{E}$ and $\mathbf{A}_{i,j} = 0$ otherwise.

In this setup, each node $v_i \in V$ corresponds to a LLM-based agent and can be formalized as $v_i = \{\texttt{Role}_i, \texttt{State}_i\}$, where $\texttt{Role}_i$ denotes the pre-defined role of agent v_i for a specific task, and $\texttt{State}_i$ stores the history response of the agent v_i and its interaction history with other agents. Each LLM-based agent receives a prompt $\mathcal{P}$ and generates the response $\mathcal{R}_i$ by:

$$\mathcal{R}_i = v_i(\mathcal{P}) = v_i(\mathcal{P}_{\text{sys}}, \mathcal{P}_{\text{usr}}), \tag{1}$$

where $\mathcal{P}_{\text{sys}}$ is the system prompt that combine the $\texttt{Role}_i$ and $\texttt{State}_i$ of the current agent, and $\mathcal{P}_{\text{usr}}$ denotes the user prompt, including current query, task instructions, and other necessary information.

Communication Protocol Given a user query $\mathcal{Q}$, a MAS performs K rounds of interaction based on the communication graph $\mathcal{G}$. At each round t, the execution order of agents is determined by a topological sort $\sigma = [v_{\sigma_1}, \ldots, v_{\sigma_N}]$, ensuring that

an agent is only activated after all its in-neighbors have produced their outputs. Each agent v_i generates its own response $\mathcal{R}_i^{(t)}$ according to the user query and the outputs of its neighbors:

$$\mathcal{R}_i^{(t)} = v_i(\mathcal{P}_{\text{sys}}, \mathcal{P}_{\text{usr}}^{(t)}), \quad (2)$$
$$\mathcal{P}_{\text{usr}}^{(t)} = \{\mathcal{Q}\} \cup \{\mathcal{R}_j^{(t)} | v_j \in \mathcal{N}(v_i)\},$$

where $\mathcal{N}(v_i) = \{v_j | (v_j, v_i) \in \mathcal{E}\}$ denotes the neighbor set of the agent v_i. After K rounds, the final output $\mathcal{O}$ is obtained by aggregating all agent outputs:

$$\mathcal{O} = \text{Aggregate}(\mathcal{R}_i^{(K)} \dots \mathcal{R}_n^{(K)}), \quad (3)$$

where the aggregation strategy $\text{Aggregate}(\cdot)$ varies across implementations, including majority voting among agents, delegating the final decision to a specific agent (the mechanism used by all MAS in this paper), or selecting the output from the last agent in the execution order. The communication rounds K can be either predefined or adaptively determined via early-stopping mechanisms.

MAS Communication Topology Design Problem In this paper, we aim to design a communication topology for a LLM-based MAS that maximizes the utility of the collective output under a given task query. Formally, Our objective is to find an optimal topology $\mathcal{G}$ from a feasible topology space $\mathbb{G}$ such that:

$$\mathcal{G} = \arg\max_{\mathcal{G} \in \mathbb{G}} \phi(\mathcal{G}(\mathcal{Q})), \quad (4)$$

where $\phi(\cdot)$ denotes a utility function that evaluates the correctness or overall quality of the system output $\mathcal{G}(\mathcal{Q})$ in response to the input query $\mathcal{Q}$. The search space $\mathbb{G}$ includes all valid DAGs over the agent set $\mathcal{V}$ that satisfy acyclicity and predefined execution constraints.

3 Analysis of Communication Topology

In this section, we conduct extensive empirical analysis to investigate how communication topologies of varying sparsity shape the actual information propagation among agents and influence collective decision-making outcomes in a multi-agent system (MAS). *First*, we discuss two causal effects of communication topologies with different sparsity: ***error propagation*** and ***insight propagation***. Specifically, we analyze how the erroneous information generated by an individual agent spreads, as well as how beneficial insights diffuse across the topology. *Next*, we evaluate how sparsity influences the effectiveness through task-oriented performance analysis, and expose the ideal information propagation characteristics of communication topologies. *Finally*, we assess the information diffusion capabilities of existing communication paradigms, including both pre-defined and automatically learned topologies, with respect to their error and insight propagation properties.

3.1 Causal Metric of Agent Impact

In order to analyze the information propagation effects of different communication topologies, we first define a new causal metric, denoted as Counterfactual Agent Propagation Effect (**CAPE**), that quantifies the causal influence of the output of a single agent on the final answer of MAS under a given communication graph. Specifically, we perform a counterfactual intervention by forcing agent v_i to produce a deliberately manipulated output, and then measure whether this change alters the final prediction made by the MAS under a fixed topology $\mathcal{G}$. The formal definition is as follows:

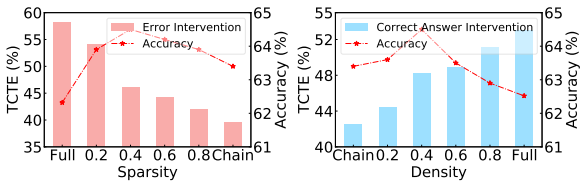
Definition 3.1 *Given a MAS under communication graph $\mathcal{G}$, for a query $\mathcal{Q}$, let $y_{\mathcal{Q}}^{orig} \in \{0, 1\}$ denote whether the final system answer is correct under the original communication process. The Counterfactual Agent Propagation Effect (CAPE) of agent v_i under $\mathcal{G}$ and query $\mathcal{Q}$ as:*

$$CAPE_i(\mathcal{G}, \mathcal{Q}) = \mathbb{I}[y_q^{cf}(\mathcal{G}, do(\mathcal{O}_i := \hat{\mathcal{O}}_i)) \neq y_q^{orig}(\mathcal{G})], \quad (5)$$

where $\text{do}(\mathcal{O}_i := \hat{\mathcal{O}}_i))$ denotes a targeted intervention that forces only agent v_i output to a counterfactual value $\hat{\mathcal{O}}_i$, $y_{\mathcal{Q}}^{cf} \in \{0, 1\}$ is the correctness of the final system prediction after the intervention, and $\mathbb{I}[\cdot]$ is the indicator function for whether the final answer of MAS is flipped.

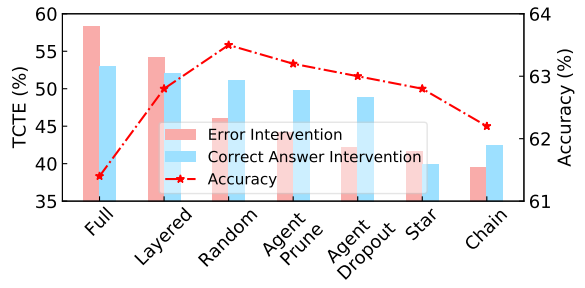
A higher CAPE value suggests that changes in the output of an individual agent, once propagated through the communication structure, have a stronger influence on the final decision of MAS. Building on the ability of CAPE to quantify agent-level influence, we further define the Total Counterfactual Topology Effect (**TCTE**) to expend the casual metric to the topology level, providing a comprehensive measure of the general sensitivity of the topology to localized interventions.

Definition 3.2 *Given a communication topology $\mathcal{G}$, its Total Counterfactual Topology Effect (TCTE)*



(a) Error propagation

(b) Insight propagation



(c) Analysis of pre-defined and learned topologies

Figure 2: Empirical results of error propagation effect and insight propagation effects.

given a query $\mathcal{Q}$ is defined as:

$$TCTE(\mathcal{G}, \mathcal{Q}) = \frac{1}{N} \sum_{v_i \in \mathcal{V}} \frac{1}{\sqrt{d_i}} \cdot CAPE_i(\mathcal{G}, \mathcal{Q}), \quad (6)$$

where d_i is the degree of agent v_i in graph $\mathcal{G}$.

In TCTE, the coefficient $1/\sqrt{d_i}$ downweights the agents with high degree, reflecting the intuition that highly connected agents are more likely to spread influence, and we wish to normalize their impact accordingly. A higher TCTE indicates that the topology is more responsive to specific changes in agent-level output, implying a greater potential to either propagate erroneous information or effectively transmit correct insights.

3.2 Empirical Analysis

In this subsection, we aim to empirically assess how different communication topologies affect the sensitivity of MAS to agent-level perturbations. We quantify the sensitivity by computing TCTE of communication graphs, and also evaluate the effectiveness of different topologies using task-oriented measurement, i.e., question-answering accuracy.

Experimental Settings We conduct our experiments on the MMLU dataset (Hendrycks et al., 2021), a benchmark for evaluating knowledge and reasoning across diverse domains. In our MAS setup, each of the six agents is an independent instance of GPT-3.5 with a distinct role and prompt template, and all communicate according to a pre-specified topology. For each topology, we construct two evaluation sets: one containing 500 questions that the MAS originally answers correctly, and another with 500 questions that the system answers incorrectly. These sets serve as the basis for simulating both detrimental and beneficial agent-level interventions under varying graph structures.

3.2.1 Analysis of Error Propagation Effect

Setup To investigate ***how different communication topologies spread the erroneous information generated by a single agent***, we focus on the 500 originally correct questions and assess the vulnerability of MAS to local errors. Specifically, we start from a fully connected communication graph (`Full`) and progressively remove edges randomly to generate topologies with different sparsity levels, until it degrades into a chain graph (`Chain`). For each graph, we apply a counterfactual intervention by modifying the system prompt of an agent to produce an incorrect output, i.e., $\hat{\mathcal{O}}_i = \mathcal{O}_{\text{error}}$, and compute the corresponding TCTE to quantify how often the final prediction flips from correct to incorrect under each topology.

Discussion of Results As shown in Figure 2a, dense topologies exhibit greater susceptibility to error propagation, wherein erroneous outputs from individual agents are more likely to compromise the final prediction. When the sparsity is zero (i.e., `Full`), TCTE reaches its peak. In contrast, sparser structures can effectively mitigate this issue. For example, in the extreme case `Chain`, TCTE is reduced by up to 19%, indicating improved robustness to localized errors.

Finding 1: Denser communication topologies are more susceptible to error spreading, while sparser topologies demonstrate stronger resistance to the error propagation effect.

3.2.2 Analysis of Insight Propagation Effect

Setup In this experiment, we examine ***how different topologies support the propagation of beneficial insight generated by each agent***. We focus on the 500 originally incorrect questions and start from a sparse `Chain` topology, gradually adding edges to increase connectivity until reaching `Full`. In each case, we perform a counterfactual intervention by injecting the correct answer into a selected agent's prompt, i.e., $\hat{\mathcal{O}}_i = \mathcal{O}_{\text{correct}}$, and compute the corresponding TCTE to measure how often the final prediction flips from incorrect to correct.

Discussion of Results As shown in Figure 2b, sparse topologies tend to impede the propagation of accurate and informative signals, preventing them from influencing the final output. When connectivity is minimal (i.e., `Chain`), TCTE reaches its lowest value, indicating poor transmission of correct insights. With increasing density, the insight propagation effect is strengthened. In `Full` structure, TCTE increases by 10.5%, suggesting that the system becomes more capable of integrating accurate agent-level inputs into the final decision.

Finding 2: Increased connectivity enhances beneficial insight propagation in MAS, while sparse topologies constrain the integration of informative signals from individual agents.

3.2.3 Effectiveness Analysis

Setup While communication topologies of different sparsity levels produce distinct error propagation and insight propagation effects, a follow-up question raises: ***How do the topological sparsity ultimately affect real-world task performance?*** To answer this question, we examine the task-oriented performance of different topologies, in term of accuracy for MMLU dataset.

Discussion of Results The task-oriented accuracies are demonstrated by the red dashed line in Figures 2a and 2b. From both cases, we can observe that a moderate sparsity contributes to better task performance, while overly sparse or dense structures lead to suboptimal outcomes. Recalling **Finding 1** and **Finding 2**, this phenomenon arises because topology with intermediate sparsity strikes a balance: it sufficiently suppresses erroneous information while effectively propagating beneficial insights, thereby maximizing the decision-making accuracy of MAS. At the agent level, an ideal topology should balance this trade-off through moderate sparsity: establishing dense connections around high-accuracy agents to amplify beneficial insights, while maintaining sparse connectivity for error-prone agents to limit misinformation propagation.

Insight: Topologies with moderate sparsity optimize MAS performance by balancing error suppression and insight propagation, achieved through dense connections around accurate agents and sparse links for error-prone agents.

3.2.4 Analysis of Existing Topologies

Setup With the **Insight** given above, we are curious about: ***How do existing communication topologies, both pre-defined and learnable, perform in balancing error propagation and insight diffusion?*** We evaluate a range of representative topologies, including pre-defined ones such as `Full`, `Layered`, `Random`, `Star`, and `Chain`, as well as topologies learned through state-of-the-art methods such as `AgentDropout` (Wang et al., 2025c) and `AgentPrune` (Zhang et al., 2025a).

Discussion of Results As shown in Figure 2c, the results reveal a consistent trend: sparse topologies are more robust to error propagation, while denser structures better support the amplification of helpful insight, which echoes our earlier analysis. However, we found that the topologies generated by `AgentDrop` and `AgentPrune` do not show superior performance compared to random graphs with moderate sparsity, indicating that topology optimization methods that explicitly balance error and insight propagation.

Limitation: Current methods show limited effectiveness in simultaneously managing error suppression and insight propagation.

4 Methodology

To address the above **Limitation**, in this section, we propose a novel **E**rror-**I**nsight **B**alanced **Learner** (EIB-LEARNER for short) for communication topology optimization in multi-agent system (MAS). Motivated by our **Insight**, EIB-LEARNER balances the error-insight trade-off by co-training two complementary graph neural network (GNN) simulators to simulate the error suppression and insight propagation given a specific query (Section 4.1), and then adaptively blending their learned inter-agent coefficients to construct robust topologies (Section 4.2). The overall pipeline of EIB-LEARNER is shown in Figure 3.

4.1 GNN-based Propagation Simulators

To balance error suppression and insight propagation in communication topologies, a critical prerequisite is identifying beneficial agents (those likely to contribute accurate insights) and error-prone agents (those susceptible to generating misinformation) within the MAS for a given query $\mathcal{Q}$, and then model the information flow among these agents.

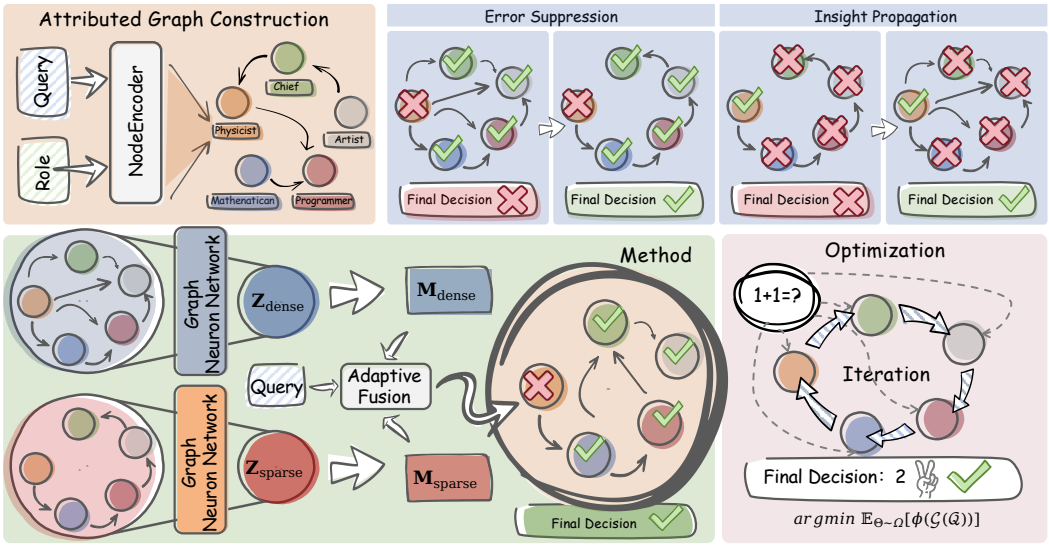


Figure 3: The overall framework of EIB-LEARNER. EIB-LEARNER simulates MAS communication via GNNs, generating two connectivity coefficient matrices to suppress error spreading (sparse view) and enhance insight propagation (dense view), which are then combined by a query-aware fusion module into an optimal topology.

Nevertheless, without prior knowledge of agent reliability in answering $\mathcal{Q}$, it is challenging to distinguish the agents and simulate their responses. To address this challenge, we utilize GNNs to simulate the responding process of MAS given a query, and hence model error and insight propagation.

MAS as an Attributed Graph While the inter-agent communication can be represented as a graph structure, we further embed the agent role information and query text into the feature input of GNNs. Specifically, the feature of each agent node v_i can be acquired by:

$$\mathbf{x}_i \leftarrow \text{NodeEncoder}(\mathcal{T}(\texttt{Role}_i), \mathcal{Q}), \quad (7)$$

where $\mathcal{T}(\cdot)$ extracts the textual description of the agent role and its corresponding prompt, and NodeEncoder$(\cdot)$ can be implemented using lightweight pre-trained text embedding models. The integration of agent role and query allows the GNN model to capture the contribution (beneficial or error-prone) of each agent given a specific query.

GNNs for Propagation Simulation With the MAS attributed graph as input, we leverage GNNs to simulate the information propagation in MAS. Formally, at the l-th GNN layer, the message representation for node v_i is updated by aggregating features from its neighbors:

$$\mathbf{m}_i^{(l)} = \text{MP}\left(\mathbf{m}_i^{(l-1)}, \{\mathbf{m}_j^{(L-1)} \mid v_j \in \mathcal{N}(v_i)\}\right), \quad (8)$$

where $\mathbf{m}_i^{(0)} = \mathbf{X}_i$ is the raw input feature of node v_i, $\mathcal{N}(v_i)$ denotes the set of neighbors, and MP$(\cdot)$ is the message passing function that aggregates and transforms neighboring information. The message passing scheme of GNNs can effectively mimic the information flow in MAS: the message aggregation of neighbors reflects the communication between agents, while the feature transformation can simulate the agent responding given a query.

In EIB-LEARNER, we design a dual-view framework, with two GNNs to simulate the error suppression and insight propagation, respectively. Inspired by **Finding 1**, we employ the most sparse `chain` topology to simulate error suppression in MAS; In the other hand, **Finding 2** motivates us to use the densest `full` topology to simulate insight propagation effect. These two different topologies determine different neighbor definitions in Eq. (8).

4.2 Error-Insight Balanced Topology Learner

Inter-Agent Coefficient Modeling Building upon the GNN simulators, we use an inner-product decoder to estimate pairwise connectivity to model the inter-agent connectivity in the optimal communication topology, which can be written as:

$$\mathbf{Z}_{\text{view}} = \text{GNN}(\mathbf{A}_{\text{view}}, \mathbf{X}), \mathbf{M}_{\text{view}} = \sigma(\mathbf{Z}_{\text{view}}^T \mathbf{Z}_{\text{view}}), \quad (9)$$

where subscript "view" can be "sparse" or "dense", corresponding to the views that simulate error suppression and insight propagation, respectively, $\mathbf{M}_{\text{view}}$ is the coefficient matrix to model inter-agent connectivity, and $\sigma(\cdot)$ denotes the sigmoid function to ensure outputs are in $[0, 1]$. The inner product

operation leverages propagated information to estimate agent compatibility, determining the optimal likelihood of connection between any two agents in the communication topology.

Adaptive Dual-View Fusion Based on our **Insight**, the optimal communication topology should balance the factors of error suppression and insight propagation, which are modeled by the sparse and dense views, respectively. To this end, we introduce a simple yet effective fusion function to fuse $\mathbf{M}_{\text{sparse}}$ and $\mathbf{M}_{\text{dense}}$ into a comprehensive coefficient matrix $\mathbf{M}_{\text{final}}$.

Concretely, we implement the fusion with a query-aware gating mechanism. Firstly, a lightweight feedforward network (i.e. MLP) takes the task query embedding $\mathbf{Q}$ (acquired by pretrained text encoder) as input and outputs the fusion weight $\alpha = [\alpha_{\text{dense}}, \alpha_{\text{sparse}}]$ via a softmax function, dynamically adjusting the contribution of each view. This allows the final topology to adapt to the semantic demands of different tasks, favoring dense propagation when the context is reliable, and leaning toward sparse filtering under uncertainty. Then, we can acquire $\mathbf{M}_{\text{final}}$ by:

$$\mathbf{M}_{\text{final}} = \alpha_{\text{dense}} \cdot \mathbf{M}_{\text{dense}} + \alpha_{\text{sparse}} \cdot \mathbf{M}_{\text{sparse}}. \quad (10)$$

The coefficient $\mathbf{M}_{\text{final}}$ is used to sample a discrete communication topology $\mathcal{G}$ that determines how messages flow between agents during inference, following the Bernoulli edge sampling strategy proposed in Zhang et al. (2025a).

Model Optimization To optimize this structure generation process, we aim to maximize the utility of the resulting communication topology, which measures the correctness of the final output $\mathcal{O} = \mathcal{G}(\mathcal{Q})$ under the learned communication topology. However, the utility function $\phi(\cdot)$ is typically non-differentiable under our task setting since it depends on downstream black-box reasoning (e.g., querying LLMs). Hence, we adopt a standard policy gradient method to estimate gradients and update model parameters:

$$\nabla_{\Theta} \mathbf{E}_{\Theta\sim\Omega}\Big[\phi(\mathcal{G}(\mathcal{Q}))\Big] \approx \tfrac{1}{M}\textstyle\sum_{m=1}^{M} \phi(\mathcal{O}_m)\nabla_{\Theta}\log(P(\mathcal{G}_m)), \quad (11)$$

where the $\Theta \sim \Omega$ are learnable parameters in EIB-LEARNER, $P(\mathcal{G}_m)$ calculates the probability of $\mathcal{G}_m$ being sampled. $\phi(\mathcal{O}_m)$ can be treated as the task-specific reward of the final answer produced under current topology. The pseudo-code algorithm of EIB-LEARNER is provided in Appendix A.

5 Experiments

5.1 Experimental Setup

Datasets We evaluate EIB-LEARNER on six benchmarks from three domains: general reasoning: MMLU (Hendrycks et al., 2021), mathematical reasoning: GSM8K (Cobbe et al., 2021), MultiArith (Roy and Roth, 2015), SVAMP (Patel et al., 2021), and AQuA (Ling et al., 2017), and code generation: HumanEval (Chen et al., 2021), following the evaluation setup adopted in G-designer (Zhang et al., 2025b). Details are in the Appendix C.

Baselines We consider a broad range of baselines covering both single-agent prompting strategies and multi-agent communication: 1) single-agent methods such as CoT (Wei et al., 2022), Self-Consistency (Wang et al., 2023); 2) multi-agent systems with fixed topologies including Chain, Tree, Complete Graph, and Random Graph (Qian et al., 2025); 3) LLM-Debate (Du et al., 2023); and 4) automated design approaches including AgentPrune (Zhang et al., 2025a), AgentDrop (Wang et al., 2025c), and G-Designer (Zhang et al., 2025b).

Implementation Details We use GPT-4o throughout all experiments, accessed via the OpenAI API. A summarizer agent is designated to aggregate the dialogue history and produce the final answer, where the round $K = 3$. This also corresponds to the number of GNN layers used. For agent representation, we implement the $\text{NodeEncoder}(\cdot)$ using the pretrained `all-MiniLM-L6-v2`, with an embedding dimension $D = 384$. We use 6 agents for the complex reasoning benchmark MMLU, 5 agents for HumanEval, and 4 agents for all mathematical reasoning benchmarks. Each experiment is optimized using $B \in \{40, 60\}$ query samples.

5.2 Performance Comparison

EIB-LEARNER achieves state-of-the-art performance on all six benchmarks. As shown in Table 1, EIB-LEARNER obtains the highest average accuracy of **91.38%**, outperforming all single-agent and multi-agent baselines which adopt the predefined or LLM-generated topology. Specifically, predefined topologies achieve an average performance of at most **85.53%** (e.g., LLM-Debate), while automated design methods perform better, with G-designer reaching up to **90.04%**. Even against recent best topology optimization methods, EIB-LEARNER still achieves an average performance improvement of **1.34%**. Experimental results ver-

Table 1: Performance comparison (%) on six benchmarks. The best results are highlighted in **bold**.

Method	MMLU	GSM8K	AQuA	MultiArith	SVAMP	HumanEval	Avg.
Vanilla	80.39	82.30	71.06	93.09	86.55	71.39	80.80
CoT	81.69 ↑1.30	86.50 ↑4.20	73.58 ↑2.52	93.25 ↑0.16	87.36 ↑0.81	74.67 ↑3.28	82.84 ↑2.04
SC (CoT)	83.66 ↑3.27	81.60 ↓0.70	75.63 ↑4.57	94.12 ↑1.03	88.59 ↑2.04	79.83 ↑8.44	83.91 ↑3.11
Chain	83.01 ↑2.62	88.30 ↑6.00	74.05 ↑2.99	93.27 ↑0.18	87.17 ↑0.62	81.37 ↑9.98	84.53 ↑3.73
Tree	81.04 ↑0.65	85.20 ↑2.90	71.23 ↑0.17	93.68 ↑0.59	88.91 ↑2.36	80.53 ↑9.14	83.43 ↑2.63
Complete	82.35 ↑1.96	80.10 ↓2.20	72.95 ↑1.89	94.53 ↑1.44	84.01 ↓2.54	79.03 ↑7.64	82.16 ↑1.36
Random	84.31 ↑3.92	86.90 ↑4.60	76.48 ↑5.42	94.08 ↑0.99	87.54 ↑0.99	82.66 ↑11.27	85.33 ↑4.53
LLM-Debate	84.96 ↑4.57	91.40 ↑9.10	77.65 ↑6.59	96.36 ↑3.27	90.11 ↑3.56	84.70 ↑13.31	87.53 ↑6.73
AgentPrune	85.07 ↑4.57	91.10 ↑8.80	80.51 ↑9.45	94.65 ↑1.56	90.58 ↑4.03	86.75 ↑15.36	88.09 ↑7.29
AgentDropout	85.62 ↑5.23	91.70 ↑9.40	80.94 ↑9.88	95.60 ↑2.51	91.04 ↑4.49	85.98 ↑14.59	88.48 ↑7.68
G-designer	86.92 ↑6.53	93.80 ↑11.50	81.60 ↑10.54	96.50 ↑3.41	93.10 ↑6.55	88.33 ↑16.94	90.04 ↑9.24
EIB-LEARNER	**88.90** ↑8.51	**95.20** ↑12.90	**83.49** ↑12.43	**96.83** ↑3.74	**94.70** ↑8.15	**89.15** ↑17.76	**91.38** ↑10.58

Table 2: Results of ablation study.

Method	MMLU	GSM8K	HumanEval	Average
Vanilla	80.39	82.30	71.39	78.02
EIB-LEARNER	**88.90**	**95.20**	**89.15**	**91.08**
w/o Dense	84.85	91.10	86.57	87.51
w/o Sparse	86.17	93.50	88.26	89.31
w/o Fusion	85.23	92.80	87.07	88.37

ify that EIB-LEARNER generates a better topology and achieves accurate and reliable decisions by balancing insight and error propagation. We provide the case study in the Appendix D.

5.3 Ablation Study

All modules in EIB-LEARNER are critical to maintaining strong reasoning performance. We conduct ablation studies on MMLU, GSM8K, and HumanEval to evaluate the impact of different communication components: (1) **w/o Dense**, which removes the dense GNN branch; (2) **w/o Sparse**, which excludes the sparse GNN branch; and (3) **w/o Fusion**, which replaces the fusion module with a direct addition operation. As shown in Table 2, removing the dense GNN branch leads to the largest performance drop. For example, on GSM8K, accuracy decreases by **4.1%**. The absence of either the sparse GNN branch or the fusion component also results in comparable degradation. These results suggest that the three modules interact synergistically, and removing any of them weakens the overall reasoning capacity of MAS.

5.4 Token Efficiency

EIB-LEARNER achieves lower token cost with higher performance. As shown in Figure 4a and 4b, although our approach does not explicitly pursue sparsity, it achieves comparable cost to sparsity-based G-designer while delivering significantly higher accuracy. This observation holds across both MMLU and GSM8K, where EIB-LEARNER maintains efficient communication without compromising reasoning quality. Compared to predefined communication topologies such as fully connected graphs and LLM-Debate, EIB-LEARNER substantially reduces token consumption which highlighting its ability to eliminate redundant interactions while preserving critical information flow.

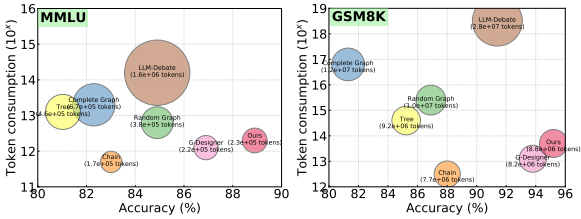
5.5 Robustness Analysis

EIB-LEARNER demonstrates strong robustness against adversarial prompt attacks. Following Zhuge et al. (2024), we simulate a system prompt attack by injecting adversarial prompts into a single agent. As shown in Figure 4c, simple topologies such as chain and tree graphs suffer substantial degradation under such attacks, with performance drops up to **11.8%**. In contrast, EIB-LEARNER exhibits strong robustness: the accuracy only drops by the least decrease of **1.24%**. This robustness arises from our balanced topology design, which mitigates error propagation through bottleneck structures while promoting rapid consensus via densely connected reliable agents.

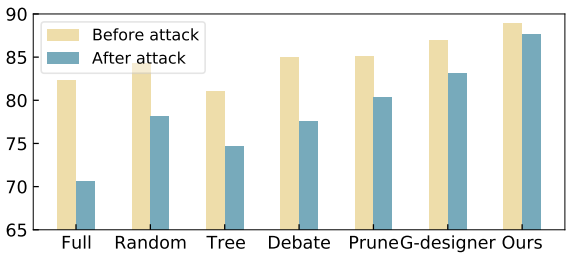
6 Related Work

6.1 LLM-based Multi-Agent Systems

Large Language Models (LLMs) have demonstrated remarkable capabilities across a wide range of tasks, from reasoning (Wang et al., 2024b, 2023, 2025b) and planning to dialogue (Hu et al., 2024a; Yi et al., 2024) and programming (Ishibashi and Nishimura, 2024; Zhang et al., 2024b). Extending from single-agent settings, recent work has explored organizing multiple LLM-based agents into collaborative systems, unlocking new poten-



(a) Token cost of MMLU (b) Token cost of GSM8K



(c) Robustness under attack

Figure 4: Experiments on token consumption and robustness against prompt injection attacks.

tial through inter-agent interaction (Guo et al., 2024a). Representative systems adopt diverse coordination strategies, including sequential reasoning pipelines (Wei et al., 2022), debate-based role playing (Li et al., 2024), and centralized planning (Zhuge et al., 2024). These advances highlight the importance of inter-agent communication in scaling LLM capabilities to more complex, open-ended tasks.

6.2 MAS as Graphs

The effectiveness of LLM-based MAS has been closely tied to the quality of their communication topologies (Zhuge et al., 2024; Miao et al., 2025; Liu et al., 2025b; Li et al., 2025; Wang et al., 2025a). Graphs provide a natural and expressive abstraction for modeling inter-agent interactions, enabling both structured information flow and flexible coordination (Hu et al., 2024b; Zhou et al., 2025). Early approaches adopt fixed topologies such as chains, stars, or fully connected graphs (Qian et al., 2025), each offering different trade-offs in information sharing and control (Li et al., 2024). More recent research explores learnable communication graphs to enhance execution efficiency and scalability. For example, AgentPrune (Zhang et al., 2025a) introduces task-specific sparse masks to suppress redundant communication; AgentDrop (Wang et al., 2025c) applies stochastic dropout to nodes and edges to improve robustness; and G-Designer (Zhang et al., 2025b) dynamically generates query-dependent topologies for better task adaptation. While these methods improve computational efficiency and economic viability by learning compact and adaptive communication structures, they generally treat sparsity as a cost-saving mechanism without explicitly analyzing when or why sparse or dense topologies lead to better multi-agent outcomes.

7 Conclusion

In this paper, we present EIB-LEARNER, a topology optimization framework for LLM-based multi-agent systems, grounded in causal analysis of information propagation. By identifying that moderately sparse structures best balance error suppression and insight preservation, EIB-LEARNER adaptively fuses dense and sparse views to form task-specific topologies. Experiments on six datasets across reasoning, math, and code tasks show that EIB-LEARNER consistently improves performance, enhances robustness, and reduces communication overhead.

Limitation

While our method demonstrates strong performance on reasoning, math, and code generation tasks, the scope of evaluation remains limited. To better assess the generalizability of our framework, future work should explore more diverse task domains, such as real-world decision-making and open-domain dialogue. In addition, the current design relies on a fixed set of predefined agent roles and manually crafted prompts, which may limit adaptability in unfamiliar or evolving scenarios.

Ethics Statement

Our research focuses exclusively on scientific questions, with no involvement of human subjects, animals, or environmentally sensitive materials. Therefore, we foresee no ethical risks or conflicts of interest. We are committed to maintaining the highest standards of scientific integrity and ethics to ensure the validity and reliability of our findings.

Acknowledgements

This work was supported by a grant from the National Natural Science Foundation of China under grants (No.62372211, 62272191), and the Science and Technology Development Program of Jilin Province (No.20250102216JC).

References

Xiaohe Bo, Zeyu Zhang, Quanyu Dai, Xueyang Feng, Lei Wang, Rui Li, Xu Chen, and Ji-Rong Wen. 2024. Reflective multi-agent collaboration based on large language models. *Advances in Neural Information Processing Systems*, 37:138595–138631.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, and 1 others. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, and 1 others. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Enjun Du, Xunkai Li, Tian Jin, Zhihan Zhang, Rong-Hua Li, and Guoren Wang. 2025a. Graphmaster: Automated graph synthesis via llm agents in data-limited environments. ArXiv preprint arXiv:2504.00711v2.

Enjun Du, Siyi Liu, and Yongqi Zhang. 2025b. Graphoracle: A foundation model for knowledge graph reasoning. ArXiv preprint arXiv:2505.11125.

Enjun Du, Siyi Liu, and Yongqi Zhang. 2025c. Mixture of length and pruning experts for knowledge graphs reasoning. ArXiv preprint arXiv:2507.20498.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B Tenenbaum, and Igor Mordatch. 2023. Improving factuality and reasoning in language models through multiagent debate. In *Forty-first International Conference on Machine Learning*.

Taicheng Guo, Xiuying Chen, Yaqi Wang, Ruidi Chang, Shichao Pei, Nitesh V Chawla, Olaf Wiest, and Xiangliang Zhang. 2024a. Large language model based multi-agents: a survey of progress and challenges. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, pages 8048–8057.

Xudong Guo, Kaixuan Huang, Jiale Liu, Wenhui Fan, Natalia Vélez, Qingyun Wu, Huazheng Wang, Thomas L Griffiths, and Mengdi Wang. 2024b. Embodied llm agents learn to cooperate in organized teams. In *Language Gamification-NeurIPS 2024 Workshop*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. In *International Conference on Learning Representations*.

Mengkang Hu, Pu Zhao, Can Xu, Qingfeng Sun, Jianguang Lou, Qingwei Lin, Ping Luo, and Saravan Rajmohan. 2024a. Agentgen: Enhancing planning abilities for large language model based agent via environment and task generation. *arXiv preprint arXiv:2408.00764*.

Shengchao Hu, Li Shen, Ya Zhang, and Dacheng Tao. 2024b. Learning multi-agent communication from graph modeling perspective. In *International Conference on Learning Representations*.

Yoichi Ishibashi and Yoshimasa Nishimura. 2024. Self-organized agents: A llm multi-agent framework toward ultra large-scale code generation and optimization. *arXiv preprint arXiv:2404.02183*.

Guohao Li, Hasan Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. 2023. Camel: Communicative agents for" mind" exploration of large language model society. *Advances in Neural Information Processing Systems*, 36:51991–52008.

Shiyuan Li, Yixin Liu, Qingsong Wen, Chengqi Zhang, and Shirui Pan. 2025. Assemble your crew: Automatic multi-agent communication topology design via autoregressive graph generation. *arXiv preprint arXiv:2507.18224*.

Yunxuan Li, Yibing Du, Jiageng Zhang, Le Hou, Peter Grabowski, Yeqing Li, and Eugene Ie. 2024. Improving multi-agent debate with sparse communication topology. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 7281–7294.

Tian Liang, Zhiwei He, Wenxiang Jiao, Xing Wang, Yan Wang, Rui Wang, Yujiu Yang, Shuming Shi, and Zhaopeng Tu. 2024. Encouraging divergent thinking in large language models through multi-agent debate. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 17889–17904.

Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. 2017. Program induction by rationale generation: Learning to solve and explain algebraic word problems. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 158–167.

Bang Liu, Xinfeng Li, Jiayi Zhang, Jinlin Wang, Tanjin He, Sirui Hong, Hongzhang Liu, Shaokun Zhang, Kaitao Song, Kunlun Zhu, and 1 others. 2025a. Advances and challenges in foundation agents: From brain-inspired intelligence to evolutionary, collaborative, and safe systems. *arXiv preprint arXiv:2504.01990*.

Yixin Liu, Guibin Zhang, Kun Wang, Shiyuan Li, and Shirui Pan. 2025b. Graph-augmented large language model agents: Current progress and future prospects. *arXiv preprint arXiv:2507.21407*.

Rui Miao, Yixin Liu, Yili Wang, Xu Shen, Yue Tan, Yiwei Dai, Shirui Pan, and Xin Wang. 2025. Blindguard: Safeguarding llm-based multi-agent systems under unknown attacks. *arXiv preprint arXiv:2508.08127*.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are NLP models really able to solve simple math word problems? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2080–2094.

Chen Qian, Zihao Xie, Yifei Wang, Wei Liu, Yufan Dang, Zhuoyun Du, Weize Chen, Cheng Yang, Zhiyuan Liu, and Maosong Sun. 2025. Scaling large-language-model-based multi-agent collaboration. In *International Conference on Learning Representations*.

Subhro Roy and Dan Roth. 2015. Solving general arithmetic word problems. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1743–1752.

Yashar Talebirad and Amirhossein Nadiri. 2023. Multi-agent collaboration: Harnessing the power of intelligent llm agents. *arXiv preprint arXiv:2306.03314*.

Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, and 1 others. 2024a. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6):186345.

Song Wang, Zhen Tan, Zihan Chen, Shuang Zhou, Tianlong Chen, and Jundong Li. 2025a. Anymac: Cascading flexible multi-agent collaboration via next-agent prediction. *arXiv preprint arXiv:2506.17784*.

Xu Wang, Zihao Li, Benyou Wang, Yan Hu, and Difan Zou. 2025b. Model unlearning via sparse autoencoder subspace guided projections. *Preprint*, arXiv:2505.24428.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations*.

Zhenhailong Wang, Shaoguang Mao, Wenshan Wu, Tao Ge, Furu Wei, and Heng Ji. 2024b. Unleashing the emergent cognitive synergy in large language models: A task-solving agent through multi-persona self-collaboration. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 257–279.

Zhexuan Wang, Yutong Wang, Xuebo Liu, Liang Ding, Miao Zhang, Jie Liu, and Min Zhang. 2025c. Agentdropout: Dynamic agent elimination for token-efficient and high-performance llm-based multi-agent collaboration. *arXiv preprint arXiv:2503.18891*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, and 1 others. 2025. The rise and potential of large language model based agents: A survey. *Science China Information Sciences*, 68(2):121101.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*.

Zihao Yi, Jiarui Ouyang, Yuwen Liu, Tianhao Liao, Zhe Xu, and Ying Shen. 2024. A survey on recent advances in llm-based multi-turn dialogue systems. *arXiv preprint arXiv:2402.18013*.

Guibin Zhang, Yanwei Yue, Zhixun Li, Sukwon Yun, Guancheng Wan, Kun Wang, Dawei Cheng, Jeffrey Xu Yu, and Tianlong Chen. 2025a. Cut the crap: An economical communication pipeline for llm-based multi-agent systems. In *International Conference on Learning Representations*.

Guibin Zhang, Yanwei Yue, Xiangguo Sun, Guancheng Wan, Miao Yu, Junfeng Fang, Kun Wang, Tianlong Chen, and Dawei Cheng. 2025b. G-designer: Architecting multi-agent communication topologies via graph neural networks. In *Forty-second International Conference on Machine Learning*.

Jintian Zhang, Xin Xu, Ningyu Zhang, Ruibo Liu, Bryan Hooi, and Shumin Deng. 2024a. Exploring collaboration mechanisms for llm agents: A social psychology view. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14544–14607.

Kechi Zhang, Jia Li, Ge Li, Xianjie Shi, and Zhi Jin. 2024b. CodeAgent: Enhancing code generation with tool-integrated agent systems for real-world repo-level coding challenges. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13643–13658.

Yusen Zhang, Ruoxi Sun, Yanfei Chen, Tomas Pfister, Rui Zhang, and Sercan Arik. 2024c. Chain of agents: Large language models collaborating on long-context tasks. *Advances in Neural Information Processing Systems*, 37:132208–132237.

Andy Zhou, Kai Yan, Michal Shlapentokh-Rothman, Haohan Wang, and Yu-Xiong Wang. 2024. Language agent tree search unifies reasoning, acting, and planning in language models. In *Proceedings of the 41st International Conference on Machine Learning*, pages 62138–62160.

Han Zhou, Xingchen Wan, Ruoxi Sun, Hamid Palangi, Shariq Iqbal, Ivan Vulić, Anna Korhonen, and Sercan Ö Arık. 2025. Multi-agent design: Optimizing agents with better prompts and topologies. *arXiv preprint arXiv:2502.02533*.

Mingchen Zhuge, Wenyi Wang, Louis Kirsch, Francesco Faccio, Dmitrii Khizbullin, and Jürgen Schmidhuber. 2024. Gptswarm: Language agents as optimizable graphs. In *Forty-first International Conference on Machine Learning*.

A Pseudo-code for EIB-LEARNER

Algorithm 1 provides the detailed pseudo-code of EIB-LEARNER, which takes the current query as input. EIB-LEARNER comprises a lightweight language encoder, two GNNs, and a gated fusion network, ensuring efficiency during execution. In training, a subset of queries is used to construct communication topologies via EIB-LEARNER, followed by multi-agent decision-making based on the generated topologies.

Table 3: Comparison of accuracy, time, and cost across different agent configurations. We sample 1000 questions from MMLU benchmark and utilize GPT-3.5 as the base LLM.

#Agents	5	7	9
Chain			
Accuracy (%)	63.33	64.55	65.07
Time (min)	53.11	90.94	126.15
Cost (USD)	4.18	10.04	15.38
Complete Graph			
Accuracy (%)	61.83	62.16	62.71
Time (min)	80.85	114.29	236.44
Cost (USD)	5.78	14.49	31.67
G-Designer			
Accuracy (%)	64.08	65.82	66.01
Time (min)	57.14	94.32	138.18
Cost (USD)	4.30	11.37	16.35
EIB-LEARNER			
Accuracy (%)	**65.47**	**67.14**	**67.82**
Time (min)	59.38	96.21	141.89
Cost (USD)	4.36	11.97	17.02

B Scalability in the number of agents

To evaluate the scalability of EIB-LEARNER to larger multi-agent settings, we report its performance across systems with 5 to 9 agents, as shown in Table 3. Notably, EIB-LEARNER achieves the most significant performance gains as the number of agents increases. More importantly, while maintaining comparable computational cost and runtime to G-Designer, EIB-LEARNER delivers consistently better results. Specifically, it achieves a 5.11% improvement in accuracy while consuming only about half the token cost of a fully connected topology. These results demonstrate the scalability and potential of our method for enabling large-scale, autonomous multi-agent collaboration.

C Dataset Statistic

We present the dataset statistics in Table 4, following the same experimental setup as G-Designer (Zhang et al., 2025b).

D Case Study

For the HumanEval task (case A in Figure 5), which requires symbolic reasoning and program synthesis, EIB-LEARNER constructs a hierarchical communication structure: the **Project Manager** initiates task planning and forwards it to the **Algorithm Designer**, who devises logic that is implemented by the **Programming Expert**. Outputs are then verified by the **Test Analyst** and corrected by the **Bug Fixer**, forming a feedback loop that suppresses local errors while ensuring functional accuracy.

For the GSM8K problem (case B in Figure 5), calculating how many friends can attend a party under budget constraints. EIB-LEARNER topology to balance efficiency and error control. The **Mathematical Analyst** models the problem and communicates with the **Math Solver** for final computation. Simultaneously, the **Programming Expert** provides structural parsing to the **Inspector**, who ensures the calculation logic is sound, avoiding propagation of misunderstandings.

These topologies align with our dual-view design by enabling efficient propagation of correct reasoning while suppressing misleading signals through targeted verification pathways.

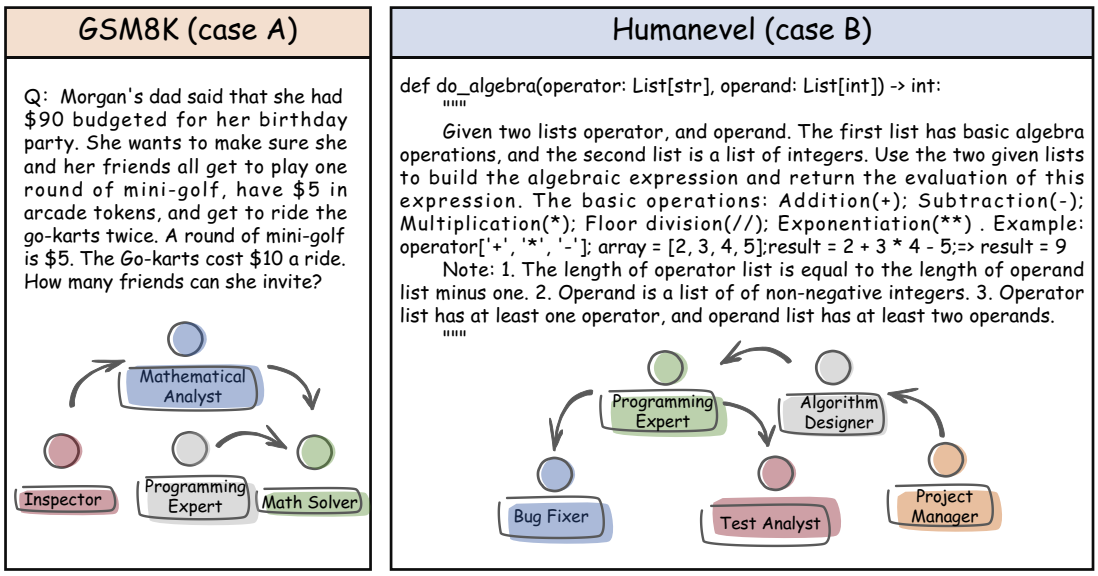


Figure 5: Case study of the communication topologies designed by EIB-LEARNER on HumanEval and GSM8K benchmarks.

Table 4: Dataset descriptions and statistics.

Category	Dataset	Answer Type	Metric	#Test	License
General reasoning	MMLU	Multi-choice	Acc.	153	MIT License
Math reasoning	GSM8K	Number	Acc.	1,319	MIT License
	MultiArith	Number	Acc.	600	Unspecified
	SVAMP	Number	Acc.	1,000	MIT License
	AQuA	Multi-choice	Acc.	254	Apache-2.0
Code generation	HumanEval	Code	Pass@1	164	MIT License

Algorithm 1: Designing workflow of EIB-LEARNER

Input: Input query $\mathcal{Q}$, a NodeEncoder, two GNN, a gating function, learning rate α

for *query* $d \in \{1, 2, \ldots, D'\}$ **do**

/* Establish multi-agent network */

for *node* $i \in \{1, 2, \ldots, N\}$ **do**

$\mathbf{x}_i \leftarrow \text{NodeEncoder}\big(\mathcal{T}(\text{Role}_i), \mathcal{Q}_d\big)$;

Obtain agent embeddings $\mathbf{X} \leftarrow [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N]^\top$;

Obtain query embeddings $\mathbf{Q}_d \leftarrow \text{NodeEncoder}\big(\mathcal{Q}_d\big)$;

/* Forward process of EIB-Learner */

Utilize the fully connected graph and chain to obtain two adjacency matrices $\mathbf{A}_{\text{dense}}$ and $\mathbf{A}_{\text{sparse}}$;

Utilize $\mathbf{A}_{\text{dense}}$ and $\mathbf{A}_{\text{sparse}}$ to encode $\mathcal{G}$ into latent representations and decoding results in two soft masks $\mathbf{M}_{\text{dense}}$ and $\mathbf{M}_{\text{sparse}}$:

$\mathbf{Z}_{\text{dense}} = \text{GNN}(\mathbf{A}_{\text{dense}}, \mathbf{X})$, $\mathbf{M}_{\text{dense}} = \sigma(\mathbf{Z}_{\text{dense}}^T \mathbf{Z}_{\text{dense}})$

$\mathbf{Z}_{\text{sparse}} = \text{GNN}(\mathbf{A}_{\text{sparse}}, \mathbf{X})$, $\mathbf{M}_{\text{sparse}} = \sigma(\mathbf{Z}_{\text{sparse}}^T \mathbf{Z}_{\text{sparse}})$

Utilize a gating function to calculate the weight α based on the query $\mathcal{Q}$:

$\alpha = \text{softmax}(\mathbf{W}_2 \cdot \text{ReLU}(\mathbf{W}_1 \mathbf{Q}_d + \mathbf{b}_1))$

Fusing two masks based on their weights:

$\mathbf{M}_{\text{final}} = \alpha_{\text{dense}} \cdot \mathbf{M}_{\text{dense}} + \alpha_{\text{sparse}} \cdot \mathbf{M}_{\text{sparse}}$.

Obtain the new communication topology $\mathcal{G}$ by sampling from $\mathbf{M}_{\text{final}}$

/* Guide multi-agent collaboration */

Obtain the the execution order of agents by topological sorting $\sigma = [v_{\sigma_1}, \ldots, v_{\sigma_N}]$

for $t \leftarrow 1$ **to** K **do**

for *node* $i \in \sigma$ **do**

$\mathcal{R}_i^{(t)} = v_i(\mathcal{P}_{\text{sys}}, \mathcal{P}_{\text{usr}}^{(t)})$;

$\mathcal{P}_{\text{usr}}^{(t)} = \{\mathcal{Q}\} \cup \{\mathcal{R}_j^{(t)} | v_j \in \mathcal{N}(v_i)\}$;

$\mathcal{O}^{(t)} = \text{Aggregate}(\mathcal{R}_i^{(t)} \ldots \mathcal{R}_n^{(t)})$;

/* Update G-Designer parameters */

$\Theta^{d+1} \leftarrow \Theta^d - \alpha \nabla_\Theta \mathcal{L}_{\text{EIB-LEARNER}}$;