ISSN:2790-1688 Volume-15-(2025)

EDCN: Expert-Driven Dynamic Recruitment and Collaboration Network for MAS

Wending Zhang*

Beijing-Dublin International College at BJUT, Beijing University Of Technology, Beijing, China.

wending.zhang@ucdconnect.ie

Abstract. Large Language Model (LLM)-driven Multi-Agent Systems (MAS) have recently demonstrated potential in complex reasoning tasks. However, real-world scenarios often impose higher demands on *"who should collaborate, how they should collaborate, and whose opinions should carry more weight."* Fixed roles and static orchestration can easily lead to capability mismatches and redundant overhead. To address this issue, this paper proposes a task-adaptive collaboration framework for multi-agent systems. Expert agents first decompose natural language tasks and dynamically recruit the most suitable executors. During execution, a directed collaboration network is constructed, where peer evaluation and historical performance are incorporated into a weighted voting scheme, allowing more stable and reliable agents to gain greater decision-making power. After task completion, evaluation results are fed back into subsequent recruitment, enabling continuous evolution. This paper conducts comparative experiments on a causal reasoning dataset. On the first 30 samples, the system achieved 22/30 correct answers, with an accuracy of 73.3%, outperforming single-agent GPT (63.3%) and majority-vote MAS without evaluation (66.7%). These results verify the advantages of the proposed framework in terms of both correctness and robustness.

Keywords: Multi-Agent Systems, Large Language Models, Team Optimization.

1. Introduction

Multi-Agent Systems (MAS) are gradually becoming a key mechanism for enabling Large Language Models (LLMs) to achieve human-like reasoning and judgment capabilities in complex tasks [1]. With the continuous advancement of LLM capabilities, LLM-driven agents have been widely applied in tasks such as question answering, generation, search, and reasoning [2]. However, a single agent often suffers from limitations such as knowledge blind spots and a narrow reasoning path, which motivates researchers to explore collaborative reasoning and *Collective Intelligence* among groups of agents [3]. In existing practice, most MAS architectures adopt static organizational structures, with predefined numbers, types, and execution logics of agents [4]. In fact, such systems lack the ability to dynamically adapt to the characteristics of the tasks themselves, and they exhibit significant bottlenecks in agent recruitment, task scheduling, collaborative voting, and feedback updating [5]. In particular, when MAS systems are applied to tasks with high uncertainty or semantic complexity, e.g., moral judgment, causal reasoning, multi-step question answering, current systems reveal the following three major challenges.

Challenge 1: Static structures constrain system performance. Static structures fix the roles, capabilities, and interaction order of agents, making it difficult to respond to diverse skill combinations and levels of task decomposition required by different tasks [6].

Challenge 2: Lack of performance feedback leads to agent redundancy or misguided system decisions. Traditional systems lack effective mechanisms for scoring and filtering agents, resulting in low-quality agents participating in the reasoning process over time, and even producing erroneous outputs due to LLM hallucinations [7].

Challenge 3: Single collaboration mode prevents efficient integration and self-optimization. Most systems rely on simple majority voting or chain-of-thought decisions, lacking weighted mechanisms under "minority obeys majority" and adaptive agent selection strategies, thus failing to achieve evolutionary capabilities [5].

To address the above challenges, this paper proposes a MAS framework that supports task-adaptive decomposition, dynamic expert agent generation and scheduling, and agent performance

scoring and evolution. The proposed system has the following design highlights. The first is the s**tructured task modeling and decomposition.** The system automatically analyzes the original task description with LLMs and decomposes it into structured subtasks, including required skills, complexity, and dependencies, thereby forming a task graph. The second is the d**ynamic expert generation and agent recruitment mechanism.** Based on task requirements and agents' historical performance, the system dynamically recruits agents from a candidate pool to form an optimal execution set. The third is the w**eighted voting and scoring update mechanism.** Instead of simple majority voting, the system conducts weighted collaboration based on agents' dynamic scores, and after task execution, provides feedback on accuracy to update scores, enabling collective learning and evolution among agents.

This paper conducts an experimental evaluation of the proposed system on a causal reasoning dataset, comparing it with majority-voting structures without scoring mechanisms and single-agent systems. Experimental results demonstrate that the system significantly outperforms baseline methods in terms of accuracy, stability, and agent selection capability, showing promising scalability and potential for real-world deployment.

2. Relevant Research

2.1 Dynamic Scheduling Mechanisms: Breaking the Bottleneck of Fixed Structures in Collaboration

In traditional approaches, MAS commonly adopt static organizational structures, meaning that the number, roles, and interaction logic of agents are predefined during system initialization. This mode demonstrated good controllability and engineering feasibility in early, relatively simple collaborative tasks. However, against the backdrop of increasingly complex task types and heightened uncertainty in agent capabilities, such structures reveal problems of poor flexibility and low resource utilization. In particular, for reasoning tasks with high semantic complexity or dense dynamic information, fixed agent orchestration structures cannot adjust resource allocation according to the actual task load, which significantly limits the adaptability and response efficiency of the system [9].

To address this problem, the DyLAN system has proposed a reasoning-process-based structural dynamic adjustment mechanism. In each reasoning round, the system evaluates agents' contribution to responses based on the Agent Importance Score (AIS) and eliminates agents with low scores, thereby achieving structural optimization and resource contraction during the reasoning stage. This mechanism does not rely on external annotations and possesses strong task independence and extensibility. On multiple reasoning benchmarks, DyLAN demonstrates higher accuracy and efficiency, with particularly notable improvements over static MAS systems in multi-step question answering and complex propositional tasks [9]. Similarly, the AgentVerse system, inspired by human organizational collaboration, introduces a task-driven recruitment mechanism. Instead of presetting agent roles, a recruiter agent generates candidate expert roles and their capability configurations according to the current task state, and dynamically assembles a collaborative team. Unlike DyLAN's elimination mechanism, AgentVerse adopts a top-down proactive team-building strategy. The capabilities of agents are described in brief resumes written by an LLM, and the system shows good transferability in real-world task scenarios such as software development and consulting [10].

However, both approaches have certain limitations in practical application. For example, although DyLAN provides an efficient scoring and structural optimization mechanism, it focuses primarily on eliminating inefficient agents, lacking proactive mechanisms based on task requirements, and thus cannot flexibly respond to diverse task demands. Similarly, although AgentVerse introduces an expert generation and recruitment mechanism that is closer to real-world collaboration, it lacks precise self-regulation and updating, which limits the long-term adaptability of its agent team formation.

To overcome these limitations, this paper proposes a MAS framework that integrates the advantages of both methods. Based on structured task modeling, the framework leverages LLMs for task-adaptive decomposition and dynamically generates expert agents with specific professional

capabilities. Combined with historical scoring information, the system recruits the most suitable agents from the candidate pool in real time to form a collaborative team, thereby integrating task-driven proactive expert generation with a refined scoring feedback mechanism. Through this dual mechanism, this approach not only maintains flexible team formation but also enables adaptive updating of agents and enhances the system's long-term generalization performance, addressing the shortcomings of both DyLAN and AgentVerse.

2.2 Adaptive Selection Mechanisms

In the construction of MAS, traditional methods typically rely on static role configurations and fixed agent orchestration logic. However, such structures struggle to achieve real-time resource adjustment and collaboration optimization when faced with changing task requirements and dynamic fluctuations in agent performance. This often results in wasted computational resources, reduced response efficiency, and severely constrains the adaptability of systems in complex environments [11].

To address this issue, the DyLAN system proposed an adaptive mechanism for dynamically adjusting agent selection based on task feedback. Specifically, in the reasoning process, DyLAN organizes reasoning tasks into multi-layer feedforward networks, with each layer consisting of multiple candidate agents, forming a sequential linkage structure through information transmission. To achieve real-time adaptation to task characteristics, DyLAN introduces an *Inference-time Agent Selection* mechanism, where an additional LLM-based ranker evaluates each agent's response quality and task alignment in real time. Agents that either mismatch the current task characteristics or demonstrate poor performance are actively eliminated, while those better adapted to the current task state are retained and activated. Furthermore, DyLAN employs the *Byzantine Consensus*, in which the reasoning process is automatically terminated when more than two-thirds of agents in a given layer reach consensus on the task, thereby achieving efficient utilization of computational resources. By combining task feedback with dynamic agent updates, this selection method enables DyLAN to proactively adapt to task difficulty and environmental changes, demonstrating superior reasoning accuracy and resource efficiency compared to static systems across datasets such as *MATH* and *HumanEval* [10].

Despite these improvements in collaboration efficiency and adaptability, DyLAN's selection mechanism primarily relies on scoring strategies during interactions, lacking dynamic modeling of agents' historical capabilities and accumulation of trust. This limitation makes the system prone to erroneously eliminating critical agents during tasks, thereby weakening overall collaboration efficiency and stability [12]. In addition, because DyLAN's agent selection logic emphasizes negative filtering to remove unqualified agents rather than proactively identifying high-quality agents most aligned with task requirements, it may lead to the absence of key roles during task execution, which restricts system performance in high-complexity tasks.

2.3 Scoring Mechanisms: Optimizing Collaboration Quality and Quantifying Agent Contribution

In multi-agent collaborative tasks, another key challenge lies in how to quantify the actual contribution of each agent during the overall reasoning process. Especially in multi-round interaction scenarios, scoring mechanisms in traditional multi-agent systems often rely on a single metric, such as accuracy or relevance, to evaluate agents, which fails to reflect their reasoning capabilities in complex tasks. Moreover, traditional scoring mechanisms often lack dynamic adjustment, making it difficult to update evaluations according to task variations or the effectiveness of agent collaboration [13].

To address these limitations of traditional multi-agent scoring mechanisms, Reference [14] proposes a scoring and selection method that combines individual task performance with network structural importance. By employing *critical node* identification, the system prioritizes retaining high-value agents in the collaboration network to enhance overall system performance. However, this

method still exhibits limitations in terms of adaptability to dynamic and multi-round tasks. In multi-round interactive tasks, the agent selection strategy fails to fully leverage historical task performance to guide subsequent recruitment. At the same time, the weighting of individual performance and structural importance does not incorporate adaptive adjustment, resulting in insufficient stability across different task scenarios.

To overcome these shortcomings, this paper proposes a dynamic scoring mechanism that integrates individual execution performance with group structural information. In this mechanism, after completing a subtask, each agent receives a *performance score (P)* calculated by combining its basic performance score, contribution score, and historical performance. Through a *weighted voting mechanism*, the scores are fed back in real time to subsequent agent recruitment stages, thereby enabling global evolution and adaptive adjustment of the agent group. This method not only enhances the influence of efficient agents in current tasks but also gradually fosters a high-quality and highly collaborative agent group in long-term, multi-round collaborations, effectively overcoming the limitations of static scoring methods in dynamic environments.

3. Methodology

3.1 System Overview

To effectively address the challenges of varying skill requirements, non-unique solution paths, and instability of language models in complex natural language tasks, this paper designs and implements a MAS architecture that incorporates task decomposition, dynamic expert coordination, collaboration network scoring, and evolutionary mechanisms. The execution process of the entire system is task-driven and consists of three phases, structured understanding, distributed execution, and self-optimization, thus forming a closed-loop collaborative system.

Specifically, the system first receives a natural language problem as input and employs an *ExpertAgent* model to analyze the task and automatically decompose it into a set of structured subtasks. Each subtask contains its semantic description, required skills, complexity label, and dependency information. The *ExpertAgent* then oversees the task structure and, based on the requirements of each subtask and the capabilities of available agents in the system, recruits the most suitable executors. During this process, the system comprehensively considers factors such as agents' historical performance and task complexity to ensure the optimal overall capability of the selected agents.

All recruited agents individually respond to their assigned subtasks, and the results are ultimately combined into the answers for the subtasks. It is worth emphasizing that agents do not work in isolation: after subtask execution, the system constructs a *Collaboration Network* based on the participating agents. In this network, nodes represent specific agents, and edge weights reflect evaluations between agents. The system further evaluates agent contributions through this graph and dynamically updates each agent's overall score by combining task performance scores with contribution scores. This scoring mechanism feeds back into agent recruitment for future tasks, establishing a more selective reuse mechanism. High-scoring agents are more likely to be selected in subsequent tasks, while low-performing agents gradually receive fewer opportunities.

By integrating task-driven dynamic expert agent recruitment with graph-based agent performance evolution, the proposed system not only enables high-quality collaborative reasoning for complex tasks but also provides long-term evolutionary self-optimization. As such, it offers a generalizable and scalable solution for the practical deployment of multi-agent collaborative architectures.

3.2 Dynamic Expert Recruitment Mechanism

The dynamic expert recruitment mechanism is illustrated in Fig. 1. In the multi-agent collaborative system, the **Expert Agent** serves as the core coordinator. Its task is not only to understand the task structure but, more importantly, to precisely recruit the most suitable execution agents from the candidate pool according to task requirements, while maximizing overall reasoning capacity without

compromising system efficiency. The execution process of the **Expert Agent** mainly consists of two stages: providing a structured sub-task graph and sub-task-driven agent recruitment.

In the first stage, the **Expert Agent** receives a Task and provides a structured sub-task graph. All sub-tasks are collected into a set denoted as Sub Task=$\{St_1, St_2 \ldots St_n\}$. Each sub-task contains a clear semantic description, a list of required skills, and a complexity label.

In the second stage, the **Expert Agent** recruits several (k) suitable agents from an agent set denoted as Agent Set = $\{A_1, A_2 \ldots A_n\}$ to form an Agent Team = $\{a_1, a_2 \ldots a_k\}$ for each sub-task. This process does not rely on preset rules or static configurations. Instead, it dynamically evaluates the suitability of each available agent by leveraging the natural language reasoning capability of large language models to assess the match between agents and tasks. The suitability is comprehensively determined based on three factors.

Historical performance: the agent's *performance score*, representing its past performance within the system.

Skill alignment: the degree of match between the agent's skill tags and the skills required by the sub-task.

Complexity adaptation: the requirement that task complexity imposes on the agent's level of experience, ensuring that low-capability agents are not assigned to high-risk tasks.

This process calls large language models to generate matching scores. For each sub-task, the system selects the agent with the highest score from the candidate agent pool for binding, thereby ensuring task execution that is both targeted and efficient.

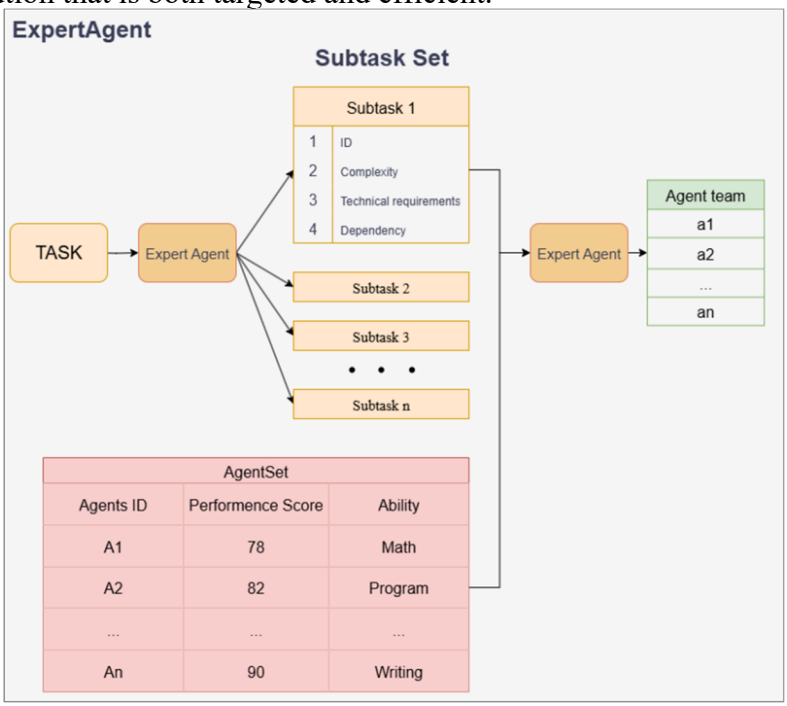

Figure 1 Dynamic Expert Sub-task Decomposition and Sub-task Recruitment Structure Diagram

3.3 Agent Collaboration Network

To enhance the reliability of agent recruitment strategies and improve the overall collaboration efficiency of the system, this study constructs an **Agent Collaboration Network** based on a graph structure after the Expert Agent completes the recruitment process, as illustrated in Fig. 2. This network is designed to compute each agent's collaborative potential and structural role, thereby enabling dynamic adjustment of their comprehensive scores. Such a mechanism not only strengthens the system's structural understanding of agent performance but also provides quantitative support for subsequent selection and evolution.

The core idea behind constructing this network is to organize all agents participating in the current task execution into a **directed graph**, where nodes represent agents and edges are weighted according

to evaluation metrics among agents. These metrics comprehensively account for other agents' evaluations of a given agent's performance in the current task, the task completion outcomes, as well as the agent's participation record and success rate in past tasks, i.e., its current performance score.

The network construction process is as follows. The system first iterates over each agent, treating each agent as a node, and then assigns scores between every pair of agents, using these scores as edge weights to form a structured directed graph. This design avoids the redundancy and computational overhead associated with fully connected networks while preserving the essential structural information. By analyzing this graph, the system can quantify each agent's collaborative contribution to the sub-task, denoted as the **contribution score** $C\_i\hat{}t$, which represents the collaborative contribution score of agent *i* at round *t*. The process continues for a total of *r* rounds, and the resulting contribution scores are integrated with task completion outcomes to update the agent's overall score.

$$C_i^r = \sum_{t=1}^{r} S_i^t / r \tag{1}$$

C_i^t represents the score of agent *i* at round *r*. S_i^t represents the score of agent *i* at round *t*.

The introduction of this graph-based mechanism ensures that agents are no longer treated as isolated executors, but rather as participants in a dynamic interactive network. This allows the system to continuously update the structural collaboration patterns among agents.

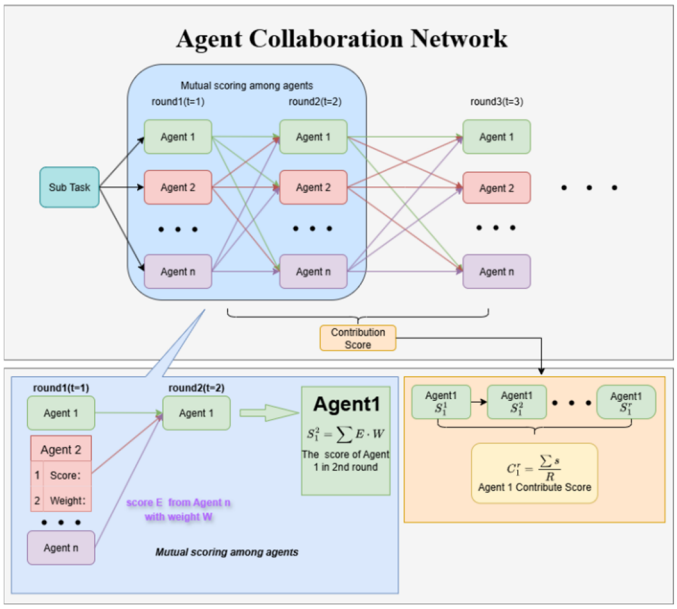

Figure 2 Agent Collaboration Network structure

3.4 Weighted Scoring Mechanisms

For the collaboration network, in order to achieve preferential selection of agents and enhance the overall performance of the system, this study designs a scoring mechanism that integrates both individual execution performance and collective structural information. This mechanism is not only applied to the weighted voting process of agents in the current task but is also continuously fed back into subsequent stages of agent recruitment, thereby promoting the gradual formation of a high-quality and highly collaborative agent group.

The scoring process begins at the task execution stage: each agent independently completes its assigned sub-task and outputs the corresponding solution. Subsequently, collaboration scores are updated based on the interactions of other agents within the collaboration network. A weighted scoring mechanism is applied, with weights defined in this study as *W*, where the weight integrates the evaluation scores of agents. In this way, agents with stronger performance exert greater influence, thereby encouraging agents that demonstrate stable performance and high success rates in collaboration.

$$\mathrm{W_m^t} = S_m^{t-1} / \sum_{m=1}^{k} S_m^{t-1} \tag{2}$$

$$S_i^t = \sum_{m=1}^{k} E_m^i \times \mathrm{W_m^t} \quad (3)$$

W_m^t denotes the weight assigned by agent *m* to other agents at round *t*. S_i^t denotes the score of agent *i* at round *t*, and E_m^i denotes the rating given by agent *m* to agent *i*.

The system uses this scoring mechanism to obtain the agents' final *consensus score* C_i^r for the sub-task. The agents' solutions are then aggregated, and a weighted voting procedure is performed, where the weights, referred to as *Result Weight*, *RW*, correspond to the agents' scores. Agents with higher scores exert greater influence on the final outcome, ultimately producing the overall output *R* of the sub-task.

$$RW_i = C_i^r / \sum_{i=1}^{k} C_i^r \quad (4)$$

RW_i denotes the weight of agent *i*'s response for the sub-task.

Furthermore, this scoring is also fed back into the recruitment of agents for subsequent tasks, enabling global evolution. The agent score update process is based on the following three types of scores.

Basic Score: In this paper, the *Basic Score* B_i is defined as the baseline performance score of agent *i*. After each task is completed, the system evaluates the result using the LLM's experience combined with the ground truth, assigning a score in the range of 0 to 100.

Contribution Score: Denoted as C_i^r. this score is obtained from the interactive collaboration network.

Initial Score: The initial score is derived from the agent's historical performance prior to the current task.

Overall Score Update Logic (Performance Score/*P*):

$$P_i^{r+1} = 0.3\boldsymbol{B_i} + \mathbf{0.3}\mathrm{C_i^r} + 0.4P_i^r \quad (5)$$

The updated overall score *P* serves as a reference for the **Expert Agent** when selecting agents and is also used as the initial score of agents when constructing the collaboration network. This mechanism combines structural feedback with task performance to enable continuous adjustment of each individual agent's capabilities. Agents that perform well in tasks become core contributors to the system, while inefficient or unstable agents experience a decrease in their scores, reducing their likelihood of being selected. This process continuously improves the overall quality and efficiency of the system.

The formation of the network, as well as the application and update of scores, is illustrated in Algorithm 1.

Algorithm 1: r-round Agent Collaboration, Weighted Voting, and Performance Update (0–100 scale)

Data: Task index t; participating agents A_t with size $k=|A_t|$;
prior performance $P_i^{(t)} \in [0,100]$ for $i \in A_t$ (initialize $P_i^{(1)} \leftarrow 70$);
each agent's answer $a_i^{(t)}$ and its correctness $\mathrm{corr}_i^{(t)} \in \{0,1\}$ (or a rubric in $[0,1]$);
peer evaluations $E_{ji} \in [0,100]$ assigned by rater j to ratee i;
number of peer-aggregation rounds $r \geq 1$; stability constant $\varepsilon > 0$.
Definition: $\mathrm{clip}_{[0,100]}(x) \triangleq \min(100, \max(0, x))$.

Result: Contribution scores $C_i^r \in [0,100]$, voting weights RW_i, aggregated output $\hat{y}^{(t)}$, and updated $P_i^{(t+1)}$.

Phase I: Round-0 initialization;
foreach $i \in A_t$ **do**
  $S_i^{(0)} \leftarrow P_i^{(t)}$; // start on 0--100 scale
end foreach
Phase II: r-round peer-weighted propagation;
for $s \leftarrow 1$ **to** r **do**
  // (2) Normalize rater weights from previous round scores
  $Z \leftarrow \sum_{j \in A_t} (S_j^{(s-1)} + \varepsilon)$;
  foreach $m \in A_t$ **do**
    $W_m^{(s)} \leftarrow \dfrac{S_m^{(s-1)} + \varepsilon}{Z}$; // $\sum_m W_m^{(s)} = 1$
  end foreach
  // (3) Peer-weighted score update (no self-rating by default)
  foreach $i \in A_t$ **do**
    $S_i^{(s)} \leftarrow \sum_{\substack{m \in A_t \\ m \neq i}} E_{mi} W_m^{(s)}$; // $S_i^{(s)} \in [0,100]$
  end foreach
end for
Phase III: Contribution, weighted voting, and performance update;
foreach $i \in A_t$ **do**
  // (1) Contribution over r rounds (0--100)
  $C_i^r \leftarrow \dfrac{1}{r} \sum_{s=1}^{r} S_i^{(s)}$;
end foreach
Normalize voting weights: $RW_i \leftarrow \dfrac{C_i^r}{\sum_{j \in A_t} C_j^r}$; // $\sum_i RW_i = 1$
Aggregate final output (weighted voting): $\hat{y}^{(t)} \leftarrow \mathrm{WeightedVote}(\{(a_i^{(t)}, RW_i)\}_{i \in A_t})$;
foreach $i \in A_t$ **do**
  // Basic execution score on 0--100 (binary or rubric)
  $B_i^{(t)} \leftarrow 100 \cdot \mathrm{corr}_i^{(t)}$;
  // (5) Performance update on 0--100 scale
  $P_i^{(t+1)} \leftarrow \mathrm{clip}_{[0,100]}\left(0.3\, B_i^{(t)} + 0.3\, C_i^r + 0.4\, P_i^{(t)}\right)$;
end foreach

4. Experiment

4.1 Data Set

To evaluate the effectiveness of our multi-agent collaboration system in handling tasks that require nuanced human-like reasoning, we employed the *causal judgment* dataset from the BIG-bench benchmark suite. This dataset focuses on questions related to *causal attribution* and *intent judgment*, typically set in real-world moral and social contexts.

4.1.1 Dataset Description

The original dataset is stored in JSON format (causal_judgment.json) and contains multiple examples. Each example includes a natural language input describing a particular scenario, a target_scores dictionary indicating human-preferred option scores, e.g., {"Yes": 1, "No": 0}, and an optional comment field providing a reference or explanation for the item. Each task instance asks whether a certain action or outcome is "*intentional"* or "*caused by a specific person"*, simulating human social and moral reasoning.

A representative example is as follows.

A company CEO is sitting in his office when his vice president enters and says, "We are considering launching a new project that will help us increase profits but will also harm the environment." The CEO responds that he does not care about the environmental damage and only wants to maximize profits. The project is eventually executed, profits increase, and the environment is harmed. The question is: Did the CEO "*intentionally"* harm the environment?

The processed version of the task is summarized in the table below:

Promt
Here is the question: How would a typical person answer each of the following questions about causation?And provide the reasoning process. \n\n[scenario text]\n\n Options: Yes/No

Through this processing method, each sample possesses a clear and evaluable structure, which can be used for prompt-based language model reasoning and automated evaluation against the ground truth, thereby facilitating the analysis of agent reasoning capabilities. In total, 30 samples were processed, each representing a binary classification task, making them suitable for testing reasoning accuracy and preference consistency.

4.1.2 Dataset Preprocessing

To adapt the dataset for the MAS, the study implemented a file processing function that converts the original JSON data into an instruction-based prompt format suitable for agent reasoning within the system.

The specific preprocessing steps are as follows.

Add a unified task prefix before the input, i.e., “How would a typical person answer...”, to clarify the task objective.

Append response options based on the keys in target_scores, e.g., “Options: Yes/No”.

Attach the target_scores and the comment as metadata for each data instance.

4.2 Baseline

To comprehensively evaluate the advantages of the proposed multi-agent collaboration system in handling complex reasoning tasks, the study designed several baseline methods, covering single-agent, large language model reasoning, and static multi-agent systems. The specific methods include the following.

4.2.1 Single-Agent GPT Reasoning (Without MAS)

This method does not employ a multi-agent system; instead, a single large language model independently performs the reasoning. The implementation involves directly providing the model with the task prompt, and the model generates the answer independently, without task decomposition, expert agent generation, scoring mechanisms, or agent collaboration processes. This method serves as a baseline, demonstrating the benchmark performance of a single large language model.

4.2.2 Majority-Voting MAS Without Collaboration Network or Scoring Mechanism

This method also employs multiple agents to generate complete answers for the main task in parallel. However, it does not incorporate the scoring mechanism proposed in this paper, that is, the historical performance of agents does not influence their decision weight. After all agents provide their answers, the system determines the final output using a simple majority-voting principle. This method emphasizes the advantage of collective decision-making but cannot effectively filter low-quality responses, and the overall result may be affected by poorly performing agents.

4.3 Results

To evaluate the performance advantage of the proposed MAS on causal judgment tasks, the study tested the preprocessed set of 30 causal judgment samples. The system was compared against the two baseline methods described above and recorded the overall score and accuracy. As shown in Table 1, the MAS system with fully enabled dynamic expert generation, agent collaboration network, and scoring mechanism successfully answered 22 out of the first 30 tasks, achieving the highest total task score, corresponding to an overall accuracy of **73.3%**. This result significantly outperforms the single-agent GPT reasoning method without MAS (19 correct answers, **63.3%** accuracy) and the majority-voting MAS without a scoring mechanism (20 correct answers, **66.7%** accuracy).

The full MAS system improved accuracy by **10 percentage points** compared to the single-agent approach and by **6.6 percentage points** compared to the MAS without a scoring mechanism. These results indicate that the MAS system incorporating dynamic expert generation and the scoring mechanism significantly enhances both reasoning correctness and stability. This advantage stems from two aspects.

Dynamic Expert Recruitment Mechanism. By adaptively selecting the agents whose abilities best match the subtask requirements during task decomposition and agent assignment, the system improves task-solving flexibility and precision, while reducing interference from irrelevant or low-quality responses.

Collaboration Network and Scoring Mechanism. These mechanisms mitigate the influence of low-quality agents, allowing high-quality agents to hold greater decision weight within tasks, thereby enhancing agent collaboration efficiency and answer reliability, ultimately improving the reliability of collective decisions.

The experimental results in Table 1 list the number of correct responses and corresponding accuracy for each method, facilitating a clearer comparison of performance differences across system configurations.

Table 1: Results on the first 30 tasks. The MAS with dynamic expert generation and agent scoring achieves the best correctness (22/30) and accuracy (73.3%), outperforming Single Agent GPT (63.3%) and majority-vote MAS without scoring (66.7%).

Method	Correct (/30)	Accuracy (%)
Single Agent GPT	19 / 30	63.3 (+0.0)
MAS (Majority Vote, no scoring)	20 / 30	66.7 (+3.4)
MAS Dynamic Expert + Scoring	**22 / 30**	**73.3** (+10.0)

5. Conclusion

To address the core bottlenecks of current multi-agent systems in task adaptability, structural flexibility, and collaboration efficiency, this paper proposes an LLM-driven multi-agent system featuring task-structured modeling, dynamic expert agent recruitment, and a weighted scoring mechanism. The system constructs a subtask graph based on the natural language input of complex tasks and dynamically schedules and executes subtasks through expert agents. During the collaboration process, a history-driven agent selection mechanism and a weighted voting rule are introduced, effectively enhancing system stability and reasoning consistency under high task uncertainty.

For experimental evaluation, a causal reasoning dataset was employed. On a randomly selected set of 30 task prompts, the system successfully completed 22 tasks, achieving an accuracy of 73.3%. Compared with the majority-voting system without a scoring mechanism and static-structured MAS, these results demonstrate a significant improvement in decision correctness and agent network collaboration, validating the effectiveness of the proposed approach in task generalization and agent quality control.

Compared with existing representative systems, such as the dynamic LLM-Agent network proposed by DyLAN and the expert recruitment mechanism of AgentVerse, notable differences exist. DyLAN emphasizes quantifying agent contributions through the Agent Importance Score during reasoning and performing structural pruning and optimization accordingly, whereas AgentVerse focuses on expert role generation and organization inspired by human collaboration, demonstrating high-quality responses in scenarios such as software development and consulting. The system proposed in this paper integrates the advantages of both approaches. It introduces a history-based agent recruitment mechanism and combines it with task-graph-driven expert scheduling, enabling

dynamic assignment of task responsibilities according to agent scores, while employing a weighted voting mechanism to enhance decision accuracy and stability. Although DyLAN excels in structural dynamism and scoring mechanisms, its agent recruitment relies on iterative feedback during the reasoning process. In contrast, AgentVerse emphasizes multi-role collaboration but lacks fine-grained scoring feedback and agent selection mechanisms. The system presented here offers a more coordinated and unified integration path between these two approaches.

Nevertheless, the current system still has certain limitations. The capabilities of agents remain highly dependent on the performance of the underlying large language model. When the model exhibits hallucinations or misinterprets context, it may lead to erroneous scoring or voting tendencies, thereby affecting the accuracy of the final output. In addition, although the system supports dynamic agent scheduling, in complex tasks, the agent network may form highly dense graph structures, resulting in increased computational overhead and response latency. Future work will need to further balance structural simplification with performance maintenance. Additionally, the system has not yet incorporated explicit memory or historical feedback mechanisms in task decomposition and scoring design, limiting agents' ability to retain long-term contextual information across multiple task rounds and constraining continuous learning and optimization.

For future work, it is planned to introduce more sophisticated collaboration mechanisms, such as integrating long- and short-term memory modules for agents to enhance context retention, enabling finer-grained dynamic dialogue and long-term knowledge transfer among agents. It is also aimed to conduct large-scale evaluations on broader benchmark datasets, e.g., MATH, MMLU, to further assess the system's generalization and stability across multi-domain reasoning tasks. Moreover, it is also planned to advance the system toward practical deployment by open-sourcing the complete training and inference framework, including task decomposition, expert scheduling, and scoring mechanisms, thereby providing reusable infrastructure and practical support for future research on multi-agent reasoning systems.

References

[1] B. Yan, L. Zhang, and Z. Zhou, "Beyond self-talk: A communication-centric survey of LLM-based multi-agent systems," arXiv preprint arXiv:2502.14321, 2024. [Online]. Available: https://arxiv.org/abs/2502.14321

[2] K. T. Tran, D. Dao, M. D. Nguyen, and Q. V. Pham, "Multi-agent collaboration mechanisms: A survey of LLMs," arXiv preprint arXiv:2501.06322, 2025. [Online]. Available: https://arxiv.org/abs/2501.06322

[3] I. S. Jacobs and C. P. Bean, "Fine particles, thin films and exchange anisotropy," in Magnetism, vol. III, G. T. Rado and H. Suhl, Eds. New York: Academic, 1963, pp. 271–350.Chudziak, J. A., & Kostka, A. (2025). Synergizing Logical Reasoning, Knowledge Management and Collaboration in Multi-Agent LLM System. arXiv:2507.02170. https://arxiv.org/abs/2507.02170

[4] Wu, Y., & Yang, Y. (2024). A survey on LLM-based multi-agent systems: workflow, infrastructure, and challenges. Vicinagearth. https://link.springer.com/article/10.1007/s44336-024-00009-2

[5] Zhang, Y., Mao, S., Ge, T., et al. (2024). LLM as a mastermind: A survey of strategic reasoning with large language models. arXiv:2404.01230. https://arxiv.org/abs/2404.01230

[6] Yan, B., Zhang, L., & Zhou, Z. (2024). Beyond self-talk: A communication-centric survey of LLM-based multi-agent systems. arXiv:2502.14321.https://arxiv.org/abs/2502.14321

[7] La Malfa, E., La Malfa, G., Marro, S., & Zhang, J. M. (2025). Large language models miss the multi-agent mark. arXiv:2505.21298. https://arxiv.org/abs/2505.21298

[8] M. Merdant, P. Vrba, and G. Koppensteiner, "Knowledge-based multi-agent architecture for dynamic scheduling in manufacturing systems," in 2008 6th IEEE International Conference on Industrial Informatics, Daejeon, Korea, Jul. 2008, pp. 1392–1397. doi: 10.1109/INDIN.2008.4618262.

[9] Anonymous, “Dynamic LLM-Agent Network: An LLM-Agent Collaboration Framework with Agent Team Optimization,” arXiv preprint arXiv:2403.12489, under review at ICLR 2024. [Online]. Available: https://arxiv.org/abs/2403.12489

[10] W. Chen, Y. Su, J. Zuo, et al., “AgentVerse: Facilitating Multi-Agent Collaboration and Exploring Emergent Behaviors in Agents,” arXiv preprint arXiv:2308.10848, 2023. [Online]. Available: https://arxiv.org/abs/2308.10848

[11] M. A. Shyaa, N. F. Ibrahim, Z. B. Zainol, R. Abdullah and M. Anbar, "Reinforcement Learning-Based Voting for Feature Drift-Aware Intrusion Detection: An Incremental Learning Framework," in IEEE Access, vol. 13, pp. 37872-37903, 2025, doi: 10.1109/ACCESS.2025.3544221.

[12] Nguyen, T.D., Bai, Q., Li, W. (2016). Capability-Aware Trust Evaluation Model in Multi-agent Systems. In: Booth, R., Zhang, ML. (eds) PRICAI 2016: Trends in Artificial Intelligence. PRICAI 2016. Lecture Notes in Computer Science(), vol 9810. Springer, Cham. https://doi.org/10.1007/978-3-319-42911-3_65

[13] Jiang, Aiwen and Wang, Duan and Peng, Chao and Wang, MingWen, Relational Reasoning Image Captioning Via Multi-Agent Retrieval-Augmented Generation. Available at SSRN: https://ssrn.com/abstract=5351688 or http://dx.doi.org/10.2139/ssrn.5351688

[14] Z. Jin, Y. Zhao, Z. Han, and B. Zhao, “Countering large-scale malicious multi-agent systems by consensus breakdown based on critical node identification,” IEEE Internet of Things Journal, 2025. [Online]. Available: https://ieeexplore.ieee.org/document/10938956