Evaluate, Scale, and Credit: A Comprehensive Study on Multi-Agent Collaboration of Large Language Models

Anonymous ACL submission

Abstract

Large Language Models based Multi-Agent Systems (LLM-MAS) perform well in many domains, but we still lack a clear understanding of the collaboration mechanism among multiple LLM-based agents. This study aims to explore three key issues: (1) Can multi-agent outperform single-agent systems? (2) Is scaling better for multi-agent systems? (3) How to credit agents and optimize collaboration? Specifically, we design five collaboration architectures and evaluate their effectiveness across different LLMs and tasks. Our findings offer significant insights for understanding the collaboration within MAS, optimizing collaboration architectures among agents, and reducing system costs. Furthermore, our conclusion will inspire and provide new perspectives for future studies on LLM-MAS.

1 Introduction

Large Language Model-based Multi-agent Systems (LLM-MAS) specialize multiple LLMs into different agents and effectively simulate complex real-world environments through the interaction among these diverse agents (Guo et al., 2024). With proven outstanding abilities in contextual understanding, reasoning, and generation, LLMs empower agents to collaboratively plan, discuss, and make decisions, imitating human team cooperation to solve real world problems (Li et al., 2023; Hong et al., 2023; Wu et al., 2023).

Recent research efforts have focused on exploring and optimizing the collaboration mechanisms of MAS driven by LLMs (Liang et al., 2023; Du et al., 2023; Chan et al., 2023), revealing two critical challenges: *architecture scaling* and *contribution crediting*. The challenge of *architecture scaling* encompasses expanding the number of agents and increasing their interaction frequency to solve more complex tasks (Zhang et al., 2023b; Chan et al., 2023; Li et al., 2024). However, while enhancing system capabilities, scaling also leads to a substantial rise in communication overhead, presenting a notable challenge in maintaining system efficiency (Zhang et al., 2023b; Yin et al., 2023). At the same time, the challenge of *contribution crediting* involves the accurate allocation of contributions among agents, which is crucial for promoting collaboration and ensuring interpretability and robustness within LLM-MAS systems (Liu et al., 2023b). Evaluating LLM-MAS systems from the perspectives of scaling and crediting not only diagnoses their current shortcomings and limitations but also directs future developments toward more efficient, effective, and scalable multi-agent collaborations.

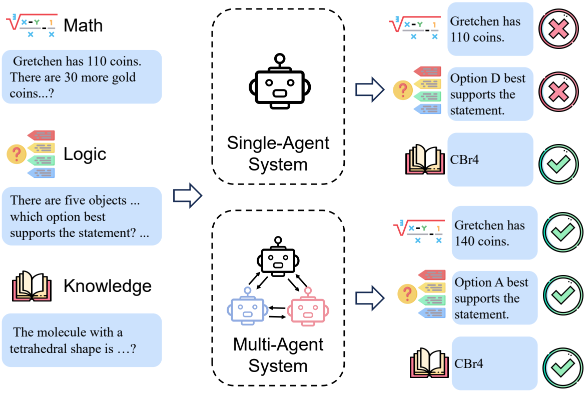

Math
Gretchen has 110 coins. There are 30 more gold coins...?
Logic
There are five objects ... which option best supports the statement? ...
Knowledge
The molecule with a tetrahedral shape is ...?
Single-Agent System
Multi-Agent System
Gretchen has 110 coins.
Option D best supports the statement.
CBr4
Gretchen has 140 coins.
Option A best supports the statement.
CBr4

Figure 1: Illustrations of single-agent system and multi-agent system.

In this paper, to comprehensively evaluate the multi-agent collaboration of large language models, we design a unified evaluation procedure and conducted systematic evaluations on 9 datasets across 3 tasks. Specifically, we design five collaboration architectures that reflect different communication patterns and the diversity of agent collaboration. This paper primarily investigates three research questions (RQ):

RQ1: Can multi-agent outperform single-agent systems? Different from single-agent systems, LLM-MAS involves multiple agents that influence each other with frequent and complex agent interactions. The interaction or communication patterns between agents, which we refer to as the collaboration architectures, can significantly affect the system performance. Some researchers have explored several possible optimal collaboration architectures (Chan et al., 2023; Chen et al., 2023b) and designed various LLM-based multi-agent systems. Yin et al. (2023) explored integrating different collaboration architectures to enhance system performance. However, past studies primarily focused on exploring specific systems, lacking a comprehensive study on the general properties of LLM-MAS. Inspired by traditional multi-agent theory, we construct several collaboration architectures and use these architectures to build multiple multi-agent systems and conduct systematic studies in different scenarios.

RQ2: Is scaling better for multi-agent systems? Cost is a crucial but often overlooked limiting factor in LLM-based multi-agent research. In this study, we analyze the scale of MAS, including time step, agent number, and the threshold of early stopping, etc. Yin et al. (2023) discussed the theoretical costs of some collaboration architectures. Li et al. (2024) systematically studied the effect of the agent number in a sampling-and-voting method. However, they did not consider the communication between agents. A question remains: How do we decide the scale or cost-related factors? Therefore, we systematically analyze the relationship between scale and performance in multi-agent.

RQ3: How to credit agents and optimize collaboration? We also address the optimization of collaboration architectures in LLMs-based MAS, which has received less attention than agent role assignments. Current strategies primarily utilize LLMs for evaluating agent outputs through ranking or rating (Liu et al., 2023b; Jiang et al., 2023b; Qin et al., 2023). This type of method, despite its prevalence, faces challenges in accuracy. In contrast, traditional multi-agent reinforcement learning (MARL) offers insights into collaboration through credit assignment, focusing on the distribution of rewards among agents based on their contributions. Inspired by MARL principles (Minsky, 1961; Sunehag et al., 2018), we explore an LLM-independent method using *Shapley value* to quantify the contributions of each collaboration in the system procedure and optimize multi-agent systems.

Our experiments provide insights into the multi-agent collaboration of large language models: 1) Multi-agent often outperform single-agent systems, and single-agent performance does not determine multi-agent benefit. 2) More agents will bring more benefits, and achieving agreement among agents is crucial for better performance. 3) By aggregating information and encouraging self-reflection among agents in the collaboration strategy, the optimized architecture system is efficient and effective.

Generally, our contributions are as follows:

- We introduce five collaborative multi-agent architectures and conduct extensive experiments in various scenarios to explore three crucial questions regarding the multi-agent collaboration of large language models.
- We investigate the connection between the scale and performance of LLM-MAS and provide an in-depth study of the agreement changes of the system and the early stopping mechanism.
- By quantifying the credits of individual agents, we propose a Shapley value-based optimization approach for LLM-MAS. This optimized structure significantly reduces communication costs across various datasets while achieving superior performance.

2 Collaboration Architectures

Traditional multi-agent research (Esmaeili et al., 2016; Damba and Watanabe, 2007; Dorri et al., 2018; Horling and Lesser, 2004) has identified and delineated various effective multi-agent architectures, including Flat, Hierarchical, Holonic, and Team. Each architecture possesses distinct advantages and is suitable for specific scenarios.

Inspired by the multi-agent theory and recent multi-agent research, we designed five unique collaboration architectures that reflect different communication patterns and the diversity of agent collaboration. Figure 2 contains five types of collaboration architectures. There are three static collaboration architectures: FULL, CYCLE, and HIERARCHICAL in Figure 2(a), and two dynamic architectures, TEAM and RANK, in Figure 2(b).

- **FULL** Inspired by the Flat structure (Dorri et al., 2018) of traditional multi-agent theory, information can be freely passed from one agent to another. In particular, when there are only two agents, the collaboration architecture degenerates into a typical debate architecture. This kind of architecture simulates information propagation in unrestricted discussions, facilitating the fast spread of information. However, such networks may lead to high costs.

- **Cycle** Inspired by Multi-Agent Debate (Liang et al., 2023), information is propagated among pairs of agents to reach a final agreement. This architecture simulates private conversation. It emphasizes how information gradually evolves and spreads over a limited number of interactions. This type of architecture has less costs, but the time required for the system to reach an agreement may be longer.

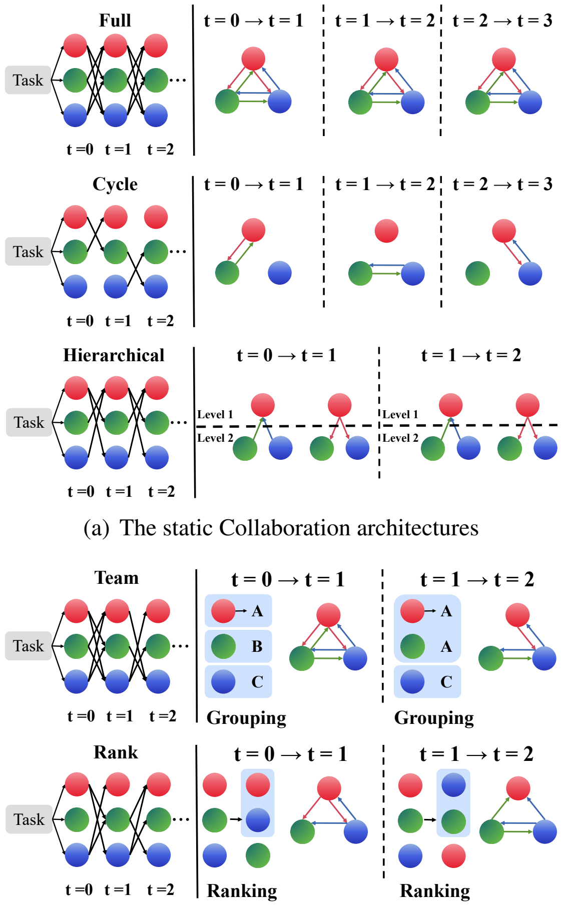
- **Hierarchical** Inspired by the Hierarchical structure (Damba and Watanabe, 2007) of the traditional multi-agent theory, information is propagated between nodes at different levels. This architecture simulates the Delphi method[1] in expert groups. This kind of architecture emphasizes aggregation and processing of the information.

- **Team** Inspired by the Team structure (Parker, 1993) of the traditional multi-agent, information flows between agents with different viewpoints(answers). This architecture simulates the propagation of information during a team discussion. This kind of architecture has no interaction between agents with the same viewpoint.

- **Rank** Inspired by the idea of agent optimization in DyLAN (Liu et al., 2023b), information and messages are sorted before it is delivered, and only top-k information can be passed to the next time step. This architecture simulates a review or screening process, such as editorial review or administrator approval, emphasizing the concern for information quality.

3 Experiments

This section introduces the dataset and LLM we used, providing a data foundation for subsequent problem analysis.



(a) The static Collaboration architectures

(b) The dynamic collaboration architectures

Figure 2: Collaboration Architectures

Tasks and Datasets. In our experiments, we used a general evaluation procedure to assess the performance of five architectures across three tasks, including: 1) Math: GSM8K (Cobbe et al., 2021), MATH (Hendrycks et al., 2021b), and SVAMP (Patel et al., 2021) datasets; 2) Knowledge: MMLU (Hendrycks et al., 2021a), CommonsenseQA (Talmor et al., 2019), and CommonsenseQA 2.0 (Talmor et al., 2022); 3) Logic: LogiQA (Liu et al., 2020), LogiQA2.0 (Liu et al., 2023a), and ReClor (Yu et al., 2020).

Model Details. We tested the proposed collaboration mechanism based on different models. Considering cost and effectiveness, we selected open-source models, e.g., Llama2-7b-Chat (Touvron et al., 2023), Mistral-7b-Instruction (Jiang et al., 2023a), and Starling-LM-7B-alpha (Zhu et al., 2023), for our experiments. Specifically, we downloaded the corresponding open-source models on hugging face and deployed the APIs using Fastchat and vLLM (Kwon et al., 2023). These three LLMs will be combined with the five archi-

[1] Delphi method: soliciting experts' opinions on a problem, organizing and summarizing them, then anonymously feeding them back to the experts, and soliciting opinions again until they reach an agreement

tectures to form 15 multi-agent systems. The maximum time step is six if not explicitly stated.

System Details. To reflect the difference between the Agents, we set the temperature of each Agent to a different value between 0 and 1 during generation. By default, we used 3 Agents with temperatures of 1, 0.6, and 0.4. Inspired by social comparison theory and review collaboration (Xu et al., 2023c), we considered generating solutions, final answers, and reviewing other agents' answers during generation. Complete prompt examples can be found in the appendix. Motivated by Liu et al. (2023b) and Practical Byzantine Fault Tolerance, when 2/3 of the agents in the system reached a consensus (i.e., the answer is the same), we made the system early stop, and the process stopped.

4 Can multi-agent systems outperform single-agent systems?

This section evaluates the multi-agent benefit. We conducted experiments with multi-agent systems composed of three LLMs and five collaboration architectures across nine datasets and analyzed the MAS performance according to the relative improvement of multi-agent systems. Moreover, we investigated the impact of the possible factors of multi-agent synergy, i.e., collaboration architecture, LLM, and task.

The benefit of MAS

Final Success Rate (e.g., accuracy) is the most commonly used metric for evaluating multi-agent systems (Du et al., 2023; Chan et al., 2023; Liu et al., 2023b; Chen et al., 2023a), which offers the advantages of simplicity and intuitiveness. However, the final success rate is highly correlated with the LLM and Task and does not reflect multi-agent synergy. To examine the benefits of multi-agent synergy, a natural idea is to consider the relative improvement in accuracy, which we refer to as accuracy improvement (Δ_{acc}).

$$\Delta_{\text{acc}} = \frac{\texttt{Perf}_m - \texttt{Perf}_s}{\texttt{Perf}_s} \qquad (1)$$

where $\texttt{Perf}_s$ and $\texttt{Perf}_m$ represent the performance (accuracy) of the single-agent[2] and system, respectively.

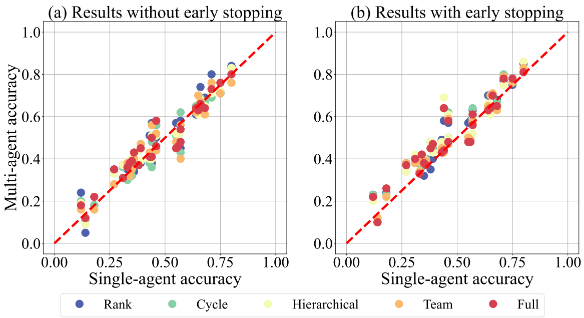


Figure 3: The scatter plot comparing multi-agent to single-agent performance.

Finding 1: *Multi-agent collaboration often help, and early stopping is necessary.* We plotted single-agent and multi-agent accuracy for all possible <Architecture, LLM, Task> triplets, totaling 135 points, as shown in Figure 3. A point above the red line indicates that the multi-agent system outperforms the single-agent. As the chart shows, 55.6% MAS showed improvement compared to single-agent. With early stopping activated, this number increased to 80%. This finding suggests that multi-agent approaches generally offer improvements, and early stopping mechanisms are crucial for maximizing system performance. Detailed data are given in the Appendix A.2.

Finding 2: *Every factor related to multi-agent synergy influences the system significantly, and single-agent performance does not determine multi-agent benefit.*

In this part, we investigated the effect of the three factors: architecture, LLM, and task. To study the effect of architecture, we formed a vector of performances for all five architectures in every possible <LLM, Task>. We averaged these vectors to indicate the relative performance of architectures. To minimize the influence of LLM and task, we performed z-score normalization or Min-max normalization on all vectors before averaging. Let $\texttt{Perf}(a, m, t)$ be the performance of a multi-agent system composed of architecture and LLM on task, $\tilde{\texttt{Perf}} = (\texttt{Perf}(\textsc{Full}), \texttt{Perf}(\textsc{Team}), ..., \texttt{Perf}(\textsc{Cycle}))$

$$\bar{\texttt{Perf}} = \frac{\sum_{(m,t)\in\mathcal{M}\times\mathcal{T}} \texttt{Norm}(\tilde{\texttt{Perf}}(m,t))}{|\mathcal{M}||\mathcal{T}|} \qquad (2)$$

The experiment results in Table 1 show that (1) different architectures led to different improvements, and the Rank architecture achieved rela-

[2] We use the results generated by greedy decoding to represent single-agent accuracy.

Metric		Single-Agent Accuracy		Multi-Agent Accuracy		Δ_{acc}	
Normalization		Min-Max	Z-score	Min-Max	Z-score	Min-Max	Z-score
Architecture	Full	\	\	0.41	-0.21	0.41	-0.21
	Cycle			0.51	0.10	0.51	0.10
	Hierarchical			0.52	0.04	0.52	0.04
	Rank			**0.55**	**0.18**	**0.55**	**0.18**
	Team			0.46	-0.11	0.46	-0.11
LLM	Llama2	0.12	-0.79	0.10	-0.97	0.60	0.30
	Mistral	0.31	-0.32	0.51	0.03	**0.66**	**0.45**
	Starling	**0.93**	**1.11**	**0.90**	**0.94**	0.15	-0.75
Task	Math	0.56	0.08	0.46	-0.10	**0.63**	**0.31**
	Knowledge	**0.77**	**0.57**	**0.84**	**0.76**	0.46	-0.10
	Logic	0.23	-0.65	0.22	-0.66	0.41	-0.20

Table 1: The analysis for the possible factors of multi-agent synergy. It is important to note that these values are **not the actual accuracy** of systems. They are the average values after Normalization across the different architectures (or LLMs, tasks).

tively the best results; (2) different LLMs led to different improvements, and Mistral achieved relatively the best results; (3) the effectiveness of the multi-agent approach also depended on the task. Math got the highest multi-agent benefits, which aligned with our expectations.

Notably, it is challenging to predict multi-agent benefits based on single-agent performance. For example, although Starling performed best with the single agent, its multi-agent benefits were less than Mistral. Knowledge tasks generally had the highest accuracy, but the multi-agent method improvement was less than Math. Besides, we plotted a scatter plot of single-agent performance and system improvement in the Appendix A.1, as shown in Figure 7, revealing no apparent correlation between $\texttt{Perf}_s$ and Δ_{acc}.

5 Is scaling better for multi-agent systems?

This section examines and analyzes the relationship between scale and performance in MAS. In particular, we considered the agent number and maximum communication rounds (time step) in MAS. Departing from Li et al. (2024), we focused on the scale of MAS with dynamic interactions among agents rather than the simple ensemble of answers.

Finding 3: *Many hands may make light work. More agents will bring more benefits.*

In this part, we explored the impact of different agent numbers in MAS. Due to max context length and expensive cost, we did not compare systems with more than five agents.

Dataset	MMLU		GSM8K		LOGIQA2	
Architecture	Full	Rank	Full	Rank	Full	Rank
1 agent	44.0%	44.0%	46.0%	46.0%	39.0%	39.0%
2 agents	55.0%	55.0%	45.0%	41.0%	39.0%	44.0%
3 agents	64.0%	57.0%	47.0%	50.0%	**47.0%**	40.0%
4 agents	**67.0%**	67.0%	51.0%	**57.0%**	44.0%	42.0%
5 agents	**67.0%**	**68.0%**	**52.0%**	51.0%	45.0%	**46.0%**

Table 2: The performance of systems with different agents. Every system here is conducted with Mistral and applied early stopping.

Table 2 shows the accuracy of different systems on different datasets. We observed an overall improvement in LLM-based agents, consistent with the findings of Li et al. (2024), which suggest that adding more agents can lead to better system performance. Although the performance did not continue to increase with five agents on the LOGIQA2 dataset, we believe that adding more and varied agents will improve its performance. It is worth pointing out that the only difference among the agents here is temperatures. Theoretically, adding agents with different roles or different LLMs will better improve performance (Chan et al., 2023; Liu et al., 2023b).

Finding 4: *Agreement is strength. Achieving agreement among agents is crucial for better performance.*

We calculated the system agreement at each time step and the proportion of the correct answer in each time step and shown the result in Figure 4. Generally, the higher system agreement could lead

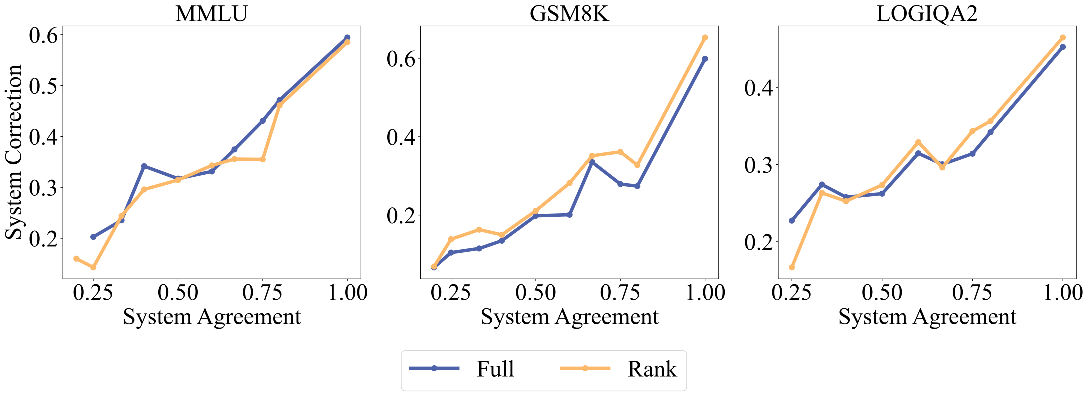


Figure 4: The agreement and system performance. This graph includes the results of 20 rounds of interaction on a dataset of 100 data points for systems ranging from 1 to 5 agents. The x-axis represents the system's agreement, specifically, the proportion of agents that reached a consensus (calculated by the proportion of the most voted answer to the total number of answers).

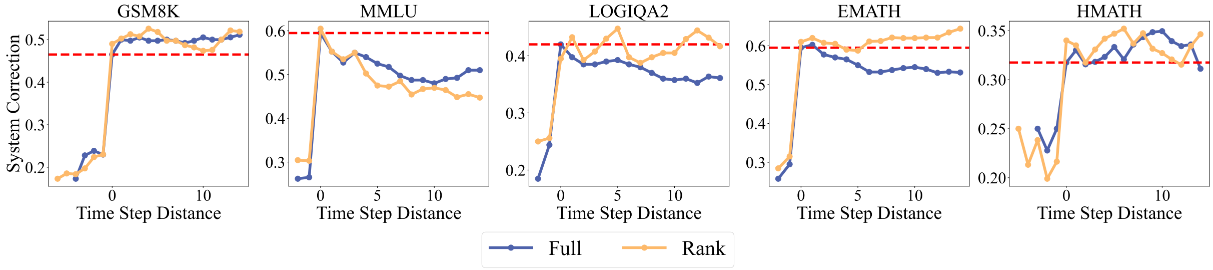


Figure 5: The agreement and system performance. The x-axis represents the distance from the current time step to the early stopping time step. For example, 0 represents the early stopping time step, 1 represents the next time step after early stopping, and -1 represents the time step before early stopping. These data come from a MAS composed of four agents based on Mistral.

to better system performance. This observation may indicate that the benefit of MAS comes from the procedure in which agents collaborate and ultimately reach a consensus. Additionally, we found that different datasets had different performance-increasing speeds. Therefore, we wondered if the agreement threshold for early stopping is unique for different datasets.

Considering that 95% of the data reached early stopping within ten time steps, we examined the ten time steps before and after reaching early stopping. As shown in Figure 5, we found that both MMLU and LogiQA2 reached their best performance at the early stopping time step. At the same time, GSM8K could further improve performance after early stopping, suggesting that using 2/3 as the early stopping threshold for GSM8K may not be reasonable. To determine the source of this observation, we additionally tested 100 sampled data of High school Mathematics Problems and Elementary Mathematics Problems in MMLU (named EMATH and HMATH), and the results revealed that EMATH showed a relatively small decrease with FULL and fluctuating correction with RANK, while HMATH showed a fluctuating increase in both architectures. We speculated the threshold might related to the task and its complexity. Math problems had a higher threshold, and the more challenging the tasks were, the higher the threshold was.

6 How to optimize collaboration?

This part demonstrates how to optimize a collaboration system. We focused on assessing the relative importance of the communication paths in each time step and optimizing the collaboration architectures. Specifically, we sampled another 200 data from GSM8K to optimize the FULL architecture time step by step.

Credit assignment in MAS

Recent LLM-MAS use LLMs to rank or rate the information output of agents, calculating contributions based on these rankings or scores. While

Architecture	Math			Knowledge			Logic			Avg↑	C_{rel}↓
	MATH	SVAMP	GSM8K	CSQA	CSQA2	MMLU	LogiQA	LogiQA2	ReClor		
OPTIMIZED (*Ours*)	**0.24**	**0.72**	**0.56**	**0.60**	**0.67**	**0.66**	0.37	**0.47**	0.41	**0.52**	**0.46**
FULL	0.22	0.70	0.47	0.58	0.64	0.64	0.38	**0.47**	0.43	0.50	1.00
RANK	0.19	0.71	0.49	0.57	0.66	0.65	**0.42**	0.45	**0.50**	**0.52**	0.66

Table 3: The performance of systems conducted with FULL, RANK, and OPTIMIZED architecture on different datasets. These systems were based on Mistral and built with 3 agents. C_{rel} indicates the relative number of communication paths, assume the path number of Full architecture to be 1.

this type of approach has achieved certain results in many related studies (Chan et al., 2023; Zhang et al., 2023b; Jiang et al., 2023b), ranking or rating text by LLMs remains an unsolved problem (Wang et al., 2023a; Shen et al., 2023). Inspired by Credit Assignment in MARL, we broke down the optimization of the collaboration architecture into identifying the relative importance and reward of each communication path between agents at any single time step.

We use the Shapley value (Shapley and Corporation, 1951) to indicate the relative importance. The Shapley value is a concept from cooperative game theory that offers a fair distribution of the total gains to the players (agents) based on their contributions to the alliance (MAS).

Suppose the set of communication paths to Agent n at time step t is S. We defined the value function $v(S)$ as the accuracy difference of Agent n[3] between time step t-1 and t. Given N paths, the formula for the Shapley value of path i is:

$$\sum_{S \subseteq N \setminus \{i\}} \frac{|S|!(|N|-|S|-1)!}{|N|!}(v(S \cup \{i\}) - v(S)) \tag{3}$$

$v(S \cup \{i\})$ is the value of the alliance contains path i and $v(S)$ is the value of the alliance without path i. A higher Shapley value suggests a more significant importance or contribution of this path.

After calculating the Shapley values of the communication paths, we removed those paths where Shapley values were lower than a certain threshold (we took the threshold as 0.002 to eliminate those paths with a small effect). This ensures an overall improvement of each time step. After optimizing a time step, we used the optimal structure to optimize the next time step, continuing this process until no positive reward path or reached the maximum time step.

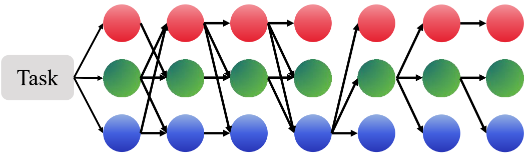


Figure 6: The optimized architecture. Contains only 46% communication paths in the Full architecture.

Finding 5: *The optimized architecture reduces cost and outperforms other architectures in many instances.*

We extracted 200 data points from the GSM8K training dataset and optimized the FULL architecture with 3 agents for 8 time steps. We applied these optimized architectures on all datasets, with the results shown in Table 3. To align with other architectures, we used only the first 6 time steps for evaluation. The optimized architectures performed well on the GSM8K and exhibited a certain degree of transfer ability on other datasets. Specifically, it outperforms FULL and RANK on 7 datasets. It is worth noting that we deleted those paths with smaller benefits during optimization, which further reduces the cost. The optimized architecture only contains 46.2% communication paths in the FULL architecture.

Finding 6: *The optimized architecture incorporates information aggregation and self-reflection.*

An interesting phenomenon occurs in optimized architecture: information aggregates to specific agents and then spreads back to all agents, consistent with the Hierarchical architecture. Furthermore, we found that in the optimized architecture, agents tend to communicate with others at early time steps and tend to make a self-reflection, which aligns with the method mentioned in Wang et al. (2023b), at the later time steps. This may reduce the propagation of misinformation after multiple rounds of interaction.

[3] we calculated the accuracy in the picked 200 training data

7 Related Work

LLM-based multi-agent. In the last few years, researchers have conducted numerous studies on LLM-MAS. Some studies focus on approaching collaborative mechanisms to enhance systems. These studies, e.g., Debate (Du et al., 2023), MAD (Liang et al., 2023), Deepwide (Zhang et al., 2023b), and ChatEval (Chan et al., 2023), concentrated on continuous debates among agents. Other studies focus on the decomposition of complex tasks, such as Camel (Li et al., 2023), ChatDev (Qian et al., 2023), AutoGen (Wu et al., 2023), and MetaGPT (Hong et al., 2023), exploring MAS for task division where different agents responsible for different sub-tasks. Additionally, a series of studies have explored how to use LLMs to simulate human behavior. This includes strategic and sandbox games like Werewolf (Xu et al., 2023a,b), Avalon (Lan et al., 2023), Minecraft (Chen et al., 2023b; Gong et al., 2023), game theory simulation (Fu et al., 2023; Mao et al., 2023; Guo et al., 2023), and sociological simulation (Park et al., 2023; Zhang et al., 2023a). However, the scale, agent credit, and factors related to multi-agent synergy have also not been comprehensively studied.

Collaboration Architecture of multi-agent. Traditional multi-agent research has proposed a variety of possible structures (Horling and Lesser, 2004) such as Flat, Hierarchical, Holonic (Esmaeili et al., 2016), Team, and Congregation (Brooks and Durfee, 2003). In the past few years, some studies have leveraged the capabilities of LLMs to construct more complex MAS. Shi et al. (2023); Du et al. (2023); Liang et al. (2023) organized multiple LLM-based agents for fixed rounds of debates. Chen et al. (2023a) organized agents in the form of a Round-Table Conference. ChatLLM (Hao et al., 2023) and WideDeep (Zhang et al., 2023b) organized agents into linear layers to enhance system capabilities. Zhang et al. (2023c) adopted a dynamic acyclic graph structure during the reasoning process. Liu et al. (2023b) proposed a dynamic architecture that can adjust according to different queries. Yin et al. (2023) proposed four architectures based on network topology.

Contribution of Agents. Evaluating the contribution of LLM agents is crucial for optimizing MAS. Credit assignment (Agogino and Tumer, 2004), introduced from traditional multi-agent, studies how to measure the impact of actions on global rewards. Extensive research has been delving into this problem, including implicit methods like policy gradients and Q-learning algorithms and explicit methods such as the Shapley value and actor-critic architecture. LLM-MAS studies primarily use extra LLMs for evaluation. Jiang et al. (2023b); Qin et al. (2023); Liu et al. (2023b) ranking outputs of agents to determine contributions. Others calculate contributions based on LLM's intermediate outcomes, such as the confidence evaluation proposed by (Yin et al., 2023), which calculates the model's confidence based on the variation in responses.

8 Conclusion and Future Direction

This paper focuses on three main questions: exploring the performance of multi-agent systems under various scenarios, investigating the influence of scale-related factors, crediting agents, and optimizing architectures. Our empirical study offers significant insights for collaboration within MAS, finding that single-agent performance does not decide the performance of multi-agent synergy. Furthermore, our study at scale suggests that adding more agents can lead to better system performance, aligning with the conclusions from (Li et al., 2024). We observed that the system agreement gradually increases as the time step increases. We also optimized the FULL architecture based on the Shapley value, which achieved the best results and demonstrated certain transferability. Our empirical study on scaling and crediting can be helpful in future studies of LLM-based multi-agent systems.

Limitations

Our study also has some limitations. First, we did not experiment with a MAS consisting of more than five agents due to the limitation of the context length of the open-source model. We plan to use models that support longer contexts for systems with more agents in the future. Besides, an interesting problem arises in Q2: Why does MAS show a performance decline after reaching early stopping on some datasets? According to our case study, this problem came from the accidentally generated error messages and the fast spreading of misinformation. We plan to analyze this phenomenon systematically in the future. Lastly, considering the extra computational costs of Shaley value, using Information Gain and a simplified method from MARL may be a better way.

References

Adrian K. Agogino and Kagan Tumer. 2004. Unifying temporal and structural credit assignment problems. In *3rd International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2004), 19-23 August 2004, New York, NY, USA*, pages 980–987. IEEE Computer Society.

Christopher H. Brooks and Edmund H. Durfee. 2003. Congregation formation in multiagent systems. *Auton. Agents Multi Agent Syst.*, 7(1-2):145–170.

Chi-Min Chan, Weize Chen, Yusheng Su, Jianxuan Yu, Wei Xue, Shanghang Zhang, Jie Fu, and Zhiyuan Liu. 2023. Chateval: Towards better llm-based evaluators through multi-agent debate.

Justin Chih-Yao Chen, Swarnadeep Saha, and Mohit Bansal. 2023a. Reconcile: Round-table conference improves reasoning via consensus among diverse llms.

Weize Chen, Yusheng Su, Jingwei Zuo, Cheng Yang, Chenfei Yuan, Chi-Min Chan, Heyang Yu, Yaxi Lu, Yi-Hsin Hung, Chen Qian, Yujia Qin, Xin Cong, Ruobing Xie, Zhiyuan Liu, Maosong Sun, and Jie Zhou. 2023b. Agentverse: Facilitating multi-agent collaboration and exploring emergent behaviors.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *ArXiv preprint*, abs/2110.14168.

Ariuna Damba and Shigeyoshi Watanabe. 2007. Hierarchical control in a multiagent system. In *Second International Conference on Innovative Computing, Informatio and Control (ICICIC 2007)*, pages 111–111. IEEE.

Ali Dorri, Salil S. Kanhere, and Raja Jurdak. 2018. Multi-agent systems: A survey. *IEEE Access*, 6:28573–28593.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B. Tenenbaum, and Igor Mordatch. 2023. Improving factuality and reasoning in language models through multiagent debate. *ArXiv preprint*, abs/2305.14325.

Ahmad Esmaeili, Nasser Mozayani, Mohammad Reza Jahed-Motlagh, and Eric T. Matson. 2016. The impact of diversity on performance of holonic multi-agent systems. *Eng. Appl. Artif. Intell.*, 55:186–201.

Yao Fu, Hao Peng, Tushar Khot, and Mirella Lapata. 2023. Improving language model negotiation with self-play and in-context learning from AI feedback. *ArXiv preprint*, abs/2305.10142.

Ran Gong, Qiuyuan Huang, Xiaojian Ma, Hoi Vo, Zane Durante, Yusuke Noda, Zilong Zheng, Song-Chun Zhu, Demetri Terzopoulos, Li Fei-Fei, and Jianfeng Gao. 2023. Mindagent: Emergent gaming interaction. *ArXiv preprint*, abs/2309.09971.

Jiaxian Guo, Bo Yang, Paul Yoo, Bill Yuchen Lin, Yusuke Iwasawa, and Yutaka Matsuo. 2023. Suspicion-agent: Playing imperfect information games with theory of mind aware GPT-4. *ArXiv preprint*, abs/2309.17277.

Taicheng Guo, Xiuying Chen, Yaqi Wang, Ruidi Chang, Shichao Pei, Nitesh V. Chawla, Olaf Wiest, and Xiangliang Zhang. 2024. Large language model based multi-agents: A survey of progress and challenges. *ArXiv preprint*, abs/2402.01680.

Rui Hao, Linmei Hu, Weijian Qi, Qingliu Wu, Yirui Zhang, and Liqiang Nie. 2023. Chatllm network: More brains, more intelligence. *ArXiv preprint*, abs/2304.12998.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021a. Measuring massive multitask language understanding. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021b. Measuring mathematical problem solving with the MATH dataset. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual*.

Sirui Hong, Xiawu Zheng, Jonathan Chen, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, Lingfeng Xiao, and Chenglin Wu. 2023. Metagpt: Meta programming for multi-agent collaborative framework. *ArXiv preprint*, abs/2308.00352.

Bryan Horling and Victor R. Lesser. 2004. A survey of multi-agent organizational paradigms. *Knowl. Eng. Rev.*, 19(4):281–316.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023a. Mistral 7b.

Dongfu Jiang, Xiang Ren, and Bill Yuchen Lin. 2023b. LLM-blender: Ensembling large language models with pairwise ranking and generative fusion. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14165–14178, Toronto, Canada. Association for Computational Linguistics.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the*

ACM SIGOPS 29th Symposium on Operating Systems Principles.

Yihuai Lan, Zhiqiang Hu, Lei Wang, Yang Wang, Deheng Ye, Peilin Zhao, Ee-Peng Lim, Hui Xiong, and Hao Wang. 2023. Llm-based agent society investigation: Collaboration and confrontation in avalon gameplay. *ArXiv preprint*, abs/2310.14985.

Guohao Li, Hasan Abed Al Kader Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. 2023. CAMEL: communicative agents for "mind" exploration of large scale language model society. *ArXiv preprint*, abs/2303.17760.

Junyou Li, Qin Zhang, Yangbin Yu, Qiang Fu, and Deheng Ye. 2024. More agents is all you need.

Tian Liang, Zhiwei He, Wenxiang Jiao, Xing Wang, Yan Wang, Rui Wang, Yujiu Yang, Zhaopeng Tu, and Shuming Shi. 2023. Encouraging divergent thinking in large language models through multi-agent debate. *CoRR*, abs/2305.19118.

Hanmeng Liu, Jian Liu, Leyang Cui, Zhiyang Teng, Nan Duan, Ming Zhou, and Yue Zhang. 2023a. Logiqa 2.0 - an improved dataset for logical reasoning in natural language understanding. *IEEE ACM Trans. Audio Speech Lang. Process.*, 31:2947–2962.

Jian Liu, Leyang Cui, Hanmeng Liu, Dandan Huang, Yile Wang, and Yue Zhang. 2020. Logiqa: A challenge dataset for machine reading comprehension with logical reasoning. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 3622–3628. ijcai.org.

Zijun Liu, Yanzhe Zhang, Peng Li, Yang Liu, and Diyi Yang. 2023b. Dynamic llm-agent network: An llm-agent collaboration framework with agent team optimization.

Shaoguang Mao, Yuzhe Cai, Yan Xia, Wenshan Wu, Xun Wang, Fengyi Wang, Tao Ge, and Furu Wei. 2023. ALYMPICS: language agents meet game theory. *ArXiv preprint*, abs/2311.03220.

Marvin Minsky. 1961. Steps toward artificial intelligence. *Proceedings of the IRE*, 49(1):8–30.

Joon Sung Park, Joseph C. O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. 2023. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology, UIST 2023, San Francisco, CA, USA, 29 October 2023- 1 November 2023*, pages 2:1–2:22. ACM.

Lynne E. Parker. 1993. Designing control laws for cooperative agent teams. In *Proceedings of the 1993 IEEE International Conference on Robotics and Automation, Atlanta, Georgia, USA, May 1993*, pages 582–587. IEEE Computer Society Press.

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, Online. Association for Computational Linguistics.

Chen Qian, Xin Cong, Cheng Yang, Weize Chen, Yusheng Su, Juyuan Xu, Zhiyuan Liu, and Maosong Sun. 2023. Communicative agents for software development. *ArXiv preprint*, abs/2307.07924.

Zhen Qin, Rolf Jagerman, Kai Hui, Honglei Zhuang, Junru Wu, Jiaming Shen, Tianqi Liu, Jialu Liu, Donald Metzler, Xuanhui Wang, and Michael Bendersky. 2023. Large language models are effective text rankers with pairwise ranking prompting. *ArXiv preprint*, abs/2306.17563.

L.S. Shapley and Rand Corporation. 1951. *Notes on the N-person Game*. Notes on the N-person Game. Rand Corporation.

Chenhui Shen, Liying Cheng, Xuan-Phi Nguyen, Yang You, and Lidong Bing. 2023. Large language models are not yet human-level evaluators for abstractive summarization. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 4215–4233, Singapore. Association for Computational Linguistics.

Zijing Shi, Meng Fang, Shunfeng Zheng, Shilong Deng, Ling Chen, and Yali Du. 2023. Cooperation on the fly: Exploring language agents for ad hoc teamwork in the avalon game. *ArXiv preprint*, abs/2312.17515.

Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinicius Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z Leibo, Karl Tuyls, et al. 2018. Value-decomposition networks for cooperative multi-agent learning based on team reward. In *Proceedings of the 17th International Conference on Autonomous Agents and Multi-Agent Systems*, pages 2085–2087.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158, Minneapolis, Minnesota. Association for Computational Linguistics.

Alon Talmor, Ori Yoran, Ronan Le Bras, Chandra Bhagavatula, Yoav Goldberg, Yejin Choi, and Jonathan Berant. 2022. Commonsenseqa 2.0: Exposing the limits of AI through gamification. *CoRR*, abs/2201.05320.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti

Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models.

Peiyi Wang, Lei Li, Liang Chen, Zefan Cai, Dawei Zhu, Binghuai Lin, Yunbo Cao, Qi Liu, Tianyu Liu, and Zhifang Sui. 2023a. Large language models are not fair evaluators.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023b. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, Ahmed Hassan Awadallah, Ryen W White, Doug Burger, and Chi Wang. 2023. Autogen: Enabling next-gen llm applications via multi-agent conversation.

Yuzhuang Xu, Shuo Wang, Peng Li, Fuwen Luo, Xiaolong Wang, Weidong Liu, and Yang Liu. 2023a. Exploring large language models for communication games: An empirical study on werewolf. *ArXiv preprint*, abs/2309.04658.

Zelai Xu, Chao Yu, Fei Fang, Yu Wang, and Yi Wu. 2023b. Language agents with reinforcement learning for strategic play in the werewolf game. *ArXiv preprint*, abs/2310.18940.

Zhenran Xu, Senbao Shi, Baotian Hu, Jindi Yu, Dongfang Li, Min Zhang, and Yuxiang Wu. 2023c. Towards reasoning in large language models via multi-agent peer review collaboration.

Zhangyue Yin, Qiushi Sun, Cheng Chang, Qipeng Guo, Junqi Dai, Xuanjing Huang, and Xipeng Qiu. 2023. Exchange-of-thought: Enhancing large language model capabilities through cross-model communication. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 15135–15153, Singapore. Association for Computational Linguistics.

Weihao Yu, Zihang Jiang, Yanfei Dong, and Jiashi Feng. 2020. Reclor: A reading comprehension dataset requiring logical reasoning. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Jintian Zhang, Xin Xu, and Shumin Deng. 2023a. Exploring collaboration mechanisms for llm agents: A social psychology view.

Xinghua Zhang, Bowen Yu, Haiyang Yu, Yangyu Lv, Tingwen Liu, Fei Huang, Hongbo Xu, and Yongbin Li. 2023b. Wider and deeper llm networks are fairer llm evaluators.

Yifan Zhang, Jingqin Yang, Yang Yuan, and Andrew Chi-Chih Yao. 2023c. Cumulative reasoning with large language models. *ArXiv preprint*, abs/2308.04371.

Banghua Zhu, Evan Frick, Tianhao Wu, Hanlin Zhu, and Jiantao Jiao. 2023. Starling-7b: Improving llm helpfulness harmlessness with rlaif.

A Appendix

A.1 The relation between single-agent performance and multi-agent benefit

In section 4, we propose the finding that single-agent performance does not determine multi-agent benefit. To further verify this finding, we made a scatter plot of single-agent system's accuracy with multi-agent benefit, as shown in Fig. 7. It can be found that there is no obvious correlation between them, which supports the conclusion of section 4.

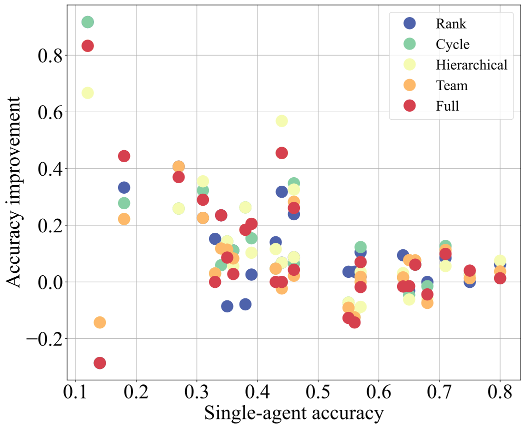


Figure 7: Single-agent accuracy and system improvement

A.2 System performance in every architecture, LLM, and dataset

In section 4, we calculated the average influence of different factors, i.e., architecture, LLM, and dataset, but the absolute performance of each factor was not shown. For this reason, we present all data in Table 4. Keep in mind that the table only contains results for the 3-agent system, considering the cost, we did not conduct such extensive experiments for systems consisting of more agents.

A.3 Shapley value of every path

In section 6, we optimized FULL architecture with Shapley value, but we didn't present the middle value of the optimization. Here, we show the Shapley value of every path in each optimization time step in Table 5. Noticing that each column depend on the optimized architecture at that time step.

A.4 Agent prompt

We show the role prompt for each agent in Table 6.

LLM	Architecture	MATH	GSM8K	SVAMP	CSQA	CSQA2	MMLU	LogicQA	LogiQA2	ReClor
llama2	Single Agent	0.14	0.18	0.57	0.31	0.57	0.44	0.38	0.33	0.27
	Full	0.10	0.26	0.61	0.40	0.56	0.44	0.45	0.33	0.37
	Cycle	0.10	0.23	0.64	0.41	0.61	0.44	0.48	0.34	0.34
	Hierarchical	0.12	0.22	0.59	0.42	0.52	0.47	0.48	0.34	0.34
	Team	0.12	0.22	0.58	0.38	0.57	0.43	0.45	0.34	0.38
	Rank	0.10	0.24	0.58	0.38	0.63	0.47	0.35	0.38	0.38
mistral	Single Agent	0.12	0.46	0.66	0.46	0.65	0.44	0.35	0.39	0.43
	Full	0.22	0.48	0.70	0.58	0.64	0.64	0.38	0.47	0.43
	Cycle	0.23	0.49	0.70	0.62	0.62	0.64	0.40	0.45	0.45
	Hierarchical	0.20	0.50	0.71	0.61	0.61	0.69	0.40	0.43	0.48
	Team	0.22	0.47	0.71	0.59	0.70	0.64	0.39	0.47	0.45
	Rank	0.23	0.50	0.70	0.57	0.63	0.58	0.32	0.40	0.49
starling	Single Agent	0.34	0.75	0.80	0.71	0.68	0.64	0.36	0.55	0.56
	Full	0.42	0.78	0.81	0.78	0.65	0.63	0.37	0.48	0.48
	Cycle	0.36	0.77	0.81	0.80	0.67	0.65	0.40	0.51	0.49
	Hierarchical	0.38	0.77	0.86	0.75	0.64	0.66	0.38	0.51	0.50
	Team	0.38	0.76	0.83	0.79	0.63	0.65	0.39	0.50	0.49
	Rank	0.42	0.75	0.85	0.77	0.68	0.70	0.40	0.57	0.58

Table 4: System accuracy on every system and dataset. Systems based on 3 agents. The max time step is 6.

Time Step	1	2	3	4	5	6	7	8
$Path(0,0)$	-0.043	0.009	0.006	-0.001	-0.009	0.008	0.010	-0.015
$Path(1,0)$	0.033	0.002	0.001	-0.008	0.018	-0.003	0.000	0.000
$Path(2,0)$	0.035	-0.031	-0.022	0.009	-0.004	-0.005	-0.005	0.000
$Path(0,1)$	0.013	0.016	0.007	-0.003	-0.032	-0.009	-0.006	-0.013
$Path(1,1)$	0.003	0.016	0.014	-0.005	0.021	0.016	-0.006	0.004
$Path(2,1)$	0.000	-0.017	-0.006	0.013	-0.004	-0.007	0.002	-0.011
$Path(0,2)$	-0.008	0.025	0.019	0.002	-0.008	-0.008	0.010	-0.001
$Path(1,2)$	0.015	0.010	0.014	-0.016	0.012	0.017	-0.020	-0.003
$Path(2,2)$	0.008	-0.020	-0.033	0.014	-0.008	-0.003	0.000	0.004

Table 5: The Shapley value for every path in every time step during optimizing. the $Path(i,j)$ denote the path from agent i to agent j

```
[System Prompt]
You are an excellent and very capable domain question solver. You are now invited to an expert group of processing and solving domain application questions. Your codename in the expert group is Expert self.rrid. As a distinguished member of the expert group, you possess the capability to a broad range of domain disciplines, allowing you to adapt and apply the appropriate methodologies to the given questions.

[User Prompt]
### Task Description
Your task is to systematically address the domain application question presented below, decipher complex question statements and elucidate your reasoning in a sequential, step-by-step fashion. Carefully utilize the provided information to work through the question. Your answer should be both concise and comprehensive, detailing the logical progression of your thought process. Besides, the expert group have provided some potential answers to this question, you should consider insights from these answers to enrich the quality and accuracy of your own answer.

### Given Question
Question: question

### Given Question Again
Read the given question again.
Question: question

### Answers by Other Experts
There are some potential answers provided by different experts for the same question. Consider these responses to cross-verify your approach, broaden your understanding, and gain alternative perspectives with diverse approaches to the question-solving process. This may help you ensure consistency and accuracy in your methodology. However, we have not verified the correctness of these answers, so be careful of the quality and relevance of these answers.
messages

### Output Format
start
Opinion: your opinion about other experts' answers
Solution: your detailed, step-by-step solution, final answer is formatted as "[ final answer here ]"
end

The output start with your opinion about other experts' answers, followed by your step-by-step solution in the next line.
Remember that your final answer in the solution is surrounded by '[' and ']', which is formatted as "[ final answer here ]".
Now take a deep breath and solve the question step by step.
```

Table 6: The prompt template for Agent. We replace the colored slot with real text before querying the LLMs. Note that we use a similar template when conducting single-agent-based experiments and ignore the Answers by Other Experts.