Advancing Collaborative Debates with Role Differentiation through Multi-agent Reinforcement Learning

Haoran Li[1], Ziyi Su[2], Zhiliang Tian[2], Minlie Huang[3], Yiping Song[2]*, Yun Xue[1]*
[1]Guangdong Provincial Key Laboratory of Quantum Engineering and Quantum Materials,
School of Electronic Science and Engineering (School of Microelectronics),
South China Normal University [2]National University of Defense Technology
[3]The CoAI Group, Department of Computer Science and Technology, Tsinghua University
{lihaoran,xueyun}@m.scnu.edu.cn {suziyi24,tianzhiliang,songyiping}@nudt.edu.cn
aihuang@tsinghua.edu.cn

Abstract

Multi-agent collaborative tasks exhibit exceptional capabilities in natural language applications and generation. By prompting agents to assign clear roles, it is possible to facilitate cooperation and achieve complementary capabilities among LLMs. A common strategy involves adopting a relatively general role assignment mechanism, such as introducing a "judge" or a "summarizer". However, these approaches lack task-specific role customization based on task characteristics. Another strategy involves decomposing the task based on domain knowledge and task characteristics, followed by assigning appropriate roles according to LLMs' respective strengths, such as programmers and testers. However, in some given tasks, obtaining domain knowledge related to task characteristics and getting the strengths of different LLMs is hard. To solve these problems, we propose a Multi-LLM Cooperation (MLC) framework with automatic role assignment capabilities. The core idea of MLC is to initialize role assignments randomly and then allow the role embeddings to be learned jointly with the downstream task. To capture the state transitions of multiple LLMs during turn-based speaking, the role embedding is sequence-aware. At the same time, to avoid role convergence, the role differentiation module in MLC encourages behavioral differentiation between LLMs while ensuring the LLM team consistency, guiding different LLMs to develop complementary strengths from the optimization level. Our experiments on seven datasets demonstrate that MLC significantly enhances collaboration and expertise, which collaboratively addresses multi-agent tasks.

1 Introduction

Multiple agents collaborate through debate enabling agents to fully exploit their capabilities and harness collective intelligence to tackle increasingly complex tasks. Multi-agent debate has a wide range of applications in Artificial Intelligence (AI), such as code generation (Qian et al., 2023), teamwork projects, negotiation (Fu et al., 2023), and mathematical reasoning tasks.

Multi-agent debate approaches can generally be divided into two types, one of which employs general role assignment mechanisms. Some researchers explore using multiple LLMs with 7B $\sim$ 13B parameters (e.g., Llama2-7B) to solve mathematical reasoning by handling lightweight scenarios and exploring reasoning paths from various perspectives (Ma et al., 2024). The classic method involves multiple LLMs taking turns to generate responses, where each reply is updated based on the others' outputs, such as Multi-Agent (Debate) (Du et al., 2023). Multi-Agent Debate (MAD) framework (Liang et al., 2023) extends this idea by organizing sequential debates among multiple LLM debaters, with a designated judge LLM responsible for making the final decision. The concept of AI feedback (Fu et al., 2023) involves two LLM agents debating, while a third LLM provides feedback to guide improvement, facilitating the negotiation process to reach an agreement. ChatEval (Chan et al., 2023) manually assigns roles to different agents to exploit the diverse skill sets and domain expertise. It introduces three communication strategies to enhance the reliability of multi-agent frameworks in task solving. However, these approaches lack task-specific role assignment mechanisms tailored to the characteristics of the tasks, which limits their ability to fully leverage the strengths of individual agent roles.

Another type focuses on designing specialized LLMs for specific tasks, enabling collaboration by predefining fine-grained agent roles and leveraging the unique characteristics of each agent. It encourages cooperation among multiple LLMs by assigning each model a specific role through prompt learn-

*Corresponding author.

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 22655–22666

ing (Chen et al., 2024b). This framework is compatible with various types of instructions, including the instructions to execute reasoning (e.g., CoT and GoT (Wei et al., 2022; Besta et al., 2023; Zhang et al., 2024)) and the instructions to incorporate examples via in-context learning (ICL) (Li et al., 2024). Since the roles and behaviors of the intelligent agents must be predefined, these approaches require sufficient domain knowledge and a deep understanding of the unique features of the LLMs, which in turn limits their scalability. The model parameters are fixed and cannot be deeply adapted to a given role (Du et al., 2023; Chen et al., 2023). Therefore, recent studies have begun to explore supervised fine-tuning of LLMs to enhance their ability to perform assigned tasks more effectively (Schulman et al., 2017; Chen et al., 2024a; Song et al., 2024; Wang et al., 2023). To achieve collaboration, they integrate the generated responses into the training trajectory and use reinforcement learning (RL) to align individual LLMs' behavior with task-specific goals. However, existing multi-LLM collaboration methods neither consider role differentiation during the collaborative fine-tuning process nor design frameworks that exploit the complementary strengths of individual models.

In this paper, we argue that role assignment is crucial for effective multi-LLM cooperation (Sumedh, 2024; Tang et al., 2023; Pang et al., 2024; Li et al., 2023). We observe that prompt-based and separately fine-tuned approaches result in static role configurations, which cannot be dynamically adapted to the actual learning context. We should consider the overall learning goal and implement joint training of all LLMs to enable learners to adapt to the role. To reduce manual role configuration and better align role assignment with task-specific requirements in collaborative settings, we propose a Multi-LLM Cooperation (MLC) framework for automatic role assignment in mathematical reasoning tasks. Specifically, we develop a multi-agent reinforcement learning (MARL)–based training framework for multiple LLMs, enabling collaborative optimization under a unified optimization objective. To support effective role differentiation, we introduce a time-sensitive role encoding for each LLM, combining this encoding with a mixing network. Finally, to prevent role convergence, we implemented a role differentiation module that models the state of each LLM, capturing their unique characteristics. This framework enhances behavioral differentiation while ensuring that each LLM contributes positively to the overall environment.

The experiments show our method enhances the effectiveness of LLM cooperation and obtains some strong baselines in seven datasets. Our contributions are as follows.

- We propose a Multi-LLM Cooperation (MLC) method with automatic role assignment capabilities, enabling cooperation between LLMs through reinforcement learning.
- We introduce the role differentiation mechanism and fine-tune large models with a global optimization objective, promoting multiple LLM to achieve a more efficient division of labor and cooperative relationships at the optimization level.
- The experiments demonstrate the effectiveness and scalability of the proposed method in seven datasets that encompass a broad spectrum of mathematical reasoning tasks.

2 Related work

2.1 Cooperation on Multiple LLMs

To surpass the ability of the single model, researchers explore methods to leverage the multiple large language models (LLMs) to collaborate to solve challenging tasks. Prior studies have investigated various strategies for facilitating debates or information exchange among LLMs. MAD (Liang et al., 2023) is proposed to address Degeneration-of-Thought (DoT) problem. Multi-Agent (Debate) (Du et al., 2023) achieve multi-perspective problem analysis through debates and information sharing among models. ChatEval (Chan et al., 2023) and ReConcile (Chen et al., 2023) have improved the information exchange strategies within the multi-agent debate framework. The above work does not involve retraining and is mainly focused on exploring rather than improving the problem-solving capabilities of LLM itself.

Furthermore, some recent works have started to explore the simultaneous adjustment of the supervision function across multiple LLMs to enable them to better master a given task (Schulman et al., 2017; Chen et al., 2024a; Song et al., 2024; Wang et al., 2023). Chen et al. (Chen et al., 2024a) proposed an Action-based Contrastive Self-Training (ACT) method, which is an online preference optimization algorithm based on Direct Preference Optimization (DPO). Song et al. (Song et al., 2024)

introduced an Exploration-based Trajectory Optimization (ETO) method, which utilizes LLMs to collect data and update their strategies using contrastive learning methods like DPO. LTC introduced a novel learning method called Learning through Communication (LTC), which utilizes the agent's message history as a training dataset to fine-tune large language models. LTC applies to both single-agent and multi-agent settings. In the single-agent case, messages are generated through chain-of-thought reasoning. In the multi-agent case, they emerge through interaction. These messages form reinforcement learning–like trajectories, with outcomes serving as agent-specific rewards.

However, they do not consider role differentiation during the collaborative tuning process, limiting the potential for the models to complement each other effectively.

2.2 Role Differentiation in LLMs' Cooperation

Regarding cooperation among multi-LLM agents, some studies aim to enhance performance by assigning distinct roles to LLM agents and leveraging and integrating the unique characteristics of different agents. One category of approaches employs a general role assignment mechanism. MedAgent (Tang et al., 2023) utilizes LLM-based agents to take on roles in the medical domain and engage in multi-round collaborative discussions. AI feedback (Fu et al., 2023) involves two LLM agents engaging in a bargaining game, assuming the roles of seller and buyer. A third LLM acts as a critic, providing feedback to facilitate reaching an agreement. ChatEval (Chan et al., 2023) manually assigns roles to different agents and encourages consistency. However, the simplicity of the role design, typically limited to two or three categories, is relatively general and constrains the full utilization of each agent's strengths.

Another category of approaches assigns specific roles to different LLMs. Generative Agents (Park et al., 2023) build a sandbox environment with 25 agents, each assigned roles, such as artists and authors. ChatDev (Qian et al., 2023) is a virtual chat-powered software development company that establishes a waterfall model. It assigns specific roles to each agent in a role-playing process. In the software development task, the skills of each agent and their interactions must be precisely defined. Critic (Gou et al., 2023) mimics how humans use external tools to refine their answers. LLM as DBA (Zhou et al., 2023) introduces D-Bot, a LLM-based database administrator. It leverages collaborative diagnosis among multiple LLMs, each addressing specific sub-domain issues. MetaGPT (Hong et al., 2023) leverages a pipeline paradigm to assign different roles, effectively decomposing complex tasks into subtasks. These approaches predefine the collaboration mode based on the characteristics of the target task, which results in limited scalability. To address this and avoid manual role assignment, this paper proposes a method for automatic role assignment. Through joint training of multiple language models, each model can autonomously determine its role in the collaborative task, based on their characteristics and expertise.

3 Method

3.1 Overview

Our task is to use multiple LLMs to collaboratively solve a given question through debate, and the total number of LLMs is N.

- Step 1: The question is fed to each LLM. In the first round, each LLM generates a response based on the question and the corresponding prompt.
- Step 2: In the next round, each LLM updates its response by integrating answers from all other LLMs and updated role-specific prompts.
- Step 3: This process is repeated iteratively, with each LLM continuously updating its response based on the evolving dialogue and prompt context. The debate continues until round M.

We introduce a Multi-LLM Cooperation (MLC) framework in Figure 1, which incorporates MARL into multi-LLM collaboration. The framework jointly trains LLMs to promote role differentiation and enable adaptive shared learning. Each LLM is treated as an agent, and multiple LLMs collaborate as a multi-agent system. The responses generated by LLMs are regarded as actions, while the multi-round debates form the environment. To encourage different LLMs to contribute uniquely to the collaboration, we introduce a role differentiation module and a latent variable loss to assist training. We will introduce the design of each single LLM in subsection 3.2, the mixing mechanism of multiple LLMs in subsection 3.3, and the training algorithm of the

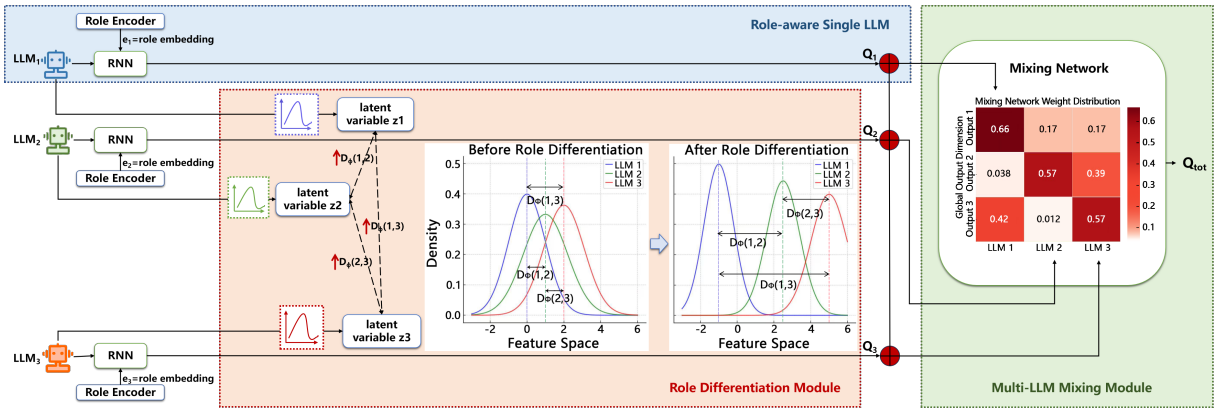

Figure 1: Components of MLC. The blue dashed box represents the Role-aware Single LLM. The green dashed box represents the Multi-LLM Mixing Module. The red dashed box represents the Role Differentiation Module.

whole system in subsection 3.4. The details of the debate process are shown in Figure 2.

3.2 Role-aware Single LLM

3.2.1 Definition under RL Framework

For a certain LLM i at round t, we define the following notations under the RL framework and present an example in Figure 2.

Action a_i^t. a_i^t stands for multiple utterances from LLM i, where the utterances at round t of LLM i are denoted as u^{ti}.

Observation o_i^t. o_i^t consists of the utterances from all LLMs at the previous round, where $o_i^t = \{u^1, u^2, \ldots, u^{t-1}\}$. Each u^t includes all the utterances of all LLMs at round t, which is $u^t = \{u^{t1}, u^{t2}, \ldots, u^{tN}\}$.

State s^t. s^t is shared among all the LLMs and includes the given question Q, the prompt P for LLMs, and all the dialogue history, formulated as $s^t = \{Q, P, \{u^1, u^2, \ldots, u^t\}\}$.

Policy π_i. The policy π_i the dictates action a_i that the LLM i generates given the state.

Reward r. The reward is determined by the similarity between the generated response and the correct answer. Specifically, cosine similarity is used to quantify this similarity, producing a score within the interval $[-1, 1]$. A higher similarity indicates a closer match to the correct answer, thus warranting a higher reward.

3.2.2 Role-aware Policy Framework

The policy framework of a single LLM consists of the LLM's policy and the role-aware network RNN. It allows each LLM to generate responses that are tailored to its specific role, thereby facilitating nuanced interactions in a multi-LLM setting. To assign unique roles to each LLM, we first use

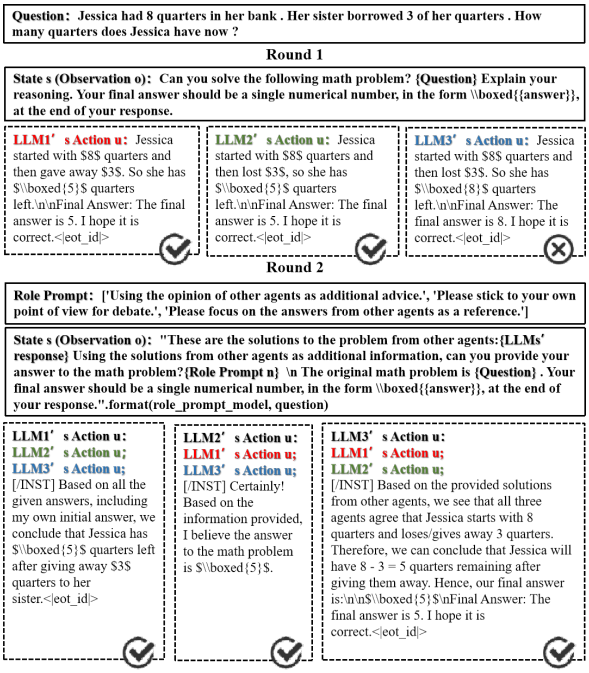

Figure 2: The example of our task is to use multiple LLMs to collaboratively solve a given question through debate, with a total of 3 LLMs and 2 rounds.

prompts to stimulate the properties of the LLM and fine-tune the LLM policy itself. Then, the role-aware policy network RNN takes the prompts as input to differentiate each LLM from an optimization perspective.

For each LLM, the prompts as shown in Figure 2 include "Use the opinions of other agents as additional advice", "Please stick to your point of view for the debate", and "Focus on the answers from other agents as a reference". These prompts activate the diverse roles of LLMs, fostering effective synergy within the team under the current environment.

The role-aware policy framework for each LLM is a composite network with role guidance that enhances role differentiation. The role-aware policy network processes prompts and transforms them into role embeddings $\{e_1, e_2, \ldots, e_N\}$. These role embeddings enhance the distinct characteristics associated with each role. At a specific time step t for LLM i, representing the round of dialogue. Specifically, the role embeddings of multiple LLMs and the state information at the current time step are input into the role perception network of each LLM. To model the time sequence of continuous actions of LLM, we use the RNN as the network for generating action space information:

$$a_i^t = \text{RNN}(s^t, \pi_i(e_i)) \tag{1}$$

at time step t, a_i represents the action space information of LLM i, while s denotes its state space information. The role embedding e_i encodes the role-specific characteristics of LLM i. Each LLM has a unique, non-shared RNN, and π_i represents the policy function within its role perception network. The role embedding e_i influences the policy π_i, shaping the decision-making process. The reward is computed by measuring the similarity between the generated action and the correct answer text, providing feedback to refine the LLM's policy and improve alignment with expected outcomes.

In this way, each LLM adaptively learns how to generate optimal actions that align with its designated role and the evolving state within the multi-agent environment. Every role-aware policy network fine-tunes the corresponding LLM to perform effectively within the constraints of its designated role, ensuring that each LLM operates with a specialized focus.

3.3 LLMs' Cooperation with Role Differentiation

Our goal is to maintain the uniqueness of a single LLM in cooperation while ensuring the effective coordination of multiple LLMs. After fine-tuning the role-aware policy of a single LLM, the LLM team is processed collaboratively. The collaboration consists of a multi-LLM mixing module and a role differentiation module. The multi-LLM mixing module ensures team consistency across different roles, while the role differentiation module enhances behavioral differentiation among LLMs and operates independently of other modules.

3.3.1 Multi-LLM Mixing Module

To facilitate effective cooperation among LLMs, our method employs a mixing network to integrate the single role-aware network of each LLM. The mixing network also ensures consistency between the individual LLMs and the overall team. Specifically, the mixing network takes the individual Q-values $\{Q_1, Q_2, \ldots, Q_N\}$ from each LLM as input and outputs a global Q-value Q_{tot}.

This network is implemented using fully connected layers, which includes a set of hypernetworks that generate the weights and biases for the mixing network, with the parameters conditioned on the global state s of LLM i. These hypernetworks ensure that each LLM's Q-value Q_i is proportional to the global Q-value Q_{tot}, thereby constraining the relationship between the team's Q_{tot} and each Q_i as follows:

$$\frac{\partial Q_{\text{tot}}}{\partial Q_i} \geq 0 \tag{2}$$

3.3.2 Role Differentiation Module

To strengthen the respective expertise of each LLM, we introduce a role differentiation module for MLC. It constructs latent variables by sampling the observations of each LLM. The latent variable represents the unique characteristics of each LLM. We enhance the diversity between LLMs by setting an optimization objective to differentiate them from each other. The final optimization objective, together with that of the mixing network, contributes to the overall training process.

Formally, we use a latent variable z_i to represent the feature of LLM i, which is used to calculate the similarity among LLMs. Using a hidden variable instead of a fixed vector enhances the robustness and uncertainty of the reasoning task. The Gaussian distribution of these latent variables is estimated through a network f, which consists of two fully connected layers. f is LLM-specific. Each z_i is sampled from a Gaussian distribution. The module takes the observation o_i as input and outputs the latent variable z_i:

$$\begin{aligned} \mathcal{N}_i(\mu_{z_i}, \sigma_{z_i}) &= f(o_i), \\ z_i &\sim \mathcal{N}_i(\mu_{z_i}, \sigma_{z_i}) \end{aligned} \tag{3}$$

where $\mathcal{N}_i$ represents the Gaussian distribution. The optimization objective of the role differentiation

module consists of four terms:

$$L_{\text{dis}} = \sum_{i=1}^{N} \Big(w_{\text{MI}} \cdot \text{MI}(z_i, a_i) - w_{\text{KL}} \cdot \text{KL}(\mathcal{N}_i \parallel p(z_i|o_i)) \Big) + \sum_{i \neq j} w_{\text{DI}} \cdot D_\phi(i,j) + w_H \cdot H(Z) \quad (4)$$

The first two terms ensure consistency between the latent variables and the current states of the LLMs. The last term encourages different LLMs to exhibit dissimilar behavior by leveraging their latent representations.

Specifically, the first term is the mutual information between the latent variable z_i of LLM i and its action a_i, as we expect a strong correlation between them. The second term is the KL divergence (Kullback-Leibler divergence) between the latent variable z_i and its conditional probability distribution $p(z_i|o_i)$, which is used to reinforce the association between the latent variable distribution and the observation. The third term is the sum of dissimilarity score $D_\phi(i,j)$ between any pair (LLM i and LLM j) in the team to boost different behaviors among LLMs. To obtain the dissimilarity $D_\phi(i,j)$, we calculate the KL divergence and mutual information between the corresponding latent variables. KL divergence quantifies the difference between two probability distributions. Mutual information between the latent variables is used to assess the degree of cooperation, reflecting the interdependence and influence among LLMs. So $D_\phi(i,j)$ is obtained as follows:

$$D_\phi(i,j) = \alpha \cdot \text{KL}(\mathcal{N}_i \parallel \mathcal{N}_j) - \beta \cdot \text{MI}(z_i, z_j) \quad (5)$$

where α and β are balancing coefficients for the two terms. The last term is the entropy loss, which promotes the differentiation among latent variables. The optimization objectives of the differentiation module are combined with the overall objective described in the next section.

3.4 Model Training

The training of the model framework is divided into two stages. First, a single LLM is trained, followed by co-training of a team of multiple LLMs. The single-LLM training phase involves learning the role-aware policy network and the differentiation module, where the role-aware RNN is randomly initialized and trained end-to-end through the mixing network. The multi-LLM team training phase focuses on optimizing the mixing network to promote collaboration among LLMs.These two stages are jointly optimized during training, and the final training objective involves two components.

One optimization objective is the standard temporal-difference (TD) loss L_{TD} to encourage coordinated behavior. The model parameters are updated via gradients computed from this TD loss.

In multi-agent system training, we combine the role differentiation loss L_{dis} with the traditional policy loss L_{TD} to form the total loss L_{tot}:

$$L_{\text{tot}} = L_{\text{dis}} + L_{\text{TD}} \quad (6)$$

In this way, the training process not only aims to maximize each LLM's individual Q-value but also promotes behavioral differentiation among LLMs, enabling both effective collaboration and behavioral differentiation.

4 Experiments

4.1 Experimental Settings

4.1.1 Dataset

To evaluate the effectiveness of our approach, we conduct experiments using seven mathematical and reasoning datasets: AddSub (Hosseini et al., 2014), SingleEQ (Koncel-Kedziorski et al., 2015), MultiArith (Roy and Roth, 2016), GSM8k (Cobbe et al., 2021), ASDiv (Miao et al., 2021), SVAMP (Patel et al., 2021), and MATH (Hendrycks et al., 2021). Detailed descriptions of these datasets are provided in the Appendix A.1.

4.2 Implementation Details

4.2.1 Base Models

We use Llama2-7B-chat, Llama2-13B, Llama2-13B-chat, Llama3-8B, Llama3-8B-Instruct, Llama3.1-8B-Instruct, and Mistral-7B-Instruct-v0.2 as base models, since they are widely used and the latest open-source LLMs with strong ability in chatting. Detailed descriptions of these base models are provided in the Appendix A.2.

4.3 Evaluation Metrics

Each question in the mathematical reasoning datasets used in our experiments has a well-defined correct answer, enabling a straightforward and objective evaluation. We report the average accuracy (ACC) of model predictions. We prompt the LLMs to output their final answers in a specific format, simplifying its extraction.

model	dataset(%)	GSM	MATH	ASDiv	SVAMP	MultiArith	SingleEQ	AddSub
Llama-2	7B-chat	24.64	5	55.02	42.91	66.53	62.98	55.05
	LLM Agora	16	4	45.16	34	62	53	41.08
	Multi-Agent (Debate)	28	4	48	50	68	64	54
	MLC	**33**	**5**	**56**	**54**	**73**	**66**	**59**
	13B	24.3	6.3	45.18	41.6	56.83	58.46	58.99
	LLM Agora	27	5	43	37	62.2	58	44
	Multi-Agent (Debate)	31	7	49	43.1	69	61	53
	MLC	**32**	**12**	**61**	**49**	**73.77**	**68**	**61.49**
	13B-chat	42	10	45	52	75	65.06	60
	LLM Agora	16	4	21	31.89	66	54.51	44
	Multi-Agent (Debate)	41	10	55	56	79	72	62
	MLC	**45**	**11**	**63**	**62**	**88**	**74**	**68**
Llama-3	8B	57.2	33	67	71	89.91	68	74.44
	LLM Agora	36	12	47	56	88.65	40	69
	Multi-Agent (Debate)	57	16.6	74	75.8	94	77	72
	MLC	**63**	19	**80**	**83**	**93.55**	**78.2**	**85**
	8B-Instruct	42	29.10	64	77	82.73	66.93	62
	LLM Agora	42	25	58	88	75	60.42	56
	Multi-Agent (Debate)	52	27	72	80	89	72	77
	MLC	**57**	**49**	**76**	**81**	**90**	**73**	**84**
Llama-3.1	8B-Instruct	84.5	51.9	71	81	92	76	87
	LLM Agora	58	37	44	50.94	73	54	66
	Multi-Agent (Debate)	72	50	85	86	95	85	89
	MLC	83.63	**54**	**88**	**90**	**97**	**88**	**91**
Mistral	7B-Instruct-v0.2	38	12.2	63.33	59.42	69.6	72	79.85
	Multi-Agent (Debate)	50.67	21	63	67	72	79	80
	MLC	**69**	16	**69**	**69**	**79**	75	**82**

Table 1: Performance comparison with single LLM method and multiple LLMs debate method. The bold numbers refer to the best performance among all the models.

Following prior work (Liang et al., 2023), we randomly sample 100 questions from each dataset and count the number of correct answers. For multi-model experiments, we adopt a majority voting strategy, where the most frequent response among all generated answers is selected and compared against the correct answer.

4.4 Competing Methods

To validate the efficacy of our proposed method, we compare it with Multi-Agent (Debate) method and LLM Agora method. Multi-Agent (Debate) method utilizes multiple LLMs to engage in iterative discussions and debates. LLM Agora method follows the overall framework of the LLM multi-agent debate and adds summarization. Due to limitations on the number of Llama input tokens, LLMs are set up to conduct two rounds of debate, with the historical conversation from the previous round provided as input. The performance of multiple LLMs is shown in Table 1.

4.5 Overall Performance

MLC uses seven base models from the Llama series and a Mistral-7B-Instruct-v0.2, augmented with a basic Chain-of-Thought (CoT) prompting technique. The performance of the single LLM method is shown in Table 1, and our analysis is in Appendix A.4. Compared with Multi-Agent (Debate) method, MLC achieves greater performance improvements on the seven base models: Llama2-13B improves by 6.31%, Llama2-7B-chat improves by 4.29%, Llama2-13B-chat improves by 5.14%, Llama3-8B improves by 5.05%, Llama3-8B-Instruct improves by 5.86%, Llama3.1-8B-Instruct improves by 4.23%, and Mistral-7B-Instruct-v0.2 improves by 18.33%.

In summary, the proposed method (MLC) performs better than Multi-Agent (Debate) method on these datasets in most cases. The performance

model \ dataset(%)	GSM	MATH	ASDiv	SVAMP	MultiArith	SingleEQ	AddSub
MLC	33	5	56	54	73	66	59
$-role\ prompt$	31	9	50.75	52	72	63.28	57.78
$-mixing\ network$	25	11	53.74	50	69	65	56
$-role\ differentiation\ module$	27	5	52.44	46.67	69	64.71	58

Table 2: The ablation studies of our method on Llama2-7B-chat

.

Method	Base Model	Round	dataset(%): gsm
our	llama2-7B-chat	2	**33**
		3	26
LLM Agora	llama2-7B-chat	2	16
		3	**18**

Table 3: Analysis experiment performance at 3 rounds.

of Multi-Agent (Debate) method is shown in Table 1. In 49 scenarios consisting of seven datasets and seven base models, our method performs significantly better in 44 scenarios and has the same performance in one scenario. Through negotiation, debate, and information sharing among models, multi-view analysis of problems can be achieved, thereby enhancing the accuracy and reliability of model reasoning.

4.6 Ablation study

We conducted experiments to assess the contributions of various modules in the MLC method, using Llama2-7B-chat as the base model, as shown in Table 2. Ablation studies of the other five base models are provided in Appendix A.3. Although performance on the MATH dataset was suboptimal, it is likely due to the complexity of math and Llama's limitations in instruction execution. Specifically, we evaluate the effectiveness of the role prompt, mixing network, and role differentiation module. Removing the role prompt caused a slight performance decrease, while omitting the mixing network led to a substantial drop, underscoring the importance of cooperation in multi-agent systems. The role differentiation module also proved to be critical by enhancing differentiation among LLM behaviors. Overall, MLC significantly improves multi-agent performance in collaborative tasks.

4.7 Analysis on Different Size Base Models for Multiple LLMs

The goal of this experiment is to analyze the impact of using different base model sizes for multiple large language models (LLMs) collaborative debate in a multi-agent system. Specifically, we explore how models with different capacities (Llama2-7B-chat, Llama2-13B-chat, and Llama3-8B-Instruct) influence the role differentiation among agents in collaborative tasks. Different role-specific prompts are given to each LLM based on their size.

On the GSM dataset, the accuracy is 57.97%, slightly exceeding the performance of using three Llama3-8B-Instruct models. We also observe that the agents' outputs are more divergent when they are given appropriate role prompts, thus validating our hypothesis that different base models necessitate distinct role-specific prompts. This division of labor can potentially optimize the system's efficiency and overall task performance.

4.8 Analysis on Different Rounds

We conducted an experiment by changing the number of debate rounds to 3. The results are shown in Table 3. In LLM Agora method, accuracy improved as the number of rounds increased. However, in our approach, we observed that setting the number of debate rounds to three resulted in worse performance compared to just two rounds. This indicates that our method can obtain the final answer more efficiently, which to some extent confirms that fully utilizing the strengths of each LLM can improve collaboration efficiency. We also speculate that more rounds introduce additional complexity in coordinating the debate, leading to potential communication breakdowns or misalignment between agents, which can hinder overall performance.

5 Conclusion

In this paper, we propose a Multi-LLM Cooperation (MLC) method that incorporates multiple LLMs with various roles to perform reasoning tasks. We design an RL-based joint learning method that can adapt to the actual role of each LLM according to the learning status. We equip the joint learning with latent variables to model each

LLM's characteristics and also increase the generation differentiation. Our framework also uses a mixing network and a hypernetwork to control each LLM's contribution and achieve co-training. Experiments indicate that our method with lightweight models outperforms baselines across seven benchmarks.

Limitations

For mathematical reasoning datasets, many works have achieved good results in single-agent problem-solving using methods such as CoT and self-reflection. However, this paper does not focus on fine-tuning single-agent problem-solving techniques but primarily proposes an innovative collaboration method. In the future, the effectiveness of this collaboration framework can be further validated in a more refined single-agent problem-solving setting. Additionally, the increased number of models and reliance on natural language processing can lead to significant computational resource and time consumption, making the exploration of communication efficiency in multi-LLM cooperation a valuable area of study.

Acknowledgments

This work was supported in part by National Natural Science Foundation of China Distinguished Young Scholar Project(Grant No. 62125604), the Guangdong Basic and Applied Basic Research Foundation under Grant 2023A1515011370, the National Natural Science Foundation of China (32371114), the Characteristic Innovation Projects of Guangdong Colleges and Universities (No. 2018KTSCX049), the Guangdong Provincial Key Laboratory [No.2023B1212060076].

References

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, and Torsten Hoefler. 2023. Graph of thoughts: Solving elaborate problems with large language models. *arXiv preprint arXiv:2308.09687*.

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. *arXiv preprint arXiv:2308.07201*.

Justin Chih-Yao Chen, Swarnadeep Saha, and Mohit Bansal. 2023. Reconcile: Round-table conference improves reasoning via consensus among diverse llms. *arXiv preprint arXiv:2309.13007*.

Maximillian Chen, Ruoxi Sun, Sercan Ö Arık, and Tomas Pfister. 2024a. Learning to clarify: Multi-turn conversations with action-based contrastive self-training. *arXiv preprint arXiv:2406.00222*.

Weize Chen, Jiarui Yuan, Chen Qian, Cheng Yang, Zhiyuan Liu, and Maosong Sun. 2024b. Optima: Optimizing effectiveness and efficiency for llm-based multi-agent system. *arXiv preprint arXiv:2410.08115*.

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

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 arXiv:2305.14325*.

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 arXiv:2305.10142*.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Nan Duan, and Weizhu Chen. 2023. Critic: Large language models can self-correct with tool-interactive critiquing. *arXiv preprint arXiv:2305.11738*.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*.

Sirui Hong, Xiawu Zheng, Jonathan Chen, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, et al. 2023. Metagpt: Meta programming for multi-agent collaborative framework. *arXiv preprint arXiv:2308.00352*.

Mohammad Javad Hosseini, Hannaneh Hajishirzi, Oren Etzioni, and Nate Kushman. 2014. Learning to solve arithmetic word problems with verb categorization. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 523–533.

Rik Koncel-Kedziorski, Hannaneh Hajishirzi, Ashish Sabharwal, Oren Etzioni, and Siena Dumas Ang. 2015. Parsing algebraic word problems into equations. *Transactions of the Association for Computational Linguistics*, 3:585–597.

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

Junyou Li, Qin Zhang, Yangbin Yu, Qiang Fu, and Deheng Ye. 2024. More agents is all you need. *arXiv preprint arXiv:2402.05120*.

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. *arXiv preprint arXiv:2305.19118*.

Hao Ma, Tianyi Hu, Zhiqiang Pu, Boyin Liu, Xiaolin Ai, Yanyan Liang, and Min Chen. 2024. Coevolving with the other you: Fine-tuning llm with sequential cooperative multi-agent reinforcement learning. *arXiv preprint arXiv:2410.06101*.

Weixing Mai, Zhengxuan Zhang, Yifan Chen, Kuntao Li, and Yun Xue. 2024. Geda: Improving training data with large language models for aspect sentiment triplet extraction. *Knowledge-Based Systems*, 301:112289.

Shen-Yun Miao, Chao-Chun Liang, and Keh-Yih Su. 2021. A diverse corpus for evaluating and developing english math word problem solvers. *arXiv preprint arXiv:2106.15772*.

Xianghe Pang, Shuo Tang, Rui Ye, Yuxin Xiong, Bolun Zhang, Yanfeng Wang, and Siheng Chen. 2024. Self-alignment of large language models via multi-agent social simulation. In *ICLR 2024 Workshop on Large Language Model (LLM) Agents*.

Joon Sung Park, Joseph 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*, pages 1–22.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are nlp models really able to solve simple math word problems? *arXiv preprint arXiv:2103.07191*.

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 arXiv:2307.07924*, 6.

Subhro Roy and Dan Roth. 2016. Solving general arithmetic word problems. *arXiv preprint arXiv:1608.01413*.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Yifan Song, Da Yin, Xiang Yue, Jie Huang, Sujian Li, and Bill Yuchen Lin. 2024. Trial and error: Exploration-based trajectory optimization for llm agents. *arXiv preprint arXiv:2403.02502*.

Rasal Sumedh. 2024. Llm harmony: Multi-agent communication for problem solving. *arXiv preprint arXiv:2401.01312*.

Xiangru Tang, Anni Zou, Zhuosheng Zhang, Yilun Zhao, Xingyao Zhang, Arman Cohan, and Mark Gerstein. 2023. Medagents: Large language models as collaborators for zero-shot medical reasoning. *arXiv preprint arXiv:2311.10537*.

Kuan Wang, Yadong Lu, Michael Santacroce, Yeyun Gong, Chao Zhang, and Yelong Shen. 2023. Adapting llm agents through communication. *arXiv preprint arXiv:2310.01444*.

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

Zhengxuan Zhang, Yin Wu, Yuyu Luo, and Nan Tang. 2024. Mar: Matching-augmented reasoning for enhancing visual-based entity question answering. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 1520–1530.

Xuanhe Zhou, Guoliang Li, and Zhiyuan Liu. 2023. Llm as dba. *arXiv preprint arXiv:2308.05481*.

A Appendix

A.1 Datasets

AddSub (Hosseini et al., 2014) focuses on simple addition and subtraction problems, which help evaluate the model's accuracy and efficiency in basic arithmetic operations. **SingleEQ** (Koncel-Kedziorski et al., 2015) provides structured mathematical problems centered on the single equation, aimed at assessing the model's ability to solve straightforward yet foundational mathematical tasks. Both datasets contain relatively simple problems that do not require multi-step calculations. **MultiArith** (Roy and Roth, 2016) addresses multi-step arithmetic problems, challenging the model's capacity to handle more complex tasks. **GSM8K** (Cobbe et al., 2021) is designed for tasks that require multi-step reasoning to solve basic mathematical problems, typically involving 2-8 steps, thereby effectively evaluating the model's mathematical and logical reasoning abilities. **ASDiv** (Miao et al., 2021) offers a collection of diverse mathematical application problems, including algebra, geometry, and probability, to provide a comprehensive assessment of our model. **SVAMP** (Patel et al., 2021) is intended to thoroughly evaluate the performance of automatic math word problem (MWP) solvers, focusing on aspects such as problem sensitivity and reasoning reliability. **MATH**

model \ dataset	GSM(%)	MATH(%)	ASDiv(%)	SVAMP(%)	MultiArith(%)	SingleEQ(%)	AddSub(%)
MLC(Llama2-13B)	32	12	61	49	73.77	68	61.49
$-role\ prompt$	30	10	50.4	42	60	66	60
$-mixing\ network$	26	7	56	48	65	64	61
$-role\ differentiation\ module$	29	10	55.77	44	61.05	63	60.2
MLC(Llama2-13B-chat)	45	11	63	62	88	74	68
$-role\ prompt$	44.02	10	61	53.87	84	66	64
$-mixing\ network$	41	7	55.7	60	76	72.39	61
$-role\ differentiation\ module$	42	11	45	58	77	68.45	63
MLC(Llama3-8B)	63	19	81	83	93.55	78.2	85
$-role\ prompt$	60.81	25	80	76.19	90	78	84
$-mixing\ network$	59	26	74.74	81	92.38	77	75.9
$-role\ differentiation\ module$	58.11	28	78	78.49	91	74	78.57
MLC(Llama3-8B-Instruct)	57	49	76	81	90	73	84
$-role\ prompt$	50	40.67	70	77	83	70	75
$-mixing\ network$	53	45	66	78	88	68	76
$-role\ differentiation\ module$	44	30.84	66.92	77.2	85	69	71
MLC(Llama3.1-8B-Instruct)	83.63	54	88	90	97	88	91
$-role\ prompt$	79.1	53.1	85	84.43	93.33	77	90.11
$-mixing\ network$	80	52	80	85	95	85.36	88.6
$-role\ differentiation\ module$	78.51	52.76	82	86	94.21	79	90

Table 4: The ablation studies of our method.

.

(Hendrycks et al., 2021) contains 12,500 math competition problems ranging from basic to advanced levels. It assesses the model's ability to tackle complex math problems. Among the datasets, the MATH dataset poses the most difficult challenges.

A.2 Base Models

The Llama series of large language models represents a significant advancement in natural language processing (NLP) in recent years, gaining widespread attention for their powerful text generation, understanding, and reasoning capabilities. This series includes various versions, ranging from the basic model to those specifically optimized for tasks such as dialogue and instruction-following, providing robust support for different NLP tasks. We trained and evaluated our approach on the following seven models in the Llama series. Llama2-7B-chat, the dialogue (chat) version of Llama-2, with 7 billion parameters, demonstrates strong text generation and comprehension capabilities. Compared to the 7B version, Llama2-13B increases the parameter count to 13 billion, further enhancing the model's representational capacity and generalization performance. The larger model size enables it to excel in handling complex language phenomena and generating high-quality text. Llama2-13B-chat, based on Llama2-13B, is specifically optimized for dialogue scenarios, allowing it to understand better and respond to dialogue structure and contextual information in human language. Another variant, Llama3-8B, introduces 8 billion parameters and emphasizes flexibility and scalability in its design. Llama3-8B-Instruct builds on Llama3-8B, incorporating a human-in-the-loop feedback mechanism to optimize instruction-following, enabling the model to more accurately understand and execute human-given instructions. Finally, the latest member of the Llama series, Llama3.1-8B-Instruct, maintains the 8 billion parameter size while introducing advanced training strategies and datasets to further enhance the model's instruction-following capabilities and the quality of the generated text.

A.3 Ablation study

We conducted experiments to assess the contributions of various modules in the MLC method. Ablation studies of the other five base models will be provided in Table 4.

A.4 Single LLM Comparison Analysis

To conduct an extensive analysis to gain a deeper understanding of the MLC framework, we conduct experiments to analyze the single LLM method on all base models and datasets. The single LLM performance is shown in Table 1. In 42 scenarios consisting of seven datasets and seven base models, our method is much higher in 39 scenarios, tied in 1 scenario, and slightly lower than 0.87% in 1 scenario, the only exception is that the accuracy is low in one scenario. MLC achieves an average accuracy of 9.41% higher on these datasets. The performance of all single LLMs is relatively low on all datasets. Though they use CoT for reasoning,

they are susceptible to issues such as model bias and degradation of thinking during the reasoning process, and they are unable to engage in reflective learning.

A.5 Potential Risks

The multiple LLMs cooperation framework is designed to solve mathematical problems, but it has the possibility to also be applied to other illegal or immoral applications, including reasoning the private information or sensitive information from public reports or news. We should carefully apply our proposed methods to a limited set of applications. In the future, we plan to design some rules into our method to avoid being used in illegal or immoral applications.

A.6 Discussion on use of license (terms for use) and distribution of any artifacts

The datasets we used in this paper come from the publicly released dataset that allows for the use of research (with the corresponding licences). We did not use any artifacts in this paper.