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

Anonymous ACL submission

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 idea is to adopt a relatively general role assignment mechanism, such as adding a "judge" or a summary role, but such methods cannot customize the task-specific role assignment mechanism according to the characteristics of the task. Another idea is to decompose the task according to domain knowledge and task characteristics, and then assign appropriate roles to LLMs according to their strengths, such as programmers and testers. However, in some given tasks, it's hard to obtain domain knowledge related to task characteristics and get the strengths of different LLMs. To solve the above problems, we propose a Multi-LLM Cooperation (MLC) method with automatic role assignment capabilities. The main idea of MCL is to randomly initialize role assignments first, and then let role embeddings learn together with downstream tasks. To record the state changes of multiple LLMs when they take turns speaking, the role embedding is sequence-aware. At the same time, to avoid role convergence, the role differentiation module of MCL encourages behavioral differences between LLMs while ensuring the consistency of the LLM team, guiding different LLMs to achieve complementary advantages from the optimization level. Our experiments on seven datasets show that our approach significantly improves debate collaboration and expertise to collaboratively solve multi-agent debate tasks[1].

1 Introduction

Multiple agents collaborate through debate, which can give full play to the capabilities of multiple agents and use collective intelligence to solve more complex tasks. The multi-agent debate has a wide range of applications in Artificial Intelligence (AI), such as coding (Qian et al., 2023), teamwork projects, negotiation (Fu et al., 2023), and mathematics reasoning tasks. Methods for multi-agent debate contain two types of approaches. One type of approach utilizes a general role assignment mechanism. Some researchers explore the use of multiple LLMs with 7B $\sim$ 13B (e.g. Llama2-7b) to solve mathematical reasoning aiming at handling lightweight scenarios and consider the reasoning paths in various perspectives (Ma et al., 2024). The classic method involves multiple LLMs taking turns to generate responses, with each answer updated based on the responses of the other LLMs, such as Multi-agent Debate (Du et al., 2023). The MAD framework (Liang et al., 2023) is proposed to be a multi-agent debate framework, with multiple debaters engaging in sequential debate, with a judge making the final decision. The concept of AI feedback (Fu et al., 2023) involves two LLM agents debate, a third LLM provides feedback for improvement, facilitating the negotiation process to reach an agreement. ChatEval (Chan et al., 2023) manually assigns roles to different agents, thereby making full use of the differentiated skills and expertise of LLMs, and introduces three communication strategies to enhance the reliability of the multi-agent framework in solving tasks. Papers on this type of approach fail to customize task-specific role assignment mechanisms based on task characteristics, limiting the ability to fully leverage the strengths of each agent's role.

Another approach involves designing specialized LLMs for specific tasks, achieving collaboration by presetting fine-grained agent roles and fully utilizing the distinct characteristics of each agent. Among these directions, some researchers encourage the multiple LLMs to cooperate while assigning each LLM a specific role via prompt learning (Chen et al., 2024b). Those methods are

[1]Our code is available at https://anonymous.4open.science/r/MLCan-D102/.

easy to implement and feasible to various kinds of instructions, including the instructions to execute reasoning (i.e. CoT (Wei et al., 2022) and GoT (Besta et al., 2023)) and the instructions to incorporate examples via ICL (Li et al., 2024). Since the roles and behaviors of the intelligent agents must be predefined, it requires sufficient domain knowledge and a thorough understanding of the unique features of the LLMs, leading to limited scalability. Additionally, with model parameters fixed, the LLMs cannot learn to deeply adapt to their assigned roles (tasks) (Du et al., 2023; Chen et al., 2023). Therefore, some recent works have begun exploring supervised fine-tuning of LLMs to enable them to better master the given tasks (Schulman et al., 2017; Chen et al., 2024a; Song et al., 2024; Wang et al., 2023). To achieve collaboration, these methods integrate the generated responses into the training trajectory and use reinforcement learning (RL) techniques to fine-tune the LLMs. However, current multi-LLM collaboration methods do not account for role differentiation during the collaborative tuning process, nor do they design a framework that leverages the complementary advantages of each model.

In this paper, we argue that the role assignments to each LLM are crucial for multiple LLM cooperation (Sumedh, 2024; Tang et al., 2023; Pang et al., 2024; Li et al., 2023). We note that prompting approaches and separate fine-tuning approaches result in LLM's role being fixed and cannot be dynamically adjusted according to the real status during learning. We should consider the overall learning goal and unify the joint learning of all LLMs to enable learners to adjust to the role. To avoid manual setting and better align with task requirements in the role setting of collaborative tasks, we propose a Multi-LLM Cooperation (MLC) framework for automatic role assignment in mathematical reasoning tasks. Specifically, we developed a MARL-based training framework for multiple LLMs, enabling collaborative optimization under a unified optimization objective. We then introduced time-sensitive role encoding for each LLM, combining this encoding with a mixing network to facilitate role differentiation during collaboration. Finally, to prevent role convergence, we implemented a role differentiation module that models the state of each LLM, capturing the unique characteristics of each. This approach 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 7 datasets. Our contributions are as follows:

- We propose a Multi-LLM Cooperation (MLC) framework with role differentiation, 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 model parameter level.
- The experiments demonstrate the proposed method's effectiveness and scalability across seven standard datasets that encompass multiple ranges 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. Such studies have conducted various explorations into how multiple LLMs can engage in debates or information exchanges. The Multi-Agent Debate (MAD) framework (Liang et al., 2023) is proposed to address the Degeneration-of-Thought (DoT) problem. In Multi-Agent (Debate), multiple debaters engage in sequential debate, with a judge making the final decision. Du et al. (Du et al., 2023) achieve multi-angle analysis of problems through debates and information sharing among models. Building upon this, 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, it primarily explores, rather than improves, the problem-solving ability 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. 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, some research endeavors to enhance performance by assigning distinct roles to LLM agents, and leveraging and integrating the unique characteristics of different agents. One type of approach utilizes 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. MAD proposed a multi-agent debate (MAD) framework, where multiple debaters debate in sequence, and a judge makes the final decision. The concept of 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, acting as an AI critic, provides feedback to facilitate the negotiation process to reach an agreement. ChatEval (Chan et al., 2023) manually assigns roles to different agents and encourages consistency among agents. However, The simplicity of the role design, typically limited to only two or three categories, is relatively general and constrains the full utilization of each agent's strengths.

Another category of approaches is to assign delicately designed roles to different LLMs. Generative Agents (Park et al., 2023) build a sandbox environment with 25 agents, each assigned different roles, such as artists and authors. ChatDev (Qian et al., 2023) is a virtual chat-powered software development company that established 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) works by mimicking the process of humans utilizing external tools to modify 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 to different agents, effectively decomposing complex tasks into subtasks. These approaches pre-define 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. By jointly training multiple language models, the models can autonomously determine their roles in the collaborative task, based on their individual 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 input 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 answer by integrating the responses from all other LLMs and the role prompts.

Step 3: This process continues iteratively, with each LLM refining its answer based on the evolving dialogue and prompts in each round. The debate continues until round M.

We introduce a Multi-LLM Cooperation (MLC) framework in Figure 1, which incorporates multi-agent reinforcement learning (MARL) into multi-LLM cooperation, conducts the collaboration, and trains between multiple LLMs to promote role differentiation to enable adaptive shared learning among LLMs. Each LLM is treated as an agent and multiple LLMs collaborate as multi-agent cooperation, the responses generated by LLMs serve as actions, multi-round debates as environments. To encourage different LLMs to make different contributions to the collaboration, we add 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 whole systems in subsection 3.4. The details of the debate process are shown in Figure 2.

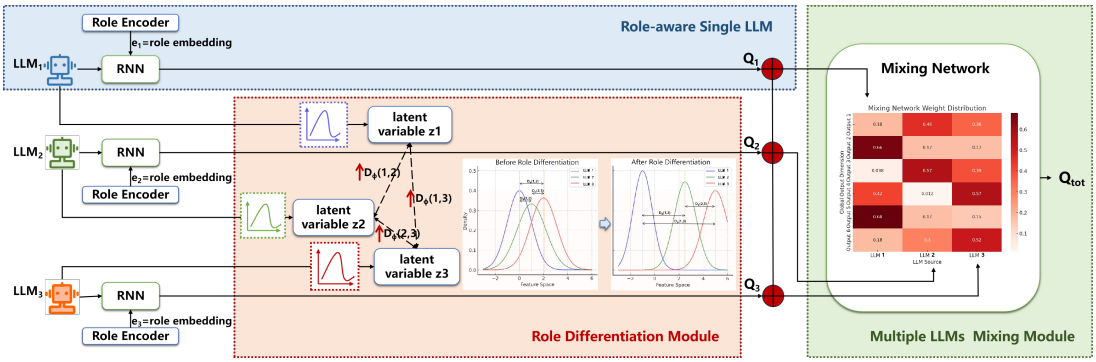

Figure 1: Components of the MLC. The blue dashed box represents the Role-aware Single LLM. The green dashed box represents the Multiple LLMs Mixing Module. The red dashed box represents the Role Differentiation Module.

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 is noted 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 LLM 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 chatting histories, which is formulated as $s^t = \{Q, P, \{u^1, u^2, \ldots, u^t\}\}$.

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

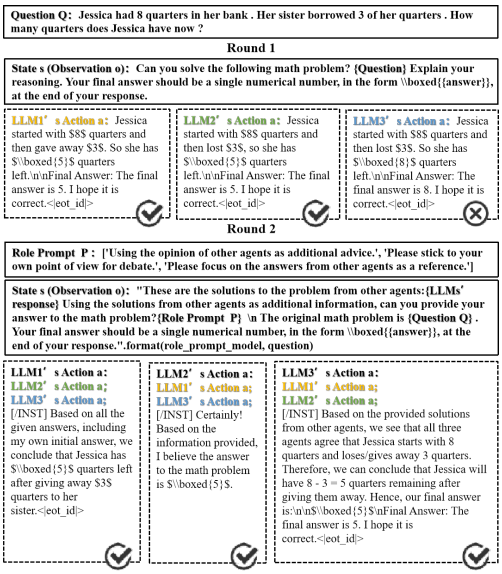

Figure 2: The example of our task is to use multiple LLMs to collaboratively solve a given question through debate, with a total of 3 LLMs and 2 rounds.

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 group 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. For the specific time step t

for LLM i, which represents the rounds of dialogue. Specifically, the role embedding information of multiple LLMs and the state space information of the corresponding current time step are input into the role perception network in the corresponding LLM. To model the time sequence of continuous actions of LLM, we use 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 i, a_i represents the action space information of LLM i, while s_i denotes its state space information. The role embedding r_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 r_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 their 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 group is processed collaboratively. The collaboration consists of a multiple LLMs mixing module and a role differentiation module. The multiple LLMs mixing module ensures team consistency across different roles, while the role differentiation module enhances behavioral diversity among LLMs and operates independently of other modules.

3.3.1 Multiple LLMs Mixing Module

To facilitate effective cooperation among LLMs, our method employs a mixing network to integrate single role-aware network of each LLM. Mixing network also ensures consistency between the individual LLMs and the overall group.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 by fully connected layers, which include a set of hypernetworks, generating the weights and biases for the mixing network, conditioning it on the global state s_i 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 group's Q_{tot} and each Q_i as $\frac{\partial Q_{\text{tot}}}{\partial Q_i} \geq 0$.

3.3.2 Role Differentiation Module

To strengthen the respective expertise of each LLM, we introduce a role differentiation module for MLC. It constructs a latent variable by sampling the observations of each LLM to represent the unique characteristics of each LLM and enhances the diversity between LLMs by setting optimization objectives to differentiate them from each other. The final optimization objective, together with the optimization objective of the mixing network, works together in the entire training process.

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

$$\mathcal{N}_i(\mu_{z_i}, \sigma_{z_i}) = f(o_i) \tag{2}$$

$$z_i \sim \mathcal{N}_i(\mu_{z_i}, \sigma_{z_i}) \tag{3}$$

where $\mathcal{N}_i$ represents Gaussian distribution. The optimization objective of the role differentiation module consists of four terms:

$$L_{\text{dis}} = \sum_{i=1}^{N} (w_{\text{MI}} \cdot \text{MI}(z_i, a_i) \tag{4}$$

$$- w_{\text{KL}} \cdot \text{KL}(\mathcal{N}_i \parallel p(z_i|o_i))) + \sum_{i \neq j} w_{\text{DI}} \cdot D_\phi(i,j) + w_H \cdot H(Z) \tag{5}$$

The first two ensure the consistency between the latent variable and the current state of LLMs. The last is to encourage different LLMS to be as dissimilar as possible through implicit variables.

Specifically, the first one is the Mutual information between the latent variables of the LLM's latent variable z_i and its action a_i, since we expect a high coexistence between these two. 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 value $D_\phi(i,j)$ between any pair (LLM i and LLM j) in the group 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 LLMs is calculated to assess the degree of cooperation, reflecting their interdependence and influence. So $D_\phi(i,j)$ is obtained as follows,

$$D_\phi(i,j) = \alpha \cdot \mathrm{KL}(\mathcal{N}_i \parallel \mathcal{N}_j) - \beta \cdot \mathrm{MI}(z_i, z_j) \quad (6)$$

where α and β are used to balance the two items. The last term is the entropy loss, which promotes the diversity among latent variables.

The optimization objectives of the differentiation module are combined with the overall one 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, and then multiple LLM groups are co-trained. The single LLM training part is the role-aware network and the differentiation module, the LLMs' group training part is the mixing network. The two stages in the framework are trained together, and the final training goal corresponds to two optimization targets.

One of the optimization targets is the standard loss of the role-aware network and the mixing network to promote the collaborative process. The parameters in the framework are updated using the gradient caused by the standard TD loss of reinforcement learning Our approach incorporates an optimization objective that enhances behavioral differentiation among LLMs. Simultaneously, all framework parameters are updated using gradients derived from the standard temporal difference(TD) loss L_{TD} in reinforcement learning. The mixing network's output, when combined with the dissimilarity loss L_{dis}, constitutes the training objective. This objective is utilized to compute the global loss for centralized training. 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}:$L_{\text{tot}} = L_{\text{dis}} + L_{TD}$.

In this way, the training process not only focuses on maximizing each agent's individual Q value or policy utility but also emphasizes the behavioral differentiation among agents.

4 Experiments

4.1 Experimental Settings

4.1.1 Dataset

To evaluate the effectiveness of our approach, we conducted 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, Mistral-7b-Instruct-v0.2 as the base model, 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.

4.2.2 Training Details

For training processing, each epoch was trained on NVIDIA 4*A100 for about 2 hours. We trained all models with Float16 numerical format and temperature was set to 0.35, top-p was set to 0.9, with all other parameters set to their default values.

4.3 Evaluation Metrics

Each question in the mathematical reasoning datasets used in this experiment has a standard correct answer, allowing for a straightforward comparison that enables accurate score calculation across different models, we report the average accuracy (ACC) of predictions. We instruct LLMs to present the final answer in a specific format through a prompt, simplifying its extraction. Following previous works (Liang et al., 2023), each dataset randomly selects 100 questions, with the number of

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.

correct answers recorded. For multi-model experiments, we use a majority voting approach, where the most frequent response among generated answers was selected for comparison with the standard correct answer.

4.4 Competing Methods

To validate the efficacy of our proposed method, we compared it with the Multi-Agent (Debate) method, the LLM Agora method, and the MALT method. The Multi-Agent (Debate) method utilizes multiple LLMs to engage in iterative conversational discussions and debates. The LLM Agora method follows the overall framework of the LLM multi-agent debate and adds additional summarization. The MALT method employs a sequential multi-agent setup with heterogeneous LLMs assigned specialized roles: a generator, verifier, and refinement model iteratively solving problems. Due to limitations in the number of Llama input tokens, LLMs were set up to conduct two rounds of debate, with the historical conversation from the previous round provided as input. The multiple LLMs performance is shown in Table 1.

4.5 Overall Performance

The MLC uses 6 base models from the Llama series and a Mistral-7b-Instruct-v0.2, augmented with a basic Chain-of-Thought (CoT) prompting technique. The single LLM method performance is shown in Table 1, and Our analysis is in appendix A.4. Compared with the Multi-Agent (Debate) method, MLC has the largest performance improvement on the 7 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 the Multi-Agent (Debate) methods on these data in most cases. The Multi-Agent (Debate) performance is shown in Table 1. In 49 scenarios consisting of 7 datasets and 7 base models, our method is much higher in 44 scenar-

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
$-dissimilarity\ model$	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.

ios and has the same performance in 1 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 shown in Table 2. Ablation studies of the other five base models will be 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 dissimilarity 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 dissimilarity module also proved to be critical by enhancing differentiation among LLM behaviors. Overall, the 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 three base models that are Llama3-8b Instruction. 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 studied the experiment of changing the rounds setting to 3. The results are shown in Table 3. On the LLM Agora method, as the number of rounds increased, the accuracy improved. 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 accomplish a reasoning task. We design an RL-based joint learning method that can adapt to the real 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 diversity. Our framework also uses a mixing network and a hypernetwork to control each LLM's contribution and achieve co-training. Experiments indicate our method with lightweight models excels in baselines over 7 benchmarks.

6 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.

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.

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** (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 \ 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
$-dissimilarity\ model$	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
$-dissimilarity\ model$	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
$-dissimilarity\ model$	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
$-dissimilarity\ model$	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
$-dissimilarity\ model$	78.51	52.76	82	86	94.21	79	90

Table 4: The ablation studies of our method.

.

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 better understand 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 our MLC, 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 7 datasets and 7 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 mathematic problems but it has a possibility to be also applied to other illegal or immoral applications, including reasoning the private information or sensitive information from the 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 public released dataset that allows for the use of

research (with the corresponddng licences). We did not use any artifacts in this paper.