# A Multiple-LLM Cooperation Framework with Role Differentiation for Mathematical Reasoning

Anonymous ACL submission

## Abstract

Large language models (LLMs) perform well on many applications and still face some challenges in mathematical reasoning. To accomplish mathematical reasoning, some researchers prompt an LLM to step-by-step reason to generate the answers. Further, they use multiple LLMs to cooperate on reasoning and assign each LLM a role (task) via prompt learning or fine-tuning on each LLM. However, the role assignments of the above approaches are still fixed and cannot be adjusted according to the real status during learning. In this paper, we propose a Multi-LLM Cooperation (MLC) method for mathematical reasoning tasks, where multiple LLMs cooperate via jointly learning with RL and dynamically adjusting the role. Specifically, each LLM serves as a role-aware policy model to generate responses. All the LLMs jointly learn via RL that equips with latent variables to model the unique characteristics of each LLM, where the uncertainty of the variables increases the generation diversity among all LLMs with various roles. Finally, we propose a mixing network to achieve co-training and propose a hypernetwork to the constraint that every LLM makes a positive contribution to the whole model. Experiments indicate our method excels in some strong baselines over 7 public datasets[1].

## 1 Introduction

Large language models (LLMs) obtain remarkable performance on various language understanding applications. In question-answering (QA) scenarios, LLMs can usually generate fluent and precise answers to satisfy the users' given questions, but LLMs are also hard to conduct reasoning about some facts or questions requiring very specific and exact answers (i.e. mathematical reasoning).

The mainstream LLM methods in mathematical reasoning explore an LLM's reasoning ability by leveraging the LLM's generation ability to carry step-by-step reasoning, including chain-of-thought (CoT) (Wei et al., 2022), tree-of-thought (ToT) (Yao et al., 2024), graph-of-thought (GoT) (Besta et al., 2023). Those methods follow human thought characteristics to gradually reason via one or multiple thinking paths and obtain satisfactory performance on mathematical reasoning tasks. However, those methods highly rely on an LLM's self-capability since they mainly conduct the reasoning via prompting the LLMs. As a result, the LLM in this branch is mainly the very large models including GPT-3.5 and GPT-4, which are heavyweight and thus hardly suitable to the scenarios that require lightweight models (e.g. Debate and MAD) (Du et al., 2023; Liang et al., 2023) Further, relying on one LLMs' reasoning paths instead of considering this questions from multiple different perspectives has a risk to lead to the reasoning results incomprehensive and incomplete.

Some researchers explore the use of multiple LLMs with 7B $\sim$ 13B (e.g. Llama2-7b) to solve mathematical reasoning aiming at handling the lightweight scenarios and consider the reasoning paths in various perspectives (Ma et al., 2024). Among these directions, some researchers encourage the multiple LLMs to cooperate while assigning each LLM a specific role (i.e. task) via prompt learning (Chen et al., 2024b). Those methods are 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). However, the model parameters are frozen and cannot learn to deeply adapt to the given role (task) of LLMs (Du et al., 2023; Chen et al., 2023). To further learn to cooperate with multiple LLMs, we can conduct supervised fune-tuning on LLMs to learn to master

[1]Our code is available at https://anonymous.4open.science/r/MARDan-2513/

the given tasks (Schulman et al., 2017; Chen et al., 2024a; Song et al., 2024; Wang et al., 2023). However, current multiple LLMs cooperation methods mainly focus on fine-tuning each LLM separately but seldom joint training over multiple LLMs with an overall target.

The role (task) 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.

In this paper, we propose a Multi-LLM Cooperation (MLC) framework for mathematical reasoning tasks, where multiple LLMs jointly learn to cooperate via reinforcement learning and learn to adjust the role from an initial role. Specifically, for each LLM, we first build a role-aware policy network with specific role embedding to generate responses as actions. We then construct a RL-based multiple LLMs jointly learning framework with role differentiation mechanism. The framework uses latent variables to represent the unique characteristics of each LLM, where the variables are further used to encourage diversity among LLMs. The framework employs a mixing network with to achieve co-training and uses a hypernetwork to ensure that every LLM has a positive contribution at the same time. Specifically, for each LLM, we first build a role-aware policy network with specific role embedding to generate responses as actions. We then construct a RL-based multiple LLMs jointly learning framework with role differentiation mechanism. The framework uses latent variables to represent the unique characteristics of each LLM, where the variables are further used to encourage diversity among LLMs. The framework employs a mixing network with a global optimization objective to achieve co-training, where a hypernetwork is introduced to ensure that every LLM has a positive contribution to the reasoning tasks. 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 Mathematical Reasoning via LLMs

LLMs have demonstrated superior capabilities on a wide range of tasks, and many studies have attempted to utilize rational prompting to improve performance on reasoning tasks. Some methods mainly explore the reasoning ability of large models through prompting techniques, including Chain of Thought (CoT) prompting (Wei et al., 2022), Tree of Thought (ToT) prompting (Yao et al., 2024), Self-Consistency (Wang et al., 2022), Self-Reflection (Shinn et al., 2024) (Madaan et al., 2024), and so on. These methods give more reasonable and accurate answers by processing history, with relatively low consumption costs.

Other studies have developed the performance of large language models through fine-tuning, such as Reinforcement Learning from Human Feedback (RLHF, e.g., PPO) (Schulman et al., 2017) and its variants. Furthermore, 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 updates their strategies using contrastive learning methods like DPO. While these fine-tuning methods offer deeper, more enduring, and more adaptable optimization capabilities for individual LLM agents, we aim to explore a new training framework that can simultaneously train and learn strategies for multiple LLM agents.

### 2.2 Cooperation on Multiple LLMs

To surpass the ability of the single model, researchers explore methods to leverage the multiple large language models (LLMs) to collabo-

rate 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 MAD, multiple debaters engage in sequential debate, with a judge making the final decision. Du et al. (Du et al., 2023) achieves 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, achieving effective improvements on several datasets.

### 2.3 Role Differentiation in LLMs' Cooperation

In terms of cooperation among multi-LLM, some research endeavors to enhance performance by assigning distinct roles to LLM-agent, leveraging and integrating the unique characteristics of different agents. Generative Agents (Park et al., 2023) build a sandbox environment with 25 agents, each assigned different roles, such as artists and authors. These agents spontaneously organized themselves to simulate believable human behavior. MedAgent (Tang et al., 2023) utilizes LLM-based agents to take on roles in the medical domain and engage in multi-round collaborative discussions. ChatDev (Qian et al., 2023) assigns roles to each agent in a chat session, with clear divisions of labor and cooperation among roles; MetaGPT (Hong et al., 2023)leverages a pipeline paradigm to assign different roles to different agents, effectively decomposing complex tasks into subtasks involving cooperation among multiple agents. 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 the player targeted for improvement, facilitating the negotiation process to reach an agreement. ChatEval (Chan et al., 2023) assigns distinct roles to individual agents and introduces three communication strategies for maintaining chat history among agents. 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, while a supervisor LLM consolidates the results. These methods involve role assignment at the linguistic level, with predefined cooperation modes based on the characteristics of the target task, leading to limited scalability. Furthermore, most methods involve similar tasks with simplistic role definitions and a limited number of role categories, which fails to fully exploit the strengths of each role.

To address this issue, this paper proposes a Multi-LLM Cooperation (MLC) method. MLC draws inspiration from the multi-agent reinforcement learning algorithm (MARL) QMIX (Rashid et al., 2020) approach, which uses a centralized critic network to decompose the joint action-value of the agent team into a combination of individual action-values for each agent. MLC applies it to the LLM multi-agent cooperation system and further introduces a role differentiation module corresponding to role embedding. In this way, MLC explores MARL to achieve cooperation among LLMs and uses role assignment at the parameter level for role differentiation.

## 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. The response is produced through the value-based network for individual agents.
- 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-agent cooperation and 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. We will introduce the design of each single LLM in subsection 3.2, the mixing mechenism of multiple

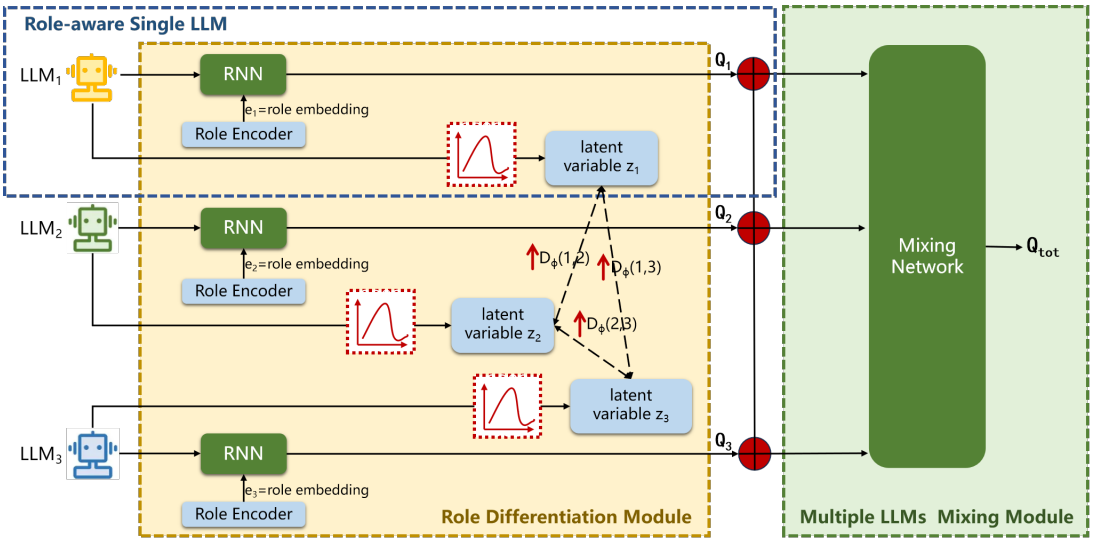


Figure 1: Components of the MLC. The blue dashed box represents the Role-aware Single LLM. The green dashed box represents the Multiple LLMs Mixing Module. The yellow dashed box represents the Role Differentiation Module.

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.

## 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 and current 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 $\pi_i$.** The policy $\pi_i$ 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.

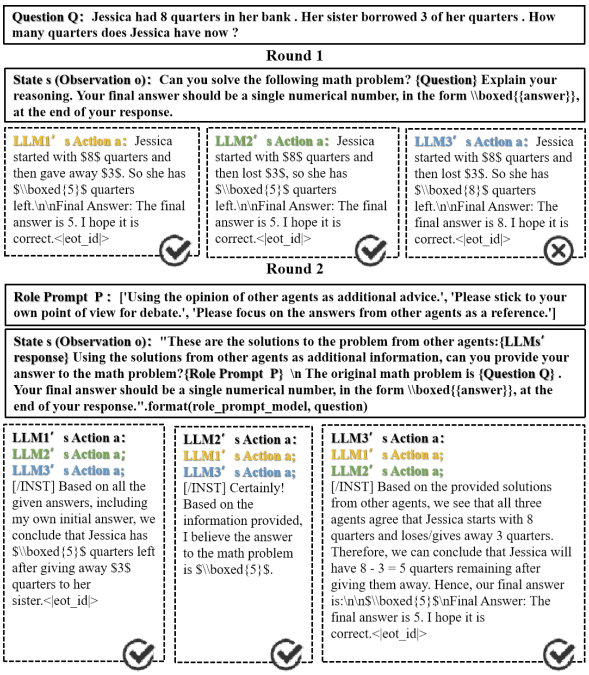


Figure 2: The example of our task is to use multiple LLMs to collaboratively solve a given question through debate, with a total of 3 LLMs and 2 rounds.

### 3.2.2 Role-aware Policy Network

The policy framework allows each LLM to generate responses that are tailored to its specific role, thereby facilitating nuanced interactions in a multi-agent setting. To assign unique roles to each LLM, we first use prompts to stimulate the properties of the LLM. Then, the role-aware policy network takes the prompt 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 group under the current environment.

The role-aware policy framework for each LLM is an RNN with role guidance to enhance role differentiation. Specifically, the role-aware 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, that is, when the round is $t+1$, the time step is $t+1$, each LLM $i$ processes the state $s^t$ and role embedding $e_i$ as input into the RNN, wherein the policy $\pi_i$ generates an action $a_i^t$:

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

where RNN computes the next state based on the current state and action using its LSTM unit. The policy $\pi_i$ (which now refers to the policy itself for LLM $i$) takes in the role embedding $e_i$ and generates the next action.

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. The role-aware agent network fine-tunes each 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. The cooperation part is divided into the Role Differentiation module and the Consistency module. The Role Differentiation module strengthens the behavioral differences between LLMs, and the Multiple LLMs Mixing Module ensures the positive consistency of the LLM group on the basis of differentiation.

#### 3.3.1 Role Differentiation Module

To strengthen the respective expertise of each LLM, we introduce a role differentiation module for MLC. It uses latent variables to represent the unique feature of each LLM and boost the diversity among LLMs by setting optimization goals to keep them away from each other.

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:

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

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

The optimization objective of the role differentiation module consists of four terms. 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 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 (Kullback-Leibler 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,

$$D_\phi(i,j) = \alpha \cdot \text{KL}(\mathcal{N}_i \parallel \mathcal{N}_j) - \beta \cdot \text{MI}(z_i, z_j) \tag{4}$$

where $\alpha$ and $\beta$ are used to balance the two items. The last term is the entropy loss, which promotes the diversity among latent variables. So the loss

function of this module is,

$$L_{\text{dis}} = \sum_{i=1}^{N} (\alpha_{\text{MI}} \cdot \text{MI}(z_i, a_i) \quad (5)$$

$$- \alpha_{\text{KL}} \cdot \text{KL}(\mathcal{N}_i \parallel p(z_i|o_i)))$$

$$+ \sum_{i \neq j} \alpha_{\text{DI}} \cdot D_{\phi}(o_i, o_j) + \alpha_H \cdot H(Z) \quad (6)$$

where $H(Z)$ is the entropy of the latent variable set $Z = \{z_1, z_2, \ldots, z_N\}$. $\alpha_{\text{MI}}$, $\alpha_{\text{KL}}$, $\alpha_{\text{DI}}$ and $\alpha_H$ are factors to balance different terms.

#### 3.3.2 Multiple LLMs Mixing Module

To facilitate effective cooperation among LLMs, MARL employs a mixing network to integrate the agent 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 $Q_{\text{tot}}$. 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 $Q_{\text{tot}}$, thereby constraining the relationship between the group's $Q_{\text{tot}}$ and each $Q_i$ as follows:

$$\frac{\partial Q_{\text{tot}}}{\partial Q_i} \geq 0 \quad (7)$$

### 3.4 Model Training

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.

Specifically, the global TD loss is computed by a single agent network to generate $Q_i$. These values are then input into a mixing network, which jointly trains multiple large models, fine-tunes the parameters, and applies collaborative constraints to calculate the $Q_{tot}$.

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 $L_{tot}$:

$$L_{\text{tot}} = L_{\text{dis}} + L_{TD} \quad (8)$$

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 as the base model, since they are widely used and the latest open-source LLMs with strong ability in chatting.

#### 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 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.

| model \ dataset | GSM | MATH | ASDiv | SVAMP | MultiArith | SingleEQ | AddSub |
|---|---|---|---|---|---|---|---|
| Multi-Agent | | | | | | | |
| Llama2-7b-chat | $28_{(+5)}$ | $4_{(+1)}$ | $48_{(+8)}$ | $50_{(+4)}$ | $68_{(+5)}$ | $64_{(+2)}$ | $54_{(+5)}$ |
| Llama2-13b | $31_{(+1)}$ | $7_{(+5)}$ | $49_{(+12)}$ | $43.1_{(+5.9)}$ | $69_{(+4.77)}$ | $61_{(+7)}$ | $53_{(+8.49)}$ |
| Llama2-13b-chat | $41_{(+4)}$ | $10_{(+1)}$ | $55_{(+8)}$ | $56_{(+6)}$ | $79_{(+9)}$ | $72_{(+2)}$ | $62_{(+6)}$ |
| Llama3-8b | $57_{(+6)}$ | $16.6_{(+2.4)}$ | $74_{(+6)}$ | $75.8_{(+7.2)}$ | $94_{(-0.45)}$ | $77_{(+1.2)}$ | $72_{(+13)}$ |
| Llama3-8b-Instruct | $52_{(+5)}$ | $27_{(+22)}$ | $72_{(+4)}$ | $80_{(+1)}$ | $89_{(+1)}$ | $72_{(+1)}$ | $77_{(+7)}$ |
| Llama3.1-8b-Instruct | $72_{(+11.63)}$ | $50_{(+4)}$ | $85_{(+3)}$ | $86_{(+4)}$ | $95_{(+2)}$ | $85_{(+3)}$ | $89_{(+2)}$ |

Table 1: Performance comparison with multiple LLMs debate method. The values in "()" indicate the performance of our method compared with that of our method. "+" indicates that our method is higher than that of the multiple LLMs debate method, and "−" indicates that our method is lower than that of the multiple LLMs debate method.

| model \ dataset | GSM | MATH | ASDiv | SVAMP | MultiArith | SingleEQ | AddSub |
|---|---|---|---|---|---|---|---|
| MLC(Llama2-7b-chat) | 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 |
| 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 2: The ablation studies of our method.

### 4.4 Competing Methods

To validate the efficacy of our proposed method, we compared it with the Multi-Agent Debate (MAD) method. This alternative methodology avoids fine-tuning the models and instead utilizes multiple LLMs to engage in iterative conversational discussions and debates. In the experiment, due to limitations in the number of Llama input tokens, three LLMs were set up to conduct two rounds of debate, with the historical conversation from the previous round provided as input each time. The multiple LLMs performance is shown in Table 1.

### 4.5 Overall Performance

MLC uses 6 base models from the Llama series, augmented with a basic Chain-of-Thought (CoT) prompting technique. This method instructs the LLMs to engage in step-by-step reasoning. Compared with the MAD method, MLC has the largest performance improvement on the 6 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%, and Llama3.1-8b-Instruct improves by 4.23%. Compared with the MAD method on 7 datasets, the largest improvements are on AddSub and ASDiv, which are 6.91% and 6.83%, respectively, 5.44% on GSM, 5.9% on MATH, 4.48% on SVAMP, and 2.7% on MultiArith.

In summary, the proposed method (MLC) performs better than the MAD methods on these data

| 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 |
| | MLC | **33** | **5** | **56** | **54** | **73** | **66** | **59** |
| | 13b | 24.3 | 6.3 | 45.18 | 41.6 | 56.83 | 58.46 | 58.99 |
| | MLC | **32** | **12** | **61** | **49** | **73.77** | **68** | **61.49** |
| | 13b-chat | 42 | 10 | 45 | 52 | 75 | 65.06 | 60 |
| | MLC | **45** | **11** | **63** | **62** | **88** | **74** | **68** |
| Llama-3 | 8b | 57.2 | 33 | 67 | 71 | 89.91 | 68 | 74.44 |
| | MLC | **63** | 19 | **80** | **83** | **93.55** | **78.2** | **85** |
| | 8B-Instruct | 42 | 29.1 | 64 | 77 | 82.73 | 66.93 | 62 |
| | MLC | **57** | **49** | **76** | **81** | **90** | **73** | **84** |
| Llama-3.1 | 8B-Instruct | 84.5 | 51.9 | 71 | 81 | 92 | 76 | 87 |
| | MLC | 83.63 | **54** | **88** | **90** | **97** | **88** | **91** |

Table 3: Performance comparison with single LLM method. The bold numbers refer to the best performance among all the models.

in most cases. The MAD performance is shown in Table 1. In 42 scenarios consisting of 7 datasets and 6 base models, our method is much higher in 41 scenarios, and slightly lower than 0.45% in 1 scenario. The average accuracy of MLC is 8.23% higher. 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 six base models. The results, shown in Table 2, demonstrate significant improvements across six datasets. 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 is proved to be critical by enhancing differentiation among LLM behaviors. Overall, the MLC significantly improves multi-agent performance in collaborative tasks.

### 4.7 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 3. In 42 scenarios consisting of 7 datasets and 6 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.

## 5 Conclusion

For the mathematical reasoning task, we propose a Multi-LLM Cooperation 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 (i.e. LLama2-7b, LLama3-8b) excels in baselines over 7 benchmarks.

## 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*.

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*.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. 2024. Self-refine: Iterative refinement with self-feedback. *Advances in Neural Information Processing Systems*, 36.

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.

Tabish Rashid, Mikayel Samvelyan, Christian Schroeder De Witt, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. 2020. Monotonic value function factorisation for deep multi-agent reinforcement learning. *Journal of Machine Learning Research*, 21(178):1–51.

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*.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2024. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems*, 36.

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*.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

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.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. 2024. Tree of thoughts: Deliberate problem solving with large language models. *Advances in Neural Information Processing Systems*, 36.

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 Limitations

Reasons that affect the results: 1. It is difficult to solve multi-step reasoning of complex mathematical problems directly using the question-and-answer method; 2. There is a forgetfulness problem in large model dialogues; 3. The final large model response has limited processing methods for obtaining answers, and the format of the answers is easily irregular.

## A.3 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.4 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.