

# MasRouter: Learning to Route LLMs for Multi-Agent System

Yanwei Yue<sup>★†</sup> Guibin Zhang<sup>★†</sup> Boyang Liu<sup>★†</sup> Guancheng Wan<sup>\*</sup>

Kun Wang<sup>◆</sup> Dawei Cheng<sup>★</sup> Yiyuan Qi<sup>◆</sup>

<sup>★</sup>Tongji University <sup>◆</sup>UCLA <sup>◆</sup>NTU <sup>◆</sup>IDEA

✉ Corresponding: wang.kun@ntu.edu.sg, qyy0180@gmail.com

## Abstract

Multi-agent systems (MAS) powered by Large Language Models (LLMs) have been demonstrated to push the boundaries of LLM capabilities, yet they often incur significant costs and face challenges in dynamic LLM selection. Current LLM routing methods effectively reduce overhead in single-agent scenarios by customizing LLM selection for each query, but they overlook the critical decisions regarding collaboration modes and agent roles in MAS. In response to this challenge, we first introduce the problem of **Multi-Agent System Routing (MASR)**, which integrates all components of MAS into a unified routing framework. Toward this goal, we propose **MasRouter**, the first high-performing, cost-effective, and inductive **MASR** solution. **MasRouter** employs collaboration mode determination, role allocation, and LLM routing through a cascaded controller network, progressively constructing a MAS that balances effectiveness and efficiency. Extensive experiments demonstrate that **MasRouter** is (1) **high-performing**, achieving a 1.8%  $\sim$  8.2% improvement over the state-of-the-art method on MBPP; (2) **economical**, reducing overhead by up to 52.07% compared to SOTA methods on HumanEval; and (3) **plug-and-play**, seamlessly integrating with mainstream MAS frameworks, reducing overhead by 17.21%  $\sim$  28.17% via customized routing. The code is available at <https://github.com/yanweiyue/masrouter>.

## 1 Introduction

Recent advances in Large Language Model(LLM) based agents (Park et al., 2023; Yao et al., 2023; Richards and et al., 2023) have demonstrated remarkable success across various tasks, including code generation (Wu et al., 2023; Guo et al., 2024), mathematical reasoning (Swan et al., 2023; Yu et al., 2024; Yan et al., 2025), and embodied actions (Wang et al., 2023; tse Huang et al., 2025a).

<sup>†</sup>These authors contributed equally.

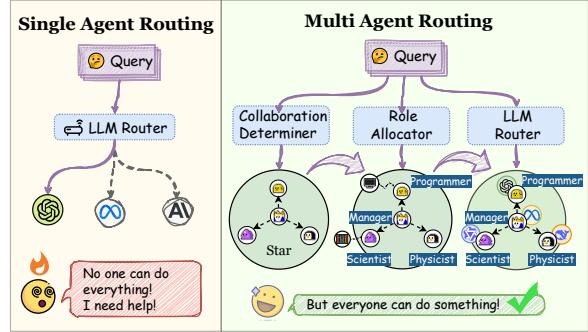


Figure 1: Paradigm comparison between single-agent routing and multi-agent routing.

Building on the impressive capabilities of single agents, the LLM-based Multi-Agent System (MAS) has been proposed (Park et al., 2023; Du et al., 2023; Hong et al., 2023; Wang et al., 2024a; Zhou et al., 2025) to harness the collective intelligence and specialized expertise of multiple agents. As the pool of available LLMs continues to grow (Chang et al., 2023; Minaee et al., 2024), how to select the appropriate LLM to power single agents or MAS has increasingly captured the attention of the research community (Hu et al., 2024a; akota et al., 2024). One might assume that, disregarding cost, choosing the largest LLM would always yield the best performance. However, larger LLMs have been shown not to always outperform their smaller counterparts (Abdin et al., 2024; Lepagnol et al., 2024; Shen et al., 2024). At times, smaller LLMs can achieve superior performance at a lower computational cost. Given this context, how to dynamically route the most appropriate LLM to empower an agent in a query-aware manner emerges as a highly compelling problem.

To address this, *LLM routing* is proposed to intelligently assign the optimal LLM for each query (Hu et al., 2024a; Srivatsa et al., 2024). Early attempts employ a router, typically based on encoder models like BERT (Devlin et al., 2019), to make a binary decision on whether to select a larger LLM (Chen et al., 2023a; Ding et al., 2024; Ong et al., 2024).

More recent practices construct a router to assess the performance and cost of multiple LLMs, subsequently selecting the model that optimizes the trade-off between exploration and exploitation (Dai et al., 2024; Mohammadshahi et al., 2024; Feng et al., 2024). Although existing LLM routing methods (Hu et al., 2024a; Stripelis et al., 2024) have been proven effective in directing the most appropriate LLM for input queries, they are limited to single-agent scenarios (Wei et al., 2022; Shinn et al., 2023; Reworkd, 2023) yet not ready for MAS. However, we argue that the availability of a routing method for MAS is even more essential. On the *performance* perspective, multi-agent systems have been proven to outperform single-agent approaches significantly (Chen et al., 2023b; Pina et al., 2023; Jin et al., 2025). On the *cost* perspective, though impressive in performance, existing multi-agent pipelines inherently introduce substantial token overhead and increased economic costs (Zhuge et al., 2024; Wang et al.), which further necessitates the use of routing to mitigate overhead. In this context, there is a critical need to address this gap: *How can we effectively route LLMs for MAS to balance performance and costs?*

A seemingly intuitive solution is to transfer single-agent LLM routing methods to MAS directly. However, MAS needs to select the appropriate agent collaboration modes (Du et al., 2023; Zhang et al., 2024c; Wang et al., 2025a; tse Huang et al., 2025b) for different tasks and establish a reasonable division of labor (Hong et al., 2023; Zhang et al., 2023), which previous routing methods fail to achieve. For example, for software development tasks, an ideal MAS routing method could design a hierarchical workflow with sequential steps such as requirements analysis, algorithm design, code development, and testing, each requiring corresponding role profiles (Ramin et al., 2020; Zingg et al., 2023). Against this backdrop, we argue that routing in MAS involves more tasks than just LLM recommendations:

- ① Collaboration Mode Determination:** Choosing the optimal communication mechanisms (*e.g.*, Chain (Qian et al., 2023), Tree (Ishibashi and Nishimura, 2024), Graph (Hao et al., 2023)) for varying task complexities. This involves identifying the most efficient and adaptable multi-agent topology (Zhuge et al., 2024; Zhang et al., 2024a) that minimizes overhead while ensuring flexibility and scalability in more complex scenarios.
- ② Dynamic Agent Number:** Determining the number of expert agents required (Huang

et al., 2024; Aghdam et al., 2024) based on the difficulty of the input.

- ③ Agent Role Allocation:** Selecting suitable role to the agent according to the query domain (Chen et al., 2023b; Feng et al., 2024) to ensure efficient task division, creating a system greater than the sum of its parts (Shang et al., 2024).
- ④ Agent LLM Routing:** Assigning each agent the appropriate LLM based on the collaborative topology and the role of each LLM (Feng et al., 2024).

In light of the scrutinizing challenges, we for the first time introduce the concept of **LLM-based Multi-Agent System Routing (MASR)**:

**Multi-Agent Systems Routing (MASR):** Given a pool of available LLMs, collaborative communication modes, and possible agent roles, an optimal MAS Router for any query  $q$  should: (1) identify appropriate multi-agent collaboration modes, (2) allocate agent roles efficiently, and (3) assign the appropriate LLM to each agent, thereby balancing performance and cost.

The formal definition of MASR is provided in Section 3. To construct a router that ideally adheres to the MASR principles, we propose an effective, token-economical, and inductive LLM-powered Multi-Agent System Router, termed **MasRouter**. Technically, **MasRouter** integrates collaboration mode determiner, agent role allocator, and agent LLM router into a unified routing framework: **① Collaboration determiner** employs a variational latent variable model to route the user query to a suitable collaboration module; **② Role allocator** progressively generates agent roles through a structured probabilistic cascade; **③ LLM router** models the LLM backbone recommendation for each agent as a multinomial distribution problem. Ultimately, **MasRouter** constructs a MAS that simultaneously **balances effectiveness and efficiency**.

Our contributions can be summarized as follows:

- **Problem Definition.** We for the first time formally define Multi-Agent System Routing (MASR), which specifies the requirements for MAS routing: assigning the appropriate collaboration mode, agent roles, and LLMs to each query, thereby improving response quality and reducing unnecessary overhead.
- **Practical Solution.** We propose **MasRouter**, a modular MASR solution that utilizes a cascaded controller network to construct a high-performing and resource-efficient MAS progressively. Besides, **MasRouter** can seamlessly inte-

grate with mainstream MAS to achieve efficient routing with significantly lower inference cost.

- **Experimental Validation.** Extensive experiments across five benchmarks show that **MasRouter** is: **(I) high-performing**, surpassing RouterDC, the state-of-the-art routing method, by 3.51% on average; **(II) economical**, reducing the overhead on HumanEval from 0.363\$ to 0.185\$; **(III) inducive and plug-and-play**, generalizing to unseen LLM backbones and collaboration modes and seamlessly combining with mainstream multi-agent systems with 17%  $\sim$  28% ↓ fewer cost.

## 2 Related Work

**Multi-Agent System.** Contemporary LLM-based multi-agent systems (MAS) can be broadly categorized into two paradigms: **(1) Fixed agentic networks** with pre-established, manually crafted architectures, from debate (Chan et al., 2023; Liang et al., 2023), collaboration (Yin et al., 2023; Wang et al., 2024c) to competitive (Zhao et al., 2023). MacNet (Qian et al., 2024) systematically analyzed typical multi-agent collaboration topologies such as chain, tree, graph, etc.; **(2) Dynamic agentic networks** that configure their structure and communication strategies based on real-time feedback and observations. ADAS (Hu et al., 2024b) and its follow-up works (Zhang et al., 2024c; Shang et al., 2024) leverage search methods such as Monte Carlo Tree Search (MCTS) and evolutionary algorithm (Zhang et al., 2025) to discover effective agent strategies. Other works like DyLAN (Liu et al., 2023), GPTSwarm (Zhuge et al., 2024), AgentPrune (Zhang et al., 2024a) and AgentDropout (Wang et al., 2025b) dynamically optimize the inter-agent topologies. Nevertheless, contemporary MAS is often LLM-homogeneous, i.e., relying exclusively on the same LLM backbone, failing to collectively organize heterogenous LLM-agents.

**Single LLM Routing** Efficient routing strategies for single LLMs have been extensively explored to balance computational cost and model performance. Early attempts on LLM routing include HybridLLM (Ding et al., 2024), RouteLLM (Ong et al., 2024) and FrugalGPT (Chen et al., 2023a), which primarily focus on binary routing and leverage techniques like sequential pipelines or preference-driven routing to enhance decision-making performance. More recent practices, including Rootoo (Dai et al., 2024), C2MAB-V (Mohammadi et al., 2024), GraphRouter (Feng

et al., 2024) and RouterDC (Chen et al., 2024) have shifted towards multi-choice selection frameworks. However, existing routing methodologies mainly focus on single-agent scenarios, and their unawareness of inter-agent topology constrains their applicability to more complex tasks and limits scalability in larger systems.

## 3 Formalization

In this section, we formalize our proposed LLM-based **Multi-Agent System Routing (MASR)** and introduce its optimization objectives.

### 3.1 Notation Establishment

**Search Space.** We first define the search space of a multi-agent system as  $\mathbb{S} = (\mathbb{M}, \mathbb{R}, \mathbb{T})$ , where  $\mathbb{M}$  denotes the pool of  $N_m$  available LLM backbones,  $\mathbb{R}$  represents the set of  $N_r$  predefined agent roles (e.g., analyst, programmer, and tester), and  $\mathbb{T}$  denotes the set of  $N_t$  collaboration modes, including structures like Chain, Tree, LLM-Debate, etc. Within the search space, a multi-agent system instance is defined as follows:

**Definition 1 (Multi-Agent System)** *The MAS  $\mathcal{S}$  combines several specific LLM-powered agents with distinct identities, working together collaboratively:*

$$\mathcal{S} = \{\{\mathcal{M}_i\}_{i=1}^k, \{\mathcal{R}_i\}_{i=1}^k, \mathcal{T}\}, \quad (1)$$

$$\mathcal{M}_i \in \mathbb{M}, \mathcal{R}_i \in \mathbb{R}, \mathcal{T} \in \mathbb{T},$$

where  $\mathcal{M}$  corresponds to the selected LLM backbones and, similarly,  $\mathcal{R}$  and  $\mathcal{T}$  represent the chosen role and collaboration mode respectively.  $k$  denotes the number of the LLM agents.

### 3.2 Definition of MASR

Based on the MAS defined above, we formalize Multi-Agent Systems Routing (MASR):

**Definition 2 (MASR)** *MASR can be represented by the mapping function  $f$ , which maps from  $\mathbb{S} = (\mathbb{M}, \mathbb{R}, \mathbb{T})$  to a MAS  $\mathcal{S}$  tailored for the query  $\mathcal{Q}$ :*

$$f : \mathbb{M} \times \mathbb{R} \times \mathbb{T} \rightarrow \mathcal{S},$$

$$\pi(\mathcal{S}) = \mathbb{P} \left( \left\{ \{\mathcal{M}_i\}_{i=1}^k, \{\mathcal{R}_i\}_{i=1}^k, \mathcal{T} \right\} \mid \mathcal{Q} \right), \quad (2)$$

$$\mathcal{M}_i \in \mathbb{M}, \mathcal{R}_i \in \mathbb{R}, \mathcal{T} \in \mathbb{T}, \mathcal{S} \subset \mathbb{S}$$

where  $\pi(\mathcal{S})$  represents the probability of selecting multi-agent system  $S$ , conditioned on  $\mathcal{Q}$ .

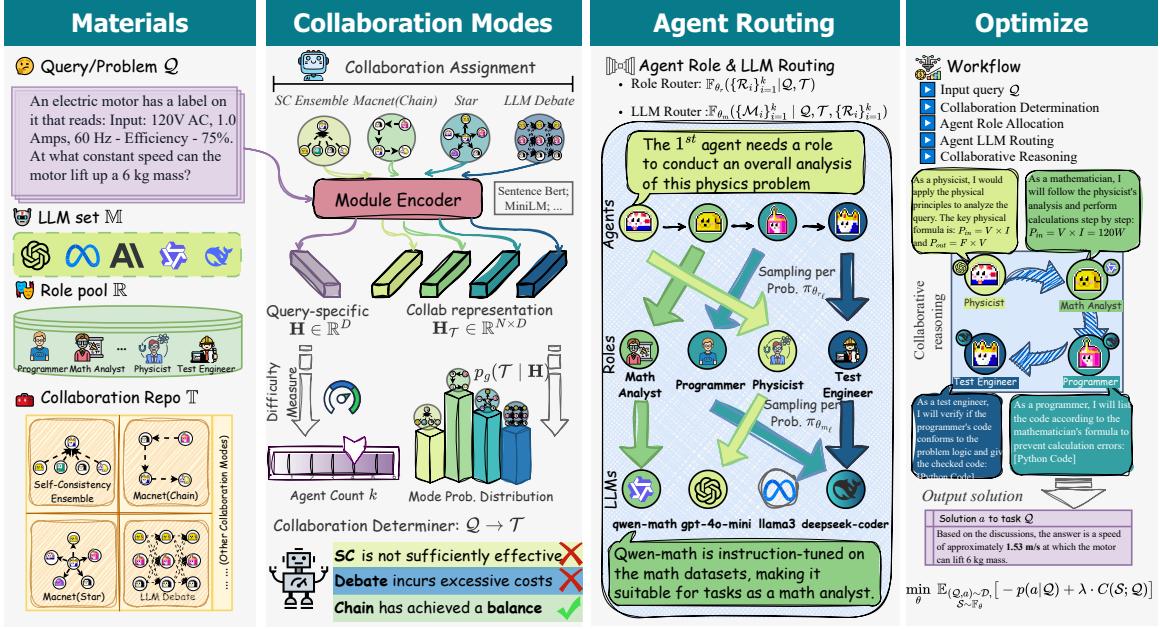


Figure 2: The overall framework of our proposed **MasRouter**.

**Optimization Objective.** Given a benchmark  $\mathcal{D}$  consisting of multiple queries  $\mathcal{Q}$  and corresponding oracle answers  $a$ , the ideal **MASR** aims to optimize a strategy to jointly balance performance and cost. The objective function is defined as:

$$\max_{\mathbb{P}(\mathcal{S}|\mathcal{Q})} \mathbb{E}_{(\mathcal{Q}, a) \sim \mathcal{D}, \mathcal{S} \in \mathbb{S} \sim \mathbb{P}(\mathcal{S}|\mathcal{Q})} \left[ \underbrace{U(\mathcal{S}; \mathcal{Q}, a)}_{\text{Utility}} - \lambda \cdot \underbrace{C(\mathcal{S}; \mathcal{Q})}_{\text{Cost}} \right], \quad (3)$$

where  $\mathbb{P}(\mathcal{S}|\mathcal{Q})$  represents the probability distribution of  $\mathcal{S}$  conditioned on  $\mathcal{Q}$ . The utility term  $U(\mathcal{S}; q, a)$  measures the performance of the MAS, while the cost term  $C(\mathcal{S}; q)$  quantifies the expected cost (e.g., LLM calls, API cost, token cost).  $\lambda$  is a trade-off parameter.

## 4 MasRouter

Figure 2 illustrates the overall framework of **MasRouter**. For a given query, **MasRouter** samples the customized components of the MAS, through the collaboration determiner (▷ Section 4.1), role allocator (▷ Section 4.2), and agent LLM router (▷ Section 4.3), together forming a task-adaptive MAS  $\mathcal{S}$ . After executing the sampled MAS, **MasRouter** jointly optimizes the parameters of each selection module based on the performance/cost feedback (▷ Section 4.4).

### 4.1 Collaboration Mode Determination

Given a query  $\mathcal{Q}$ , the core objective of **MasRouter** is to customize an appropriate MAS from the

search space  $\mathbb{S}$  based on the complexity and domain of the query, thereby generating a sufficiently high-quality response:

$$p(a|\mathcal{Q}) = \int \mathcal{O}(a|\mathcal{S}) \mathbb{F}_\theta(\mathcal{S}|\mathcal{Q}) d\mathcal{S}, \mathcal{S} \subset \mathbb{S}, \quad (4)$$

where  $\mathbb{F}_\theta$  represents the controller network parameterized by  $\theta$ , which takes  $\mathcal{Q}$  and computes the underlying distribution of  $\mathcal{S}$ .  $\mathcal{O}(\cdot|\cdot)$  denotes the conditional likelihood of obtaining the solution  $a$  by executing  $\mathcal{S}$ .  $\mathbb{F}_\theta$  is formulated as follows:

$$\mathbb{F}_\theta = \mathbb{F}_{\theta_m} \circ \mathbb{F}_{\theta_r} \circ \mathbb{F}_{\theta_t}, \quad (5)$$

where  $\mathbb{F}_{\theta_t} : \mathcal{Q} \rightarrow \mathcal{T}$  is the collaboration mode determiner,  $\mathbb{F}_{\theta_r} : (\mathcal{Q}, \mathcal{T}) \rightarrow \{\mathcal{R}_i\}_{i=1}^k$  denotes the role allocator, and  $\mathbb{F}_{\theta_m} : (\mathcal{Q}, \mathcal{T}, \{\mathcal{R}_i\}_{i=1}^k) \rightarrow \{\mathcal{M}_i\}_{i=1}^k$  represent the LLM backbone router. Inspired by human collaboration (Woolley et al., 2010; Chen et al., 2023b), **MasRouter** first constructs a team management framework for the MAS using  $\mathbb{F}_{\theta_t}$ , then recruits suitable talents and defines the division of tasks using  $\mathbb{F}_{\theta_r}$ , and finally endows each agent with the unique intelligence through  $\mathbb{F}_{\theta_m}$ . For  $\mathbb{F}_{\theta_t}$ , since the relationship between collaborative modes and queries is generally difficult to characterize explicitly, we employ a variational latent model to capture their underlying semantic associations:

$$\mathbb{F}_{\theta_t}(\mathcal{T} | \mathcal{Q}) = \int p_g(\mathcal{T} | \mathbf{H}) p_h(\mathbf{H} | \mathcal{Q}) d\mathbf{H}, \mathcal{T} \in \mathbb{T} \quad (6)$$

where  $p_h(\mathbf{H} | \mathcal{Q})$  denotes the prior probability distribution of the latent representation, and  $p_g(\mathcal{T} | \mathbf{H})$  decodes the probability of collaborative patterns, implemented as follows:

$$\begin{aligned} p_h(\mathbf{H} | \mathcal{Q}) &= \mathcal{N}(\mathbf{H}; \mu_t(\mathcal{Q}), \text{diag}(\sigma_t^2(\mathcal{Q}))), \\ p_g(\mathcal{T} | \mathbf{H}) &\propto \exp\left(\frac{\mathbf{H}^\top \mathbf{H}_\mathcal{T}}{\tau}\right), \tau > 0 \end{aligned} \quad (7)$$

where  $\mu_t(\cdot)$  and  $\sigma_t^2(\cdot)$  obtains the mean and variance of  $\mathbf{H}$ , respectively, and  $\mathbf{H}_\mathcal{T} = g_\phi(f_\psi(\mathcal{T}), \mathbf{H})$  embeds the relationship between the query and the collaborative patterns into the latent space. Here,  $f_\psi : \mathcal{Q} \rightarrow \mathbb{R}^D$  is a text encoder (*e.g.*, Sentence-BERT (Reimers, 2019), MiniLM (Wang et al., 2020)) that extracts the semantic information of the query.  $g_\phi : \mathbb{R}^D \times \mathbb{R}^D \rightarrow \mathbb{R}^D$  produces the refined representation of the candidate  $\mathcal{T}$ .

With  $\mathbb{F}_{\theta_t}$ , we have now customized the collaborative patterns for  $\mathcal{Q}$ . Nevertheless, the number of agents utilized in  $\mathcal{T}$  remains undetermined. We leverage the hidden embedding of the query to derive the number of agents  $k = \lceil \delta(\mathbf{H}) \cdot \gamma \rceil$ , where  $\delta : \mathbb{R}^D \rightarrow [0, 1]$  is a learnable complexity mapping function, and  $\gamma$  is the hyperparameter representing the maximum number of agents.

## 4.2 Agent Role Allocation

For a MAS  $\mathcal{S} = \{\{\mathcal{M}_i\}_{i=1}^k, \{\mathcal{R}_i\}_{i=1}^k, \mathcal{T}\}$ , Section 4.1 has specified  $\mathcal{T}$  and  $k$ . Subsequently, we will assign the appropriate role  $\mathcal{R}_i$  to each agent in  $\mathcal{S}$ . Roles among different agents often have a sequential order and interdependencies. For example, we first need a programmer to write the code, and then a test engineer to validate and debug it. Correspondingly, the **role allocator**  $\mathbb{F}_{\theta_r}$  formalizes role generation through a structured probabilistic cascade:

$$\mathbb{F}_{\theta_r}(\{\mathcal{R}_i\}_{i=1}^k | \mathcal{Q}, \mathcal{T}) = \prod_{\ell=1}^k \pi_{r_\ell}(\mathcal{R}_\ell | \mathcal{Q}, \{\mathcal{R}_j\}_{j=1}^{\ell-1}, \mathcal{T}), \quad (8)$$

where  $\pi_{r_\ell}$  denotes the probability of generating the  $\ell$ -th role is based on  $\mathcal{Q}$ , the selected  $\mathcal{T}$ , and the prior  $\ell - 1$  role profiles. We iteratively compute it as follows:

$$\pi_{r_\ell}(\mathcal{R}_\ell | \mathcal{Q}, \mathcal{T}, \{\mathcal{R}_j\}_{j=1}^{\ell-1}) \propto \exp\left(\frac{\tilde{\mathbf{H}}_{\mathcal{R}_{\ell-1}}^\top \mathbf{H}_{\mathcal{R}_\ell}}{\tau}\right), \quad (9)$$

$$R_\ell \in \mathbb{R}, \tau > 0,$$

where  $\tilde{\mathbf{H}}_{\mathcal{R}_\ell} = \text{FFN}(\mathbf{H} \| \mathbf{H}_\mathcal{T} \| \text{Norm}(\sum_{j=0}^{\ell-1} \mathbf{H}_{\mathcal{R}_j}))$  denotes the implicit representation of the accumulated semantics from the role allocation process

of the first  $\ell$  roles under the  $\mathcal{Q}$  and  $\mathcal{T}$ .  $\mathbf{H}_{\mathcal{R}_\ell} = g_\phi(f_\psi(\mathcal{R}_\ell), \tilde{\mathbf{H}}_{\mathcal{R}_{\ell-1}})$  captures the dynamic features exhibited by the current candidate role within the context of the previously assigned roles. Through Equation (9), **MasRouter** progressively determines the roles for all agents in  $\mathcal{S}$ . Afterward, the remaining task is to provide each agent with its driving force by routing to an appropriate LLM backbone.

## 4.3 Agent LLM Routing

Each LLMs have its own strengths and drawbacks (Barandoni et al., 2024), and the goal of LLM routing is to leverage their unique capabilities. For example, for mathematical problems, we would opt for an LLM that is particularly proficient in mathematics or one that has been specifically fine-tuned. Therefore, we posit that assigning an LLM to a specific agent primarily depends on the task’s domain and difficulty, as well as their corresponding role. We implement  $\mathbb{F}_{\theta_m}$  by computing the probability of selecting  $\mathcal{M}_i$  based on the query and the preceding role routing. It then views the process of LLM routing for multiple agents as a multinomial distribution problem:

$$\begin{aligned} &\mathbb{F}_{\theta_m}(\{\mathcal{M}_i\}_{i=1}^k | \mathcal{Q}, \mathcal{T}, \{\mathcal{R}_i\}_{i=1}^k) \\ &= \binom{k}{n_1, n_2, \dots, n_{N_m}} \cdot \prod_{\ell=1}^{N_m} \pi_m^{n_\ell}(\mathcal{M}_\ell | \mathcal{Q}, \mathcal{T}, \{\mathcal{R}_i\}_{i=1}^k), \end{aligned} \quad (10)$$

where  $\binom{k}{n_1, n_2, \dots, n_{N_m}} = \frac{k!}{n_1! n_2! \dots n_{N_m}!}$  is the multinomial coefficient. It represents the number of ways to assign  $k$  agents to  $N_m$  different LLMs, with the  $i$ -th LLM selected  $n_i$  times.  $\pi_m$  denotes the probability of each LLM being selected in the global context:

$$\pi_m(\mathcal{M}_\ell | \mathcal{Q}, \mathcal{T}, \{\mathcal{R}_i\}_{i=1}^k) \propto \exp\left(\frac{\tilde{\mathbf{H}}_{\mathcal{M}_\ell}^\top \mathbf{H}_{\mathcal{M}_\ell}}{\tau}\right), \quad (11)$$

where  $\tilde{\mathbf{H}}_{\mathcal{M}} = \text{FFN}(\mathbf{H} \| \mathbf{H}_\mathcal{T} \| \text{Norm}(\sum_{j=1}^k \mathbf{H}_j))$  aggregates the embedding of the query, collaborative patterns, and selected roles.  $\mathbf{H}_{\mathcal{M}_\ell} = g_\phi(f_\psi(\mathcal{M}_\ell), \tilde{\mathbf{H}}_{\mathcal{M}})$  computes the latent representation of each LLM. Based on  $\tilde{\mathbf{H}}_{\mathcal{M}}$  and  $\mathbf{H}_{\mathcal{M}_\ell}$ , the compatibility between each LLM and the constructed system is obtained, which is proportional to the probability of selecting  $\mathcal{M}_\ell$ .

As stated above, we have customized a MAS for the query. Only one final hurdle is left before achieving end-to-end training: the number of agents  $k$  becomes non-differentiable due to the rounding operation. To ensure smooth gradient

flow, we replace  $k$  with its pre-rounded floating-point value and approximate the multinomial coefficient in Equation (10) as follows:

$${k \choose n_1, n_2, \dots, n_{N_M}} \approx \frac{\Gamma(\delta(\mathbf{H}) \cdot \gamma + 1)}{\Gamma(n_1 + 1)\Gamma(n_2 + 1)\dots\Gamma(n_{N_M} + 1)}, \quad (12)$$

where  $\Gamma(\cdot)$  denotes the Gamma function.

#### 4.4 Optimization

The optimization objective of **MasRouter** is presented as follows:

$$\min_{\theta} \mathbb{E}_{\substack{(\mathcal{Q}, a) \sim \mathcal{D}, \\ \mathcal{S} \sim \mathbb{F}_{\theta}}} [-p(a|\mathcal{Q}) + \lambda \cdot C(\mathcal{S}; \mathcal{Q})] \quad (13)$$

where  $C(\cdot)$  represents the cost evaluation of multi-agent systems, and  $\lambda$  is the trade-off parameter. The term  $p(a|\mathcal{Q})$  in Equation (13) corresponds to Equation (4), which was computed in the previous sections. Through this optimization objective, we balance effectiveness and efficiency by maximizing the probability of generating correct solutions while minimizing token expenditure. Then following standard approaches in multi-agent structure design (Zhuge et al., 2024; Zhang et al., 2024b), we apply policy gradient (Williams, 1992) to approximate and optimize Equation (13).

We summarize the notations in Appendix A, with the algorithmic workflow in Appendix B.

## 5 Experiments

### 5.1 Experimental Setup

**Dataset and Benchmarks** We opt for MMLU (Hendrycks et al., 2021a), GSM8K (Cobbe et al., 2021), MATH (Hendrycks et al., 2021b), HumanEval (Chen et al., 2021), MBPP (Austin et al., 2021), covering a diverse range of reasoning and problem-solving tasks. For the MATH dataset, we select 519 problems from different levels using stratified sampling.

**Baselines** We compare our method with (1) single-agent approaches, including CoT (Wei et al., 2022), ComplexCoT (Fu et al., 2022), Self-Consistency (Wang et al., 2023b); (2) fixed multi-agent topologies including Chain, Tree, and Complete Graph (formally defined in (Qian et al., 2024)), LLM-Debate (Chan et al., 2023); (3) dynamic multi-agent systems like GPTSwarm (Zhuge et al., 2024), AgentPrune (Zhang et al., 2024b) and AFlow (Zhang et al., 2024c); (4) single LLM routers including PromptLLM introduced in (Feng et al., 2024), RouteLLM (Ong et al., 2024), FrugalGPT (Chen et al., 2023a) and RouterDC (Chen et al., 2024).

**LLM Backbones** We select LLMs with varying sizes and capacities, including gpt-4o-mini-0718 (OpenAI, 2024), claude-3.5-haiku (Anthropic, 2024), gemini-1.5-flash (Team et al., 2024) and llama-3.1-70b (Dubey et al., 2024) as the lilm pool. Deepseek-v3 (DeepSeek-AI et al., 2024) is used to validate **MasRouter**'s inductive capabilities. The temperature is always set as 1.

**Implementation Details** We set the learning rate  $\alpha = 0.01$ , the temperature  $\tau = 1$ , the cost penalty  $\lambda \in \{5, 15, 25\}$  and the num of iteration  $K \in \{5, 10\}$ , the agent's maximum amount  $\gamma = 6$ . In the MAS baselines, the number of agents equals  $\gamma$ . We use all the LLMs mentioned in the LLM Backbones as the *candidate LLM pool* for the routing method. The collaboration modes repository includes CoT, Reflexion (Shinn et al., 2023; Wang et al., 2024b), self-consistency, LLM debate, and Macnet (Chain & Complete graph). The role pool comprises 26 roles with diverse capabilities, such as programmers using compilers and researchers with access to Wikipedia. The details of the candidate pools can be found in Appendix E.

### 5.2 Performance & Cost Analysis

In this section, we compare **MasRouter** with twenty baselines across five benchmarks. We verify that **MasRouter** is:

**High-performing.** The experimental results in Table 1 demonstrates that **MasRouter** excels at constructing an effective multi-agent system. Specifically, **MasRouter** achieves the best performance across all of the five datasets, outperforming RouterDC, the SOTA LLM routing method, by 3.51% on average. On the MBPP dataset, **MasRouter** outperforms AgentPrune and AFlow by 8.20% and 1.80% at pass@1, respectively.

**Token-economical.** As shown in Figure 3, **MasRouter** achieves the best performance on the Pareto front of cost-effectiveness on the MBPP dataset. Compared to AFlow, **MasRouter** not only achieves a 1.8% ~ 8.0% improvement in performance but also reduces the inference overhead by 40.22% ~ 43.78%.

**Training resource-saving.** As shown in Table 12, compared to trainable MAS pipelines like GPTSwarm and AFlow, we achieve savings of 69.57% and 83.51% on the MMLU dataset, respectively. This is because our method does not require exhaustive traversal and validation of each agentic

Method	LLM	Mul.	Rout.	MMLU	GSM8K	MATH	HumanEval	MBPP	Avg.
Vanilla	gpt-3.5-turbo	✗	✗	69.28	77.97	44.12	72.05	70.20	66.72
	gpt-4o-mini	✗	✗	77.81	93.17	66.09	85.71	72.20	79.00
	claude-3.5-haiku	✗	✗	67.97	92.16	65.89	86.33	73.40	77.15
	gemini-1.5-flash	✗	✗	80.04	92.67	74.39	82.61	73.00	80.54
	llama-3.1-70b	✗	✗	79.08	92.68	60.31	80.75	68.20	76.20
CoT (Wei et al., 2022)	gpt-4o-mini	✗	✗	78.43	93.68	67.24	86.69	69.60	79.13
ComplexCoT (Fu et al., 2022)	gemini-1.5-flash	✗	✗	81.35	92.92	74.34	81.37	73.00	80.60
SC(CoT) (Wang et al., 2023a)	gpt-4o-mini	✗	✗	81.05	93.43	67.05	87.58	75.80	80.98
SC(ComplexCoT) (Wang et al., 2023a)	gemini-1.5-flash	✗	✗	80.74	92.01	75.11	80.12	71.80	79.96
Chain (Qian et al., 2024)	gpt-4o-mini	✓	✗	82.01	94.40	64.72	85.63	75.40	80.43
Tree (Qian et al., 2024)	gemini-1.5-flash	✓	✗	83.01	93.13	72.10	82.50	73.20	80.79
Complete Graph (Qian et al., 2024)	gpt-4o-mini	✓	✗	82.98	93.89	65.11	87.50	75.60	81.02
LLM-Debate (Du et al., 2023)	gemini-1.5-flash	✓	✗	81.74	94.91	71.36	77.50	73.60	79.82
GPTSwarm (Zhuge et al., 2024)	gpt-4o-mini	✓	✗	83.06	94.66	67.63	85.00	75.20	81.11
Agentprune (Zhang et al., 2024b)	gemini-1.5-flash	✓	✗	81.35	94.40	68.60	83.75	74.20	80.46
AFlow (Zhang et al., 2024c)	gpt-4o-mini	✓	✗	81.04	94.66	64.68	84.38	73.60	79.67
PromptLLM (Feng et al., 2024)	gemini-1.5-flash	✓	✗	80.40	93.98	72.45	79.38	73.40	79.92
RouteLLM (Ong et al., 2024)	gpt-4o-mini	✓	✗	82.80	94.66	68.85	86.28	75.40	81.60
FragalGPT (Chen et al., 2023a)	gemini-1.5-flash	✓	✗	83.22	93.98	73.35	82.36	74.80	81.54
RouterDC (Chen et al., 2024)	gpt-4o-mini	✓	✗	83.02	94.89	68.45	86.80	75.40	81.71
MasRouter (Ours)	gemini-1.5-flash	✓	✗	83.10	93.88	73.54	82.55	75.80	81.77
	gpt-4o-mini	✓	✗	83.10	92.30	73.35	90.06	82.20	84.20
	gemini-1.5-flash	✓	✗	82.35	94.91	72.70	85.69	76.00	82.33
	LLM Pool	✗	✓	78.43	93.92	73.03	86.33	73.60	81.06
	LLM Pool	✗	✓	81.04	93.42	71.29	83.85	72.60	80.44
	LLM Pool	✗	✓	76.24	90.76	67.05	87.31	74.40	79.15
	LLM Pool	✗	✓	82.01	93.68	73.46	87.75	75.20	82.42
	LLM Pool	✓	✓	<b>84.25</b>	<b>95.45</b>	<b>75.42</b>	<b>90.62</b>	<b>84.00</b>	<b>85.93</b>

Table 1: Performance comparison with vanilla, single agent, multi-agent, and single-agent routing methods. The best results are highlighted in bold, and the runner-ups are underlined. The LLM pool includes the economical and advanced LLMs mentioned in Section 5.1. "Mul." and "Rout." indicate whether the method supports a multi-agent setting and whether it supports the LLM routing, respectively. ✗ and ✓ indicate whether these features are supported.

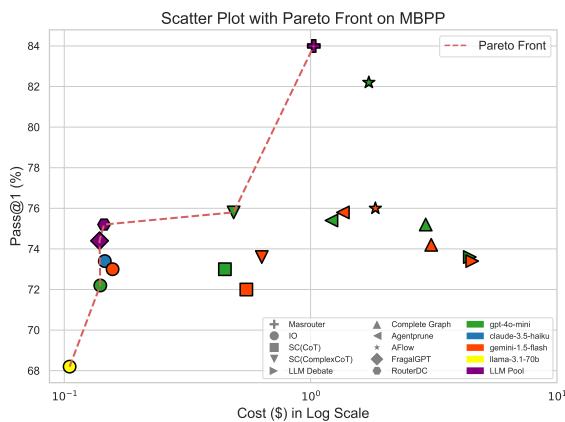


Figure 3: The comparison of the performance and inference cost on the MBPP dataset. Different shapes of the scatter points represent various types of baselines, while the different colors of the points indicate the use of different LLM backbones.

structure. We present the detailed cost-performance data in Appendix D.

### 5.3 Plug-in to Existing MAS

Considering that contemporary MAS is often LLM-homogeneous, *i.e.*, relying exclusively on a single powerful model like gpt-4o, **MasRouter** can serve as a plug-and-play solution, seamlessly as-

Dataset	Method	LLM	Performance	Cost
MMLU	MAD	gpt	81.50	\$25.56
		gemini	80.94	\$27.02
	+MasRouter		<u>82.20</u> (↑ 0.70)	\$19.39
HumanEval	MAD	gpt	86.05	\$1.248
		gemini	82.95	\$1.526
	+MasRouter		<u>87.60</u> (↑ 1.55)	\$1.096
GSM8K	MAD	gpt	94.60	\$5.664
		gemini	94.40	\$5.492
	+MasRouter		<u>94.91</u> (↑ 0.31)	\$4.702
MMLU	MacNet	gpt	82.98	\$7.812
		gemini	81.74	\$8.482
	+MasRouter		<u>83.40</u> (↑ 0.36)	\$5.892
HumanEval	MacNet	gpt	86.82	\$0.488
		gemini	83.72	\$0.568
	+MasRouter		<u>88.37</u> (↑ 1.55)	\$0.404
GSM8K	MacNet	gpt	94.69	\$2.142
		gemini	94.31	\$2.016
	+MasRouter		<u>94.89</u> (↑ 0.20)	\$1.774

Table 2: Comparison of performance and cost before and after integrating with **MasRouter**. gpt and gemini are abbreviations for gpt-4o-mini and gemini-1.5-flash, respectively. The MacNet method uses the optimal structure reported in the paper.

signing an optimal LLM backbone to each agent within them, resulting in significantly less inference cost and comparable performance. In Table 2, we combine **MasRouter** with the well-established

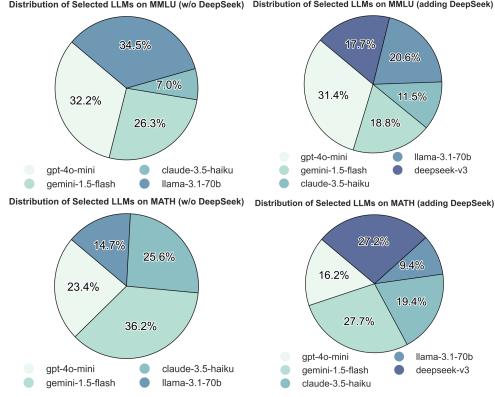


Figure 4: The selected LLM distribution of **MasRouter** on MMLU and MATH benchmark.

MAS methods MAD (Du et al., 2023) and MacNet. **MasRouter** improves the performance of MAD by 1.55% at pass@1 on the HumanEval dataset while reducing cost by 17.21% ~ 28.17%. On larger datasets, the significant reduction in overhead is even more notable; when integrated with MAD, **MasRouter** saves the inference cost by 6.17 ~ 7.63\$ on MMLU. Overall, **MasRouter** can serve as a plugin to support economical multi-agent development.

#### 5.4 Inductive Ability Analysis

In this section, we validate that **MasRouter** can easily generalize to unseen LLMs without intensive re-training resources. Figure 4 illustrates the distribution of LLMs selected by **MasRouter** before and after the addition of Deepseek-v3 on MMLU and MATH, with the new model being chosen 12.17% and 27.19% the time, respectively. By intelligently selecting the new, stronger model, **MasRouter** improved the accuracy on MMLU from 84.25% to 85.40% and also improved the accuracy on HumanEval from 90.62% to 91.41%.

Dataset	GSM8K		MATH		
	Metric	Accuracy (%)	Cost (\$)	Accuracy (%)	Cost (\$)
Vanilla <b>MasRouter</b>		95.45	1.59	75.42	3.58
<b>MasRouter w/o</b> $\mathbb{F}_{\theta_t}$		93.84	2.38	72.77	4.48
<b>MasRouter w/o</b> $\mathbb{F}_{\theta_r}$		94.70	1.67	73.01	3.63
<b>MasRouter w/o</b> $\mathbb{F}_{\theta_m}$		93.36	1.98	71.08	4.16
<b>MasRouter w/o</b> $C(\cdot)$		95.63	2.45	75.18	5.07

Table 3: Ablation study of **MasRouter**.

#### 5.5 Framework Analysis

**Ablation Study** We conduct an ablation study on the four key modules in **MasRouter**: (1) **w/o**  $\mathbb{F}_{\theta_t}$ , which replaces the collaboration determination  $\mathbb{F}_{\theta_t}$  with random selection, (2) **w/o**  $\mathbb{F}_{\theta_r}$ , which replaces

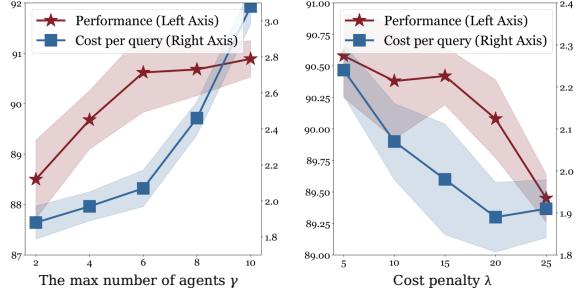


Figure 5: Sensitivity analysis of **MasRouter** on HumanEval. The unit of cost per query (right) and performance (left) is  $10^{-3} \cdot \$$  and  $pass@1 (\%)$ , respectively.

the role allocator  $\mathbb{F}_{\theta_r}$  with random selection, (3) **w/o**  $\mathbb{F}_{\theta_m}$ , which replaces the LLM router  $\mathbb{F}_{\theta_r}$  with random selection, and (4) **w/o**  $C(\cdot)$ , which remove the cost evaluation in Equation (13). As shown in Table 3, removing the  $\mathbb{F}_{\theta_m}$  results in the largest performance decline by 2.09% and 4.34%. This is because the performance of the base models on the dataset varies significantly, making selecting the appropriate LLM to solve the problem crucial. Removing  $C(\cdot)$  does not significantly impact the performance, but it disrupts the adaptive capability of **MasRouter** to query difficulty, leading to an increase in overhead by 54.09% and 41.62%.

**Sensitivity Analysis** We analyze the sensitivity of **MasRouter** to two core parameters: the maximum number of the agents  $\gamma$ , the cost penalty coefficient  $\lambda$  in Equation (13). The results are presented in Figure 5. **For the parameter**  $\gamma$ , we observe a significant performance improvement as  $\gamma$  increases from 2 to 6 (88.50% → 90.62%). However, further increases from 6 to 10 yield only marginal performance gains while incurring the 1.5× per-query inference costs. Considering both performance and cost, we select  $\gamma = 6$ . **For the parameter**  $\lambda$ , as  $\lambda$  increases from 5 to 25, we find that larger values lead **MasRouter** to favor more cost-efficient solutions, reducing the overhead by 17.78%, albeit with a slight performance degradation of approximately 1.3%. We balance effectiveness and cost by dynamically adjusting this value.

#### 5.6 Case Study

We present a detailed case study and visualization of the routing process of **MasRouter** across five benchmarks. We sincerely refer the readers to Appendix C for details.

### 6 Conclusion

In this paper, we *for the first time* introduce **Multi-Agent System Routing (MASR)**, which

aims to intelligently allocate collaboration patterns, agent roles, and LLMs for each query, thereby constructing a customized MAS. Based on this concept, we present **MasRouter**, the first high-performing, economical, and inductive MASR solution. **MasRouter** progressively builds a list of mutually adaptive roles, selects an LLM proficient in the task domain, and ultimately achieves a balance between effectiveness and efficiency. We believe **MasRouter** paves the way for the automation, economization, and scalability of MAS, contributing to the development of collective intelligence and the agent scaling law.

## Limitations

**MasRouter** is based on the assumption that each model in the LLM Pool is excellent. However, in practice, some LLMs may be attacked or poisoned. In multi-agent systems, a poisoned LLM, as a bad apple, could mislead other LLMs, significantly reducing the reasoning performance and ultimately producing misleading responses. In such cases, intuitively, **MasRouter** should identify the attacked LLM and exhibit strong robustness. This would be a promising application direction for **MasRouter**. In addition, our supplementary experiments verified that fine-tuning a large language model as the router using training data collected from **MasRouter** can further enhance its performance. However, due to time constraints, we have not provided a detailed discussion yet. We will address these issues as future research objectives.

## References

- Marah Abdin, Jyoti Aneja, Hany Awadalla, Ahmed Awadallah, Ammar Ahmad Awan, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Jianmin Bao, Harkirat Behl, Alon Benhaim, Misha Bilenko, Johan Bjorck, Sébastien Bubeck, Martin Cai, Qin Cai, Vishrav Chaudhary, Dong Chen, Dongdong Chen, Weizhu Chen, Yen-Chun Chen, Yi-Ling Chen, Hao Cheng, Parul Chopra, Xiyang Dai, Matthew Dixon, Ronen Eldan, Victor Fragoso, Jianfeng Gao, Mei Gao, Min Gao, Amit Garg, Allie Del Giorno, Abhishek Goswami, Suriya Gunasekar, Emman Haider, Jun-heng Hao, Russell J. Hewett, Wenxiang Hu, Jamie Huynh, Dan Iter, Sam Ade Jacobs, Mojgan Javaheripi, Xin Jin, Nikos Karampatziakis, Piero Kauffmann, Mahoud Khademi, Dongwoo Kim, Young Jin Kim, Lev Kurilenko, James R. Lee, Yin Tat Lee, Yuanzhi Li, Yunsheng Li, Chen Liang, Lars Liden, Xihui Lin, Zeqi Lin, Ce Liu, Liyuan Liu, Mengchen Liu, Weishung Liu, Xiaodong Liu, Chong Luo, Piyush Madan, Ali Mahmoudzadeh, David Majercak, Matt Mazzola, Caio César Teodoro Mendes, Arindam Mitra, Hardik Modi, Anh Nguyen, Brandon Norick, Barun Patra, Daniel Perez-Becker, Thomas Portet, Reid Pryzant, Heyang Qin, Marko Radmilac, Liliang Ren, Gustavo de Rosa, Corby Rosset, Sambudha Roy, Olatunji Ruwase, Olli Saarikivi, Amin Saied, Adil Salim, Michael Santacroce, Shital Shah, Ning Shang, Hiteshi Sharma, Yelong Shen, Swadheen Shukla, Xia Song, Masahiro Tanaka, Andrea Tupini, Praneetha Vaddamanu, Chunyu Wang, Guanhua Wang, Lijuan Wang, Shuhang Wang, Xin Wang, Yu Wang, Rachel Ward, Wen Wen, Philipp Witte, Haiping Wu, Xiaoxia Wu, Michael Wyatt, Bin Xiao, Can Xu, Jiahang Xu, Weijian Xu, Jilong Xue, Sonali Yadav, Fan Yang, Jianwei Yang, Yifan Yang, Ziyi Yang, Donghan Yu, Lu Yuan, Chenruidong Zhang, Cyril Zhang, Jianwen Zhang, Li Lyra Zhang, Yi Zhang, Yue Zhang, Yunan Zhang, and Xiren Zhou. 2024. *Phi-3 technical report: A highly capable language model locally on your phone*. [Preprint](#), arXiv:2404.14219.
- Maryam Akhavan Aghdam, Hongpeng Jin, and Yanzhao Wu. 2024. *Da-moe: Towards dynamic expert allocation for mixture-of-experts models*. [Preprint](#), arXiv:2409.06669.
- Marija akota, Maxime Peyrard, and Robert West. 2024. *Fly-swat or cannon? cost-effective language model choice via meta-modeling*. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining, WSDM ’24*, page 606–615. ACM.
- Anthropic. 2024. Model card addendum: Claude 3.5 haiku and upgraded claudie 3.5 sonnet. Technical report, Anthropic.
- Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. 2021. *Program synthesis with large language models*. [Preprint](#), arXiv:2108.07732.
- Simone Barandoni, Filippo Chiarello, Lorenzo Cascone, Emilio Marralle, and Salvatore Puccio. 2024. *Automating customer needs analysis: A comparative study of large language models in the travel industry*. [Preprint](#), arXiv:2404.17975.
- 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 e-prints](#).
- Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, Wei Ye, Yue Zhang, Yi Chang, Philip S. Yu, Qiang Yang, and Xing Xie. 2023. *A survey on evaluation of large language models*. [Preprint](#), arXiv:2307.03109.
- Lingjiao Chen, Matei Zaharia, and James Zou. 2023a. *Frugalgpt: How to use large language models while reducing cost and improving performance*. [Preprint](#), arXiv:2305.05176.

- Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kapolan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. 2021. Evaluating large language models trained on code.
- Shuhao Chen, Weisen Jiang, Baijiong Lin, James T Kwok, and Yu Zhang. 2024. Routerdc: Query-based router by dual contrastive learning for assembling large language models. [arXiv preprint arXiv:2409.19886](#).
- Weize Chen, Yusheng Su, Jingwei Zuo, Cheng Yang, Chenfei Yuan, Chen Qian, Chi-Min Chan, Yujia Qin, Yaxi Lu, Ruobing Xie, Zhiyuan Liu, Maosong Sun, and Jie Zhou. 2023b. Agentverse: Facilitating multi-agent collaboration and exploring emergent behaviors in agents. [Preprint, arXiv:2308.10848](#).
- Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. [arXiv preprint, abs/2110.14168](#).
- Xiangxiang Dai, Jin Li, Xutong Liu, Anqi Yu, and John C. S. Lui. 2024. Cost-effective online multi-llm selection with versatile reward models. [Preprint, arXiv:2405.16587](#).
- DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Haowei Zhang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Li, Hui Qu, J. L. Cai, Jian Liang, Jianzhong Guo, Jiaqi Ni, Jiashi Li, Jiawei Wang, Jin Chen, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, Junxiao Song, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Lei Xu, Leyi Xia, Liang Zhao, Litong Wang, Liyue Zhang, Meng Li, Miaojun Wang, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingming Li, Ning Tian, Panpan Huang, Peiyi Wang, Peng Zhang, Qiancheng Wang, Qihao Zhu, Qinyu Chen, Qiushi Du, R. J. Chen, R. L. Jin, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, Runxin Xu, Ruoyu Zhang, Ruyi Chen, S. S. Li, Shanghao Lu, Shangyan Zhou, Shanhua Chen, Shaoqing Wu, Shengfeng Ye, Shengfeng Ye, Shirong Ma, Shiyu Wang, Shuang Zhou, Shuiping Yu, Shunfeng Zhou, Shuting Pan, T. Wang, Tao Yun, Tian Pei, Tianyu Sun, W. L. Xiao, Wangding Zeng, Wanjia Zhao, Wei An, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, X. Q. Li, Xiangyue Jin, Xianzu Wang, Xiao Bi, Xiaodong Liu, Xiaohan Wang, Xiaojin Shen, Xiaokang Chen, Xiaokang Zhang, Xiaosha Chen, Xiaotao Nie, Xiaowen Sun, Xiaoxiang Wang, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xingkai Yu, Xinnan Song, Xinxia Shan, Xinyi Zhou, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, Y. K. Li, Y. Q. Wang, Y. X. Wei, Y. X. Zhu, Yang Zhang, Yan-hong Xu, Yanhong Xu, Yanping Huang, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Li, Yaohui Wang, Yi Yu, Yi Zheng, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Ying Tang, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yu Wu, Yuan Ou, Yuchen Zhu, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yukun Zha, Yunfan Xiong, Yunxian Ma, Yuting Yan, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Z. F. Wu, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhen Huang, Zhen Zhang, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhibin Gou, Zhicheng Ma, Zhi-gang Yan, Zhihong Shao, Zhipeng Xu, Zhiyu Wu, Zhongyu Zhang, Zhuoshu Li, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Ziyi Gao, and Zizheng Pan. 2024. Deepseek-v3 technical report. [Preprint, arXiv:2412.19437](#).
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. [Preprint, arXiv:1810.04805](#).
- Dujian Ding, Ankur Mallick, Chi Wang, Robert Sim, Subhabrata Mukherjee, Victor Ruhle, Laks V. S. Lakshmanan, and Ahmed Hassan Awadallah. 2024. Hybrid llm: Cost-efficient and quality-aware query routing. [Preprint, arXiv:2404.14618](#).
- Yilun Du, Shuang Li, Antonio Torralba, Joshua B. Tenenbaum, and Igor Mordatch. 2023. Improving factuality and reasoning in language models through multiagent debate. [CoRR, abs/2305.14325](#).
- Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. [arXiv preprint arXiv:2407.21783](#).
- Tao Feng, Yanzhen Shen, and Jiaxuan You. 2024. Graphrouter: A graph-based router for llm selections. [Preprint, arXiv:2410.03834](#).
- Yao Fu, Hao Peng, Ashish Sabharwal, Peter Clark, and Tushar Khot. 2022. Complexity-based prompting for multi-step reasoning. In [The Eleventh International Conference on Learning Representations](#).

- Daya Guo, Qihao Zhu, Dejian Yang, Zhenda Xie, Kai Dong, Wentao Zhang, Guanting Chen, Xiao Bi, Y. Wu, Y. K. Li, Fuli Luo, Yingfei Xiong, and Wenfeng Liang. 2024. Deepseek-coder: When the large language model meets programming – the rise of code intelligence. [Preprint](#), arXiv:2401.14196.
- Rui Hao, Linmei Hu, Weijian Qi, Qingliu Wu, Yirui Zhang, and Liqiang Nie. 2023. Chatllm network: More brains, more intelligence.
- Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021a. Measuring massive multitask language understanding. [Proceedings of the International Conference on Learning Representations \(ICLR\)](#).
- Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021b. Measuring mathematical problem solving with the math dataset. [NeurIPS](#).
- Sirui Hong, Xiawu Zheng, Jonathan Chen, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, Lingfeng Xiao, and Chenglin Wu. 2023. Metagpt: Meta programming for multi-agent collaborative framework.
- Qitian Jason Hu, Jacob Bieker, Xiuyu Li, Nan Jiang, Benjamin Keigwin, Gaurav Ranganath, Kurt Keutzer, and Shriyash Kaustubh Upadhyay. 2024a. Routerbench: A benchmark for multi-llm routing system. [Preprint](#), arXiv:2403.12031.
- Shengran Hu, Cong Lu, and Jeff Clune. 2024b. Automated Design of Agentic Systems. [arXiv preprint](#). ArXiv:2408.08435.
- Quzhe Huang, Zhenwei An, Nan Zhuang, Mingxu Tao, Chen Zhang, Yang Jin, Kun Xu, Kun Xu, Liwei Chen, Songfang Huang, and Yansong Feng. 2024. Harder tasks need more experts: Dynamic routing in moe models. [Preprint](#), arXiv:2403.07652.
- Yoichi Ishibashi and Yoshimasa Nishimura. 2024. Self-organized agents: A llm multi-agent framework toward ultra large-scale code generation and optimization. [arXiv preprint](#) arXiv:2404.02183.
- Can Jin, Hongwu Peng, Qixin Zhang, Yujin Tang, Dimitris N Metaxas, and Tong Che. 2025. Two heads are better than one: Test-time scaling of multi-agent collaborative reasoning. [arXiv preprint](#) arXiv:2504.09772.
- Pierre Lepagnol, Thomas Gerald, Sahar Ghannay, Christophe Servan, and Sophie Rosset. 2024. Small language models are good too: An empirical study of zero-shot classification. In [Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation \(LREC-COLING 2024\)](#), pages 14923–14936, Torino, Italia. ELRA and ICCL.
- 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. Work in progress.
- Zijun Liu, Yanzhe Zhang, Peng Li, Yang Liu, and Diyi Yang. 2023. Dynamic llm-agent network: An llm-agent collaboration framework with agent team optimization. [CoRR](#), abs/2310.02170.
- Shervin Minaee, Tomas Mikolov, Narjes Nikzad, Meysam Chenaghlu, Richard Socher, Xavier Amatriain, and Jianfeng Gao. 2024. Large language models: A survey. [Preprint](#), arXiv:2402.06196.
- Alireza Mohammadshahi, Arshad Rafiq Shaikh, and Majid Yazdani. 2024. Routoo: Learning to route to large language models effectively. [Preprint](#), arXiv:2401.13979.
- Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chang, Tianhao Wu, Joseph E. Gonzalez, M Waleed Kadous, and Ion Stoica. 2024. Routellm: Learning to route llms with preference data. [Preprint](#), arXiv:2406.18665.
- OpenAI. 2024. Gpt-4o mini: Advancing cost-efficient intelligence.
- Joon Sung Park, Joseph C. O'Brien, Carrie J. Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. 2023. Generative agents: Interactive simulacra of human behavior.
- Rafael Pina, Varuna De Silva, and Corentin Artaud. 2023. Discovering causality for efficient cooperation in multi-agent environments. [CoRR](#), abs/2306.11846.
- Chen Qian, Xin Cong, Cheng Yang, Weize Chen, Yusheng Su, Juyuan Xu, Zhiyuan Liu, and Maosong Sun. 2023. Communicative agents for software development. 25 pages, 9 figures, 2 tables.
- Chen Qian, Zihao Xie, Yifei Wang, Wei Liu, Yufan Dang, Zhuoyun Du, Weize Chen, Cheng Yang, Zhiyuan Liu, and Maosong Sun. 2024. Scaling large-language-model-based multi-agent collaboration. [arXiv preprint](#) arXiv:2406.07155.
- Frederike Ramin, Christoph Matthies, and Ralf Teusner. 2020. More than code: Contributions in scrum software engineering teams. In [Proceedings of the IEEE/ACM 42nd International Conference on Software Engineering Workshops, ICSE '20](#), page 137–140. ACM.
- N Reimers. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. [arXiv preprint](#) arXiv:1908.10084.
- Reworkd. 2023. Agentgpt. <https://github.com/reworkd/AgentGPT>.

- Toran Bruce Richards and et al. 2023. Auto-gpt: An autonomous gpt-4 experiment. <https://github.com/Significant-Gravitas/Auto-GPT>.
- Yu Shang, Yu Li, Keyu Zhao, Likai Ma, Jiahe Liu, Fengli Xu, and Yong Li. 2024. Agentsquare: Automatic llm agent search in modular design space. *Preprint*, arXiv:2410.06153.
- Weizhou Shen, Chenliang Li, Hongzhan Chen, Ming Yan, Xiaojun Quan, Hehong Chen, Ji Zhang, and Fei Huang. 2024. Small llms are weak tool learners: A multi-llm agent. *arXiv preprint arXiv:2401.07324*.
- Noah Shinn, Beck Labash, and Ashwin Gopinath. 2023. Reflexion: an autonomous agent with dynamic memory and self-reflection. *arXiv preprint arXiv:2303.11366*.
- KV Aditya Srivatsa, Kaushal Kumar Maurya, and Ekaterina Kochmar. 2024. Harnessing the power of multiple minds: Lessons learned from llm routing. *Preprint*, arXiv:2405.00467.
- Dimitris Stripelis, Zhaozhuo Xu, Zijian Hu, Alay Dilipbhai Shah, Han Jin, Yuhang Yao, Jipeng Zhang, Tong Zhang, Salman Avestimehr, and Chao Yang He. 2024. TensorOpera router: A multi-model router for efficient LLM inference. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 452–462, Miami, Florida, US. Association for Computational Linguistics.
- Melanie Swan, Takashi Kido, Eric Roland, and Renato P. dos Santos. 2023. Math agents: Computational infrastructure, mathematical embedding, and genomics. *Preprint*, arXiv:2307.02502.
- Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, Anmol Gulati, Garrett Tanzer, Damien Vincent, Zhufeng Pan, Shibo Wang, et al. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *Preprint*, arXiv:2403.05530.
- Jen tse Huang, Eric John Li, Man Ho Lam, Tian Liang, Wenxuan Wang, Youliang Yuan, Wenxiang Jiao, Xing Wang, Zhaopeng Tu, and Michael R. Lyu. 2025a. How far are we on the decision-making of llms? evaluating llms’ gaming ability in multi-agent environments. *Preprint*, arXiv:2403.11807.
- Jen tse Huang, Jiaxu Zhou, Tailin Jin, Xuhui Zhou, Zixi Chen, Wenxuan Wang, Youliang Yuan, Michael R. Lyu, and Maarten Sap. 2025b. On the resilience of llm-based multi-agent collaboration with faulty agents. *Preprint*, arXiv:2408.00989.
- Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. 2023. Voyager: An Open-Ended Embodied Agent with Large Language Models. *arXiv e-prints*, arXiv:2305.16291.
- Qian Wang, Zhenheng Tang, ZICHEN JIANG, Nuo Chen, Tianyu Wang, and Bingsheng He. Agenttaxo: Dissecting and benchmarking token distribution of llm multi-agent systems. In *ICLR 2025 Workshop on Foundation Models in the Wild*.
- Qian Wang, Tianyu Wang, Qinbin Li, Jingsheng Liang, and Bingsheng He. 2024a. Megaagent: A practical framework for autonomous cooperation in large-scale llm agent systems. *arXiv preprint arXiv:2408.09955*.
- Qian Wang, Jiaying Wu, Zhenheng Tang, Bingqiao Luo, Nuo Chen, Wei Chen, and Bingsheng He. 2025a. What limits llm-based human simulation: Llms or our design? *arXiv preprint arXiv:2501.08579*.
- Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *Advances in Neural Information Processing Systems*, 33:5776–5788.
- Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023a. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*.
- Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023b. Self-consistency improves chain of thought reasoning in language models. In *ICLR*. OpenReview.net.
- Yutong Wang, Jiali Zeng, Xuebo Liu, Fandong Meng, Jie Zhou, and Min Zhang. 2024b. TasTe: Teaching large language models to translate through self-reflection. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6144–6158, Bangkok, Thailand. Association for Computational Linguistics.
- Zhenhailong Wang, Shaoguang Mao, Wenshan Wu, Tao Ge, Furu Wei, and Heng Ji. 2024c. Unleashing the emergent cognitive synergy in large language models: A task-solving agent through multi-persona self-collaboration. In *NAACL*. Association for Computational Linguistics.
- Zhexuan Wang, Yutong Wang, Xuebo Liu, Liang Ding, Miao Zhang, Jie Liu, and Min Zhang. 2025b. Agentdropout: Dynamic agent elimination for token-efficient and high-performance llm-based multi-agent collaboration. *Preprint*, arXiv:2503.18891.
- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models.
- Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8:229–256.

- Anita Williams Woolley, Christopher F. Chabris, Alex Pentland, Nada Hashmi, and Thomas W. Malone. 2010. Evidence for a collective intelligence factor in the performance of human groups. *Science*, 330(6004):686–688.
- Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Shaokun Zhang, Erkang Zhu, Beibin Li, Li Jiang, Xiaoyun Zhang, and Chi Wang. 2023. Autogen: Enabling next-gen llm applications via multi-agent conversation framework.
- Yibo Yan, Shen Wang, Jiahao Huo, Philip S Yu, Xuming Hu, and Qingsong Wen. 2025. Mathagent: Leveraging a mixture-of-math-agent framework for real-world multimodal mathematical error detection. *arXiv preprint arXiv:2503.18132*.
- Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations*.
- Zhangyue Yin, Qiushi Sun, Cheng Chang, Qipeng Guo, Junqi Dai, Xuanjing Huang, and Xipeng Qiu. 2023. Exchange-of-thought: Enhancing large language model capabilities through cross-model communication. *Preprint*, arXiv:2312.01823.
- Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T. Kwok, Zhen-guo Li, Adrian Weller, and Weiyang Liu. 2024. Metamath: Bootstrap your own mathematical questions for large language models. *Preprint*, arXiv:2309.12284.
- Guibin Zhang, Kaijie Chen, Guancheng Wan, Heng Chang, Hong Cheng, Kun Wang, Shuyue Hu, and Lei Bai. 2025. Evoflow: Evolving diverse agentic workflows on the fly. *arXiv preprint arXiv:2502.07373*.
- Guibin Zhang, Yanwei Yue, Zhixun Li, Sukwon Yun, Guancheng Wan, Kun Wang, Dawei Cheng, Jeffrey Xu Yu, and Tianlong Chen. 2024a. Cut the crap: An economical communication pipeline for llm-based multi-agent systems. *Preprint*, arXiv:2410.02506.
- Guibin Zhang, Yanwei Yue, Zhixun Li, Sukwon Yun, Guancheng Wan, Kun Wang, Dawei Cheng, Jeffrey Xu Yu, and Tianlong Chen. 2024b. Cut the crap: An economical communication pipeline for llm-based multi-agent systems. *arXiv preprint arXiv:2410.02506*.
- Jiayi Zhang, Jinyu Xiang, Zhaoyang Yu, Fengwei Teng, Xionghui Chen, Jiaqi Chen, Mingchen Zhuge, Xin Cheng, Sirui Hong, Jinlin Wang, Bingnan Zheng, Bang Liu, Yuyu Luo, and Chenglin Wu. 2024c. AFlow: Automating Agentic Workflow Generation. *arXiv preprint ArXiv:2410.10762*.
- Jintian Zhang, Xin Xu, and Shumin Deng. 2023. Exploring collaboration mechanisms for llm agents: A social psychology view. *arXiv preprint arXiv:2310.02124*.
- Qinlin Zhao, Jindong Wang, Yixuan Zhang, Yiqiao Jin, Kaijie Zhu, Hao Chen, and Xing Xie. 2023. Competeai: Understanding the competition behaviors in large language model-based agents. *arXiv preprint arXiv:2310.17512*.
- Xuhui Zhou, Zhe Su, Sophie Feng, Jiaxu Zhou, Jette Huang, Hsien-Te Kao, Spencer Lynch, Svitlana Volkova, Tongshuang Sherry Wu, Anita Woolley, Hao Zhu, and Maarten Sap. 2025. Sotopia-s4: a user-friendly system for flexible, customizable, and large-scale social simulation. *Preprint*, arXiv:2504.16122.
- Mingchen Zhuge, Wenyi Wang, Louis Kirsch, Francesco Faccio, Dmitrii Khizbulin, and Jürgen Schmidhuber. 2024. Gptswarm: Language agents as optimizable graphs. In *Forty-first International Conference on Machine Learning*.
- Christian Zingg, Alexander von Gernler, Carsten Arzig, Frank Schweitzer, and Christoph Gote. 2023. Detecting and optimising team interactions in software development. *Preprint*, arXiv:2302.14609.

## A Notations

We conclude the commonly used notations in Table 4 for reference.

## B Algorithm Workflow

We conclude the overall algorithm workflow of **MasRouter** in Algorithm 1.

## C Case Study

As shown in Tables 5 to 9, we visualize the customized MAS designed by **MasRouter** for varying query difficulties on the five benchmarks.

## D Detailed Cost-Performance Data

### D.1 Inference Cost

In this section, we present the specific overhead and performance of various baselines on the MBPP (Table 10) and HumanEval (Table 11) datasets. The scatter plot with Pareto Front on HumanEval is shown in Figure 6.

### D.2 Training Cost

In Table 12, we compare the training overhead of the SOTA methods that require training with **MasRouter** on MATH and MMLU.

## E The Module Profile

In this section, we present the profiles of each module. We generate the LLM profile following GraphRouter (Feng et al., 2024) and construct the role pool following the method of Macnet (Qian

Notation	Definition
$\mathbb{S} = \{\mathbb{M}, \mathbb{R}, \mathbb{T}\}$	Candidate space containing LLM pool $\mathbb{M}$ , roles set $\mathbb{R}$ , and collaboration modes set $\mathbb{T}$
$\mathbb{M}$	Pool of available LLM backbones
$\mathbb{R}$	Set of predefined agent roles (e.g., Analyst, Developer, Tester)
$\mathbb{T}$	Set of collaboration modes (e.g., Chain, Tree, Debate)
$k$	Number of agents in the multi-agent system
$\mathcal{S} = \{\{\mathcal{M}_i\}_{i=1}^k, \{\mathcal{R}_i\}_{i=1}^k, \mathcal{T}\}$	Multi-agent system with LLMs $\mathcal{M}_i$ , roles $\mathcal{R}_i$ , and collaboration mode $\mathcal{T}$
$\mathcal{M}_i$	Selected LLM backbone for the $i$ -th agent
$\mathcal{R}_i$	Selected role for the $i$ -th agent
$\mathcal{Q}$	Input query to the multi-agent system
$a$	Oracle answer corresponding to the query $\mathcal{Q}$
$f : \mathcal{M} \times \mathcal{R} \times \mathcal{T} \rightarrow \mathcal{S}$	MASR mapping function assigning components to queries
$\pi(\mathcal{S})$	Probability of selecting system $\mathcal{S}$ given query $\mathcal{Q}$
$U(\mathcal{S}; \mathcal{Q}, a)$	Utility function measuring MAS performance
$C(\mathcal{S}; \mathcal{Q})$	Cost function quantifying token expenditure
$\lambda$	Trade-off parameter between utility and cost
$\mathbb{F}_\theta = \mathbb{F}_{\theta_t} \circ \mathbb{F}_{\theta_r} \circ \mathbb{F}_{\theta_m}$	Controller network for collaboration, role allocation, and LLM routing
$\mathbf{H}$	Latent variable capturing query semantics
$\mathbf{H}_{\mathcal{T}}$	Refined representation of the candidate collaboration mode $\mathcal{T}$
$\tau$	Temperature parameter in probability decoding
$\gamma$	Hyperparameter for maximum number of agents
$p(a \mathcal{Q})$	Conditional likelihood of generating answer $a$ via MAS
$\Gamma(z)$	Gamma function approximating non-integer factorials
$\delta(\mathbf{H})$	Complexity mapping function determining the number of agents
$f_\psi(\cdot)$	Encoder extracting semantic information from the query $\mathcal{Q}$
$g_\phi(\cdot)$	Fusion module producing refined representations

Table 4: The notations that are commonly used throughout the manuscript.

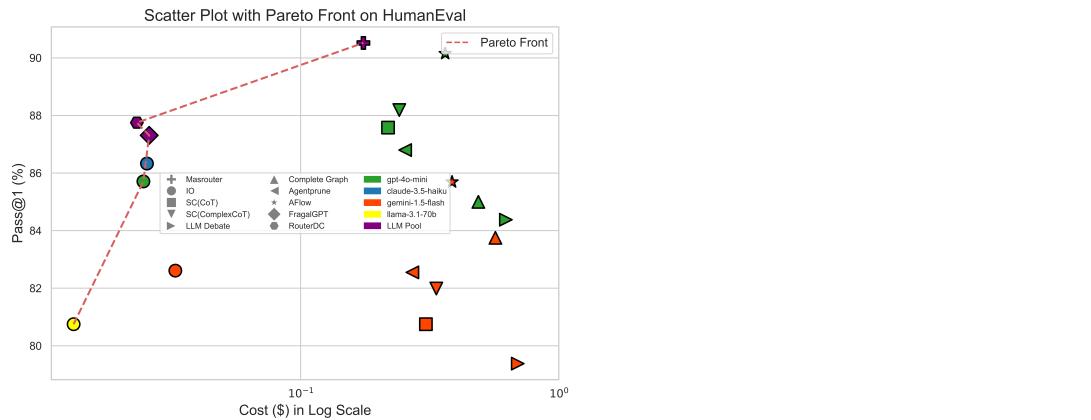


Figure 6: The comparison of the performance and inference cost on the HumanEval dataset. Different shapes of the scatter points represent various types of baselines, while the different colors of the points indicate the use of different LLM backbones.

et al., 2024). We have selected three distinct roles for each task, as presented in the “LLM Profile” and “Role Profile” boxes.

---

**Algorithm 1** Workflow of **MasRouter**

---

**Input** : Benchmark  $\mathcal{D}$ , encoder  $f_\psi(\cdot)$ , fusion module  $g_\phi(\cdot)$ , learning rate  $\alpha$ , search space  $\mathbb{S} = \{\mathbb{M}, \mathbb{R}, \mathbb{T}\}$

**for** query  $\mathcal{Q} \in \mathcal{D}$  **do**

**for** iteration  $t \in \{1, 2, \dots, K\}$  **do**

/\* Collaboration Mode Determination \*/

Sample latent vector  $\mathbf{H} \sim \mathcal{N}(\mu_t(\mathcal{Q}), \text{diag}(\sigma_t^2(\mathcal{Q})))$   $\triangleright$  Eq.(7)

Compute collaboration mode probability:  $p(\mathcal{T}|\mathbf{H}) \propto \exp(\mathbf{H}^\top \mathbf{H}_\mathcal{T}/\tau)$

Determine agent count:  $k = \lceil \delta(\mathbf{H}) \cdot \gamma \rceil$   $\triangleright$  Dynamic scaling

/\* Agent Role Allocation \*/

**for**  $\ell = 1$  **to**  $k$  **do**

Compute role probability:  $\pi_{r_\ell} \propto \exp(\tilde{\mathbf{H}}_{\mathcal{R}_{\ell-1}}^\top \mathbf{H}_{\mathcal{R}_\ell}/\tau)$

Select role  $\mathcal{R}_\ell$  via cascaded inference // Eq.(9)

/\* Agent LLM Routing \*/

Aggregate context:  $\tilde{\mathbf{H}}_\mathcal{M} = \text{FFN}(\mathbf{H} || \mathbf{H}_\mathcal{T} || \text{Norm}(\sum \mathbf{H}_{\mathcal{R}_i}))$

**for** each agent  $i \in \{1, \dots, k\}$  **do**

Compute LLM compatibility:  $\pi_m(\mathcal{M}_i) \propto \exp(\tilde{\mathbf{H}}_\mathcal{M}^\top \mathbf{H}_{\mathcal{M}_i}/\tau)$  // Eq.(11)

Assign LLM  $\mathcal{M}_i$  with multinomial sampling

/\* Optimization \*/

Compute reward  $R = U(\mathcal{S}; \mathcal{Q}, a) - \lambda \cdot C(\mathcal{S}; \mathcal{Q})$  // Eq.(3)

Update  $\theta$  via policy gradient:  $\theta \leftarrow \theta - \alpha \nabla_\theta \mathbb{E}[-R]$  // Section 4.4

---

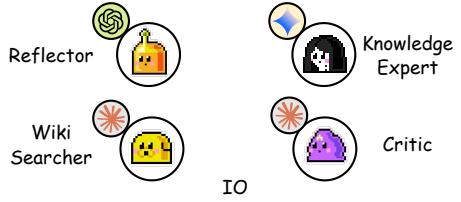
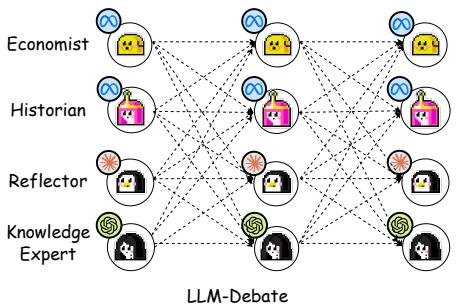
Query	MasRouter Workflow
<p>Bentham defines the fecundity of a pleasure or pain as:          Option A: its chance of occurring. Option B: the degree to which it is felt. Option C: its chance of being followed by sensations of the same kind. Option D: how long it lasts.</p>	
<p>This jurisdiction has the following bribery statute in effect: "Any person who offers or gives a thing of value to a government officeholder in exchange for official action is guilty of bribery." A real estate developer owned a large parcel of land in the suburbs. Although the developer wanted to build an office building on the property, the land was zoned residential. Due to the residential zoning, the developer could not pursue his planned development unless he received a variance from the building commission. The developer held a meeting with a member of the building commission to solicit his approval in securing a zoning variance. To do so, the developer gave the commission member \$10,000 in exchange for his support in approving the zoning variance. Thereupon, the commission member voted to approve the variance, thus making it possible for the developer to commence construction of the office building. The developer was subsequently prosecuted for conspiracy to commit bribery. During the course of the trial, the commission member testified that he faked the agreement with the developer and would have approved the zoning variance regardless of whether the developer gave him any money. Furthermore, in his defense, the developer presented evidence that the other six members of the building commission voted affirmatively to approve the variance. If the jury believed that the commission member would have approved the variance even had he not received the \$10,000, the developer should be found Option A: guilty, because the commission member's agreement to accept the \$10,000 was sufficient to form a conspiratorial objective. Option B: guilty, because he gave the commission member the \$10,000 in exchange for his approval of the zoning variance. Option C: not guilty, because the commission member did not receive a thing of value, since he would have approved the variance regardless of receiving any payment from the developer. Option D: not guilty, because there was no true agreement between the parties.</p>	

Table 5: MMLU dataset

Query	MasRouter Workflow
<p>Grandma Jones baked 5 apple pies for the fireman's luncheon. She cut each pie into 8 pieces and set the five pies out on the buffet table for the guests to serve themselves. At the end of the evening, after the guests had taken and eaten their pieces of pie, there were 14 pieces of pie remaining. How many pieces were taken by the guests?</p>	<p>Chain</p>
<p>The combined age of Peter, Paul and Jean is 100 years old. Find the age of Peter knowing that Paul is 10 years older than John and that Peter's age is equal to the sum of Paul and John's age.</p>	<p>Complete Graph</p>

Table 6: GSM8K dataset

Query	MasRouter Workflow
<p>What is the value of <math>(4 \times 12) - (4 + 12)</math>?</p>	<p>CoT</p>
<p>In the diagram, <math>K</math>, <math>O</math> and <math>M</math> are the centers of the three semi-circles. Also, <math>OC = 32</math> and <math>CB = 36</math>. [asy] pair A, K, O, C, M, B, X, Y, Z; O=(0,0); C=(32,0); M=(50,0); B=(68,0); A=(-68,0); K=(A+C)/2; X=(0,68); Y=(-18,50); Z=(50,18); path nom, bigc, middlec, smallc; nom=A--B--(-100,100)--(-100,100)--cycle; bigc=A..X..B--cycle; middlec=A..Y..C--cycle; smallc=C..Z..B--cycle; fill(bigc, gray(.5)); fill(middlec, white); fill(smallc, white); draw(smallc); draw(middlec); draw(bigc); draw(A--B); label("A", A, S); label("K", K, S); label("O", O, S); label("M", M, S); label("C", C, S); label("B", B, S); dot(K); dot(O); dot(M); [/asy] What is the length of <math>AC\\$</math>?</p>	<p>Complete Graph</p>

Table 7: MATH dataset

Query	MasRouter Workflow
<pre>def is_bored(S: str) -&gt; int: """ You'll be given a string of words, and your task is to count the number of boredoms. A boredom is a sentence that starts with the word "I". Sentences are delimited by '.', '?' or '!'. For example:  &gt;&gt;&gt; is_bored('Hello world') 0 &gt;&gt;&gt; is_bored('The sky is blue. The sun is shining. I love this weather') 1 """</pre>	
<pre>def car_race_collision(n: int) -&gt; int: """ Imagine a road that's a perfectly straight infinitely long line. n cars are driving left to right; simultaneously, a different set of n cars are driving right to left. The two sets of cars start out being very far from each other. All cars move in the same speed. Two cars are said to collide when a car that's moving left to right hits a car that's moving right to left. However, the cars are infinitely sturdy and strong; as a result, they continue moving in their trajectory as if they did not collide.  This function outputs the number of such collisions. """</pre>	

Table 8: HumanEval dataset

Query	MasRouter Workflow
<p>Write a function to sort the given list based on the occurrence of first element of tuples. Your code should pass these tests:</p> <pre>assert sort_on_occurrence([(1, 'Jake'), (2, 'Bob'), (1, 'Cara')]) == [(1, 'Jake'), ('Cara', 2), (2, 'Bob', 1)]; assert sort_on_occurrence([('b', 'ball'), ('a', 'arm'), ('b', 'b'), ('a', 'ant')]) == [('b', 'ball', 'b', 2), ('a', 'arm', 'ant', 2)]; assert sort_on_occurrence([(2, 'Mark'), (3, 'Maze'), (2, 'Sara')]) == [(2, 'Mark'), ('Sara', 2), (3, 'Maze', 1)]</pre>	
<p>Write a function to find out the maximum sum such that no two chosen numbers are adjacent for the given rectangular grid of dimension <math>2 \times n</math>. Your code should pass these tests:</p> <pre>assert max_sum_rectangular_grid([[1, 4, 5], [2, 0, 0]], 3) == 7 assert max_sum_rectangular_grid([[1, 2, 3, 4, 5], [6, 7, 8, 9, 10]], 5) == 24; assert max_sum_rectangular_grid([[7, 9, 11, 15, 19], [21, 25, 28, 31, 32]], 5) == 81.</pre>	

Table 9: MBPP dataset

Method	LLM	Score(%)	Cost(\$)
IO	gpt-4o-mini	72.20	0.143
IO	claude-3.5-haiku	73.40	0.146
IO	gemini-1.5-flash	73.00	0.157
IO	llama-3.1-70b	68.20	0.105
SC(CoT)	gpt-4o-mini	73.00	0.449
SC(CoT)	gemini-1.5-flash	72.00	0.548
SC(ComplexCoT)	gpt-4o-mini	75.60	0.487
SC(ComplexCoT)	gemini-1.5-flash	73.60	0.633
LLM Debate	gpt-4o-mini	73.60	4.427
LLM Debate	gemini-1.5-flash	73.40	4.529
Macnet(Complete Graph)	gpt-4o-mini	75.20	2.932
Macnet(Complete Graph)	gemini-1.5-flash	74.20	3.088
Agentprune	gpt-4o-mini	75.00	1.215
Agentprune	gemini-1.5-flash	75.60	1.352
AFlow	gpt-4o-mini	82.20	1.723
AFlow	gemini-1.5-flash	76.00	1.832
FragalGPT	llm pool	74.40	0.139
RouterDC	llm pool	75.20	0.145
<b>MasRouter</b>	llm pool	84.00	1.039

Table 10: Inference Cost-Performance on MBPP Dataset

Method	LLM	Score(%)	Cost(\$)
IO	gpt-4o-mini	85.71	0.025
IO	claude-3.5-haiku	86.33	0.025
IO	gemini-1.5-flash	82.61	0.032
IO	llama-3.1-70b	80.75	0.013
SC(CoT)	gpt-4o-mini	87.58	0.218
SC(CoT)	gemini-1.5-flash	80.75	0.306
SC(ComplexCoT)	gpt-4o-mini	88.19	0.241
SC(ComplexCoT)	gemini-1.5-flash	81.99	0.335
LLM Debate	gpt-4o-mini	84.38	0.624
LLM Debate	gemini-1.5-flash	79.38	0.693
Macnet(Complete Graph)	gpt-4o-mini	85.00	0.488
Macnet(Complete Graph)	gemini-1.5-flash	83.75	0.568
Agentprune	gpt-4o-mini	86.80	0.254
Agentprune	gemini-1.5-flash	82.55	0.271
AFlow	gpt-4o-mini	90.15	0.363
AFlow	gemini-1.5-flash	85.69	0.386
FragalGPT	llm pool	87.31	0.026
RouterDC	llm pool	87.75	0.023
<b>MasRouter</b>	llm pool	90.52	0.185

Table 11: Inference Cost-Performance on HumanEval Dataset

Method	MATH			MMLU		
	Prompt token	Completion token	Total cost (\$)	Prompt token	Completion token	Total cost (\$)
GPTSwarm	23,031,287	6,943,173	7.63\$	15,525,155	3,983,745	4.70\$
AFlow	321,813,314	28,083,445	21.75\$	13,085,019	11,239,502	8.67\$
<b>MasRouter</b>	3,235,288	2,499,530	3.56\$	4,459,674	2,904,656	1.43\$

Table 12: Training Cost comparison between **MasRouter** and state-of-the-art baselines on MATH and MMLU.

## LLM Profile

```
llm_profile = [
{'Name': 'gpt-4o-mini',
'Description': 'GPT-4o Mini is a smaller version of the GPT-4o language model, designed for faster inference and reduced memory usage. It retains the same capabilities as the full-size model, but with fewer parameters.\n\nThe model costs $0.15 per million input tokens and $0.6 per million output tokens\n\nIn General Q&A Benchmark MMLU, GPT-4o-mini achieves an accuracy of 77.8.\nIn Reasoning Benchmark GPQA, GPT-4o-mini achieves an accuracy of 40.2.\nIn Coding Benchmark HumanEval, GPT-4o-mini achieves an accuracy of 85.7.\nIn Math Benchmark MATH, GPT-4o-mini achieves an accuracy of 66.09.'},
{'Name': 'claude-3-5-haiku-20241022',
'Description': 'The new Claude 3.5 Haiku combines rapid response times with improved reasoning capabilities, making it ideal for tasks that require both speed and intelligence. Claude 3.5 Haiku improves on its predecessor and matches the performance of Claude 3 Opus.\n\nThe model costs $1.0 per million input tokens and $5.0 per million output tokens\n\nIn General Q&A Benchmark MMLU, claude-3-5-haiku achieves an accuracy of 67.9.\n\nIn Reasoning Benchmark GPQA, claude-3-5-haiku achieves an accuracy of 41.6.\nIn Coding Benchmark HumanEval, claude-3-5-haiku achieves an accuracy of 86.3.\n\nIn Math Benchmark MATH, claude-3-5-haiku achieves an accuracy of 65.9.'},
{'Name': 'gemini-1.5-flash-latest',
'Description': 'Gemini 1.5 Flash was purpose-built as our fastest, most cost-efficient model yet for high volume tasks, at scale, to address developers feedback asking for lower latency and cost.\n\nThe model costs $0.15 per million input tokens and $0.6 per million output tokens\n\nIn General Q&A Benchmark MMLU, gemini-1.5-flash achieves an accuracy of 80.0.\n\nIn Reasoning Benchmark GPQA, gemini-1.5-flash achieves an accuracy of 39.5.\nIn Coding Benchmark HumanEval, gemini-1.5-flash achieves an accuracy of 82.6.\n\nIn Math Benchmark MATH, gemini-1.5-flash achieves an accuracy of 74.4.'},
{'Name': 'llama-3.1-70b-instruct',
'Description': 'The Meta Llama-3.1-70b-instruct multilingual large language model (LLM) is a pretrained and instruction tuned generative model in 70B (text in/text out).\n\nThe model costs $0.2 per million input tokens and $0.2 per million output tokens\n\nIn General Q&A Benchmark MMLU, Llama 3.1 achieves an accuracy of 79.1.\nIn Reasoning Benchmark GPQA, Llama 3.1 achieves an accuracy of 46.7.\nIn Coding Benchmark HumanEval, Llama 3.1 achieves an accuracy of 80.7.\nIn Math Benchmark MATH, Llama 3.1 achieves an accuracy of 60.3.'},
{'Name': 'deepseek-chat',
'Description': 'DeepSeek-V3 is a powerful open-source Mixture-of-Experts (MoE) language model developed by Chinese AI company DeepSeek, featuring 671 billion total parameters with 37 billion activated per token, achieving performance comparable to leading closed-source models like GPT-4.\n\nThe model costs $0.27 per million input tokens and $1.1 per million output tokens\n\nIn General Q&A Benchmark MMLU, deepseek-chat achieves an accuracy of 88.5.\nIn Reasoning Benchmark GPQA, deepseek-chat achieves an accuracy of 59.1.\nIn Coding Benchmark HumanEval, deepseek-chat achieves an accuracy of 88.4.\nIn Math Benchmark MATH, deepseek-chat achieves an accuracy of 85.1'},
]
```

## Role Profile

```
Math_role_profile = [
{
  "Name": "MathAnalyst",
  "MessageAggregation": "Normal",
```

```

    "Description": "You are a mathematical analyst. You will be given a math problem, analysis and code from other agents.  

    You need to first analyze the problem solving process, where the variables are represented by letters.  

    Then you substitute the values into the analysis process to perform calculations and get the results.",  

    "OutputFormat": "Calculation",  

    "PostProcess": "None",  

    "PostDescription": "None",  

    "PostOutputFormat": "None"
},
{
    "Name": "MathTeacher",
    "MessageAggregation": "PHP",
    "Description": "You are an excellent math teacher and always teach your students math problems correctly.  

    And I am one of your students. You will be given a math problem, teach me step by step how to solve the problem.",  

    "OutputFormat": "Calculation",  

    "PostProcess": "None",  

    "PostDescription": "None",  

    "PostOutputFormat": "None"
},
{
    "Name": "Inspector",
    "MessageAggregation": "Normal",
    "Description": "You are an Inspector. You will be given a math problem, analysis and code from other agents.  

    Check whether the logic/calculation of the problem solving and analysis process is correct(if present).  

    Check whether the code corresponds to the solution analysis(if present).  

    Give your own solving process step by step based on hints",  

    "OutputFormat": "Answer",  

    "PostProcess": "None",  

    "PostDescription": "None",  

    "PostOutputFormat": "None"
}
]
Coding_role_profile = [
{
    "Name": "Algorithm Designer",
    "MessageAggregation": "PythonInnerTest",
    "Description": "You are an algorithm designer. You will be given a function signature and its docstring by the user.  

    You need to specify the specific design of the algorithm, including explanations of the algorithm, usage instructions, and API references.  

    You can refer to specific examples. When the implementation logic is complex, you can give the pseudocode logic of the main algorithm.  

    Your reply will be more concise. Preferably within fifty words.",  

    "OutputFormat": "Text",  

    "PostProcess": "None",  

    "PostDescription": "None",  

    "PostOutputFormat": "None"
},
{
    "Name": "BugFixer",
    "Description": "You are a programming expert. You will be given a function signature and its docstring by the user. Use a Python code block to write your full implementation (restate the function signature).",  

    "OutputFormat": "CodeCompletion",  

    "PostProcess": "PythonInnerTest",  

    "PostDescription": "You need to provide modified and improved python code based on the current code implementation and problems that arise during testing.  

    You can refer to specific examples. Write your full implementation (restate the function signature). ",  

    "PostOutputFormat": "CodeCompletion"
},

```

```

{
  "Name": "Test Analyst",
  "MessageAggregation": "PythonInnerTest",
  "Description": "You are a Test Analyst. You will be given a function signature and its docstring by the user.  
You need to provide problems in the current code or solution based on the test data and possible test feedback in the question.  
You need to provide additional special use cases, boundary conditions, etc . that should be paid attention to when writing code.  
You can point out any potential errors in the code.Your reply should be more concise.Preferably within fifty words.",
  "OutputFormat": "Text",
  "PostProcess": "None",
  "PostDescription": "You are a programming expert. You will be given a function signature and its docstring by the user.  
Give your own answers to problems that arise in other implementations.  
Use a Python code block to write your full implementation (restate the function signature).",
  "PostOutputFormat": "CodeCompletion"
}
]

Commensense_role_profile =[  

{
  "Name": "Critic",
  "MessageAggregation": "Normal",
  "Description": "You are an excellent critic. Please point out potential issues in other agent's analysis point by point. Give your critical opinion. Finally give the final result",
  "OutputFormat": "Answer",
  "PostProcess": "None",
  "PostDescription": "None",
  "PostOutputFormat": "None"
},  

{
  "Name": "WikiSearcher",
  "MessageAggregation": "Normal",
  "Description": "Please give several key entities that need to be searched in wikipedia to solve the problem. ",
  "OutputFormat": "Keys",
  "PostProcess": "Wiki",
  "PostDescription": "You are a knowlegable expert in question answering.  
Please answer the question based on the explanation of the question keywords obtained from the wikipedia search.",
  "PostOutputFormat": "Answer"
},  

{
  "Name": "Historian",
  "MessageAggregation": "Normal",
  "Description": "You research and analyze cultural, economic, political, and social events in the past, collect data from primary sources and use it to develop theories about what happened during various periods of history.",
  "OutputFormat": "Answer",
  "PostProcess": "None",
  "PostDescription": "None",
  "PostOutputFormat": "None"
}
]

```

## Reasoning Profile

```

reasoning_profile = [  

{
  'Name': 'IO',

```

```
'Description': 'In single-agent IO reasoning, a single agent directly gives an output based on the input.'  
},  
{ 'Name': 'CoT',  
  'Description': 'In single-agent CoT reasoning, a single agent reasons step-by-step to achieve a goal.'  
},  
{ 'Name': 'Chain',  
  'Description': 'In multi-agent chain reasoning, multiple agents sequentially reason and pass information in a chain-like manner.'  
},  
{ 'Name': 'FullConnected',  
  'Description': 'In multi-agent full-graph reasoning, multiple agents reason collectively over the entire graph structure.'  
},  
{ 'Name': 'Debate',  
  'Description': 'In multi-agent debate reasoning, multiple agents engage in a structured argumentative dialogue to explore different perspectives, challenge assumptions, and reach a consensus.'  
},  
{ 'Name': 'Reflection',  
  'Description': 'In reflection reasoning, multiple agents reflect on their own reasoning processes and outcomes to improve their performance.'  
},  
]
```