A Scalable Multi-LLM Collaboration System with Retrival-based Selection and Exploration-Exploitation-Driven Enhancement

Anonymous submission

Abstract

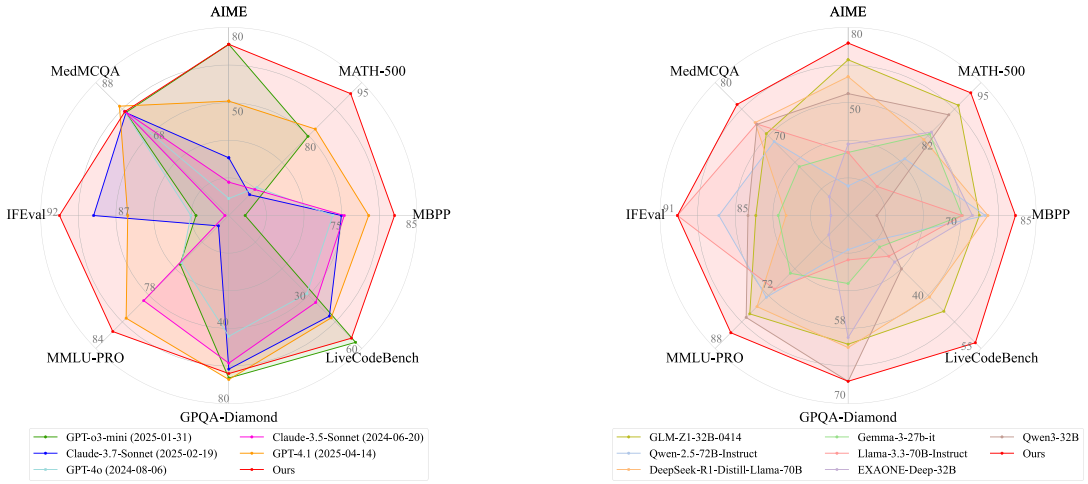
Existing multi-LLM collaboration systems often encounter scalability challenges when integrating new LLMs and tasks, leading to suboptimal performance. To address this, we propose SMCS, a Scalable Multi-LLM Collaboration System designed to effectively coordinate multiple open-source LLMs. The system consists of two core components: a Retrieval-based Prior Selection (RPS) module, which dynamically selects the most suitable LLMs for each input, and an Exploration–Exploitation-Driven Posterior Enhancement (EPE) module, which fosters response diversity and selects high-quality outputs through a hybrid scoring mechanism. Experiments on eight mainstream benchmarks validate the effectiveness of our system: by integrating fifteen open-source LLMs, SMCS outperforms prevailing closed-source LLMs, e.g., *GPT-4.1*(**+5.36%**) and *GPT-o3-mini*(**+5.28%**) across multiple tasks. Remarkably, it even exceeds the average of best results on different datasets with open-source LLMs (**+2.86%**), pushing the upper bound of intelligence.

1 Introduction

Recently, Large Language Models (LLMs) (OpenAI, 2025; Anthropic, 2025a,b) have achieved remarkable success across diverse NLP tasks. With the development of LLM training techniques, a growing number of heterogeneous LLMs, particularly open-source LLMs trained on disparate data, have emerged. Due to structural diversity and bias in the training data, these LLMs possess diverse specialized skills and are expert in distinct areas. Therefore, a pivotal and valuable question naturally arises: how can we sustainably harness and scale up the vast and diverse ensemble of LLMs to continually push the performance frontier or the intelligence upper bound?

To answer this question, a general approach is to construct a Multi-LLM Collaboration System (MCS). The MCS aims to orchestrate interactions among multiple LLMs, enable information exchange and integration, and generate high-quality responses. Emerging works have explored the construction of MCS, which can be broadly divided into two categories: (1) MCS via prior LLM selection. These approaches (Chen et al., 2025; Lu et al., 2023; Shnitzer et al., 2023; Chen et al., 2024d) select appropriate LLMs before response generation by leveraging prior knowledge corresponding to LLMs, such as their performance on standard benchmarks or model embeddings obtained from training on specific datasets. By selecting the most suitable models for each given question in advance, these methods aim to increase the likelihood of generating high-quality responses. (2) MCS via posterior response enhancement. These approaches (Chen et al., 2024c, 2023a; Gui et al., 2024; Choudhury, 2025) assess the quality of responses after each LLM has generated its answer, using inter- or intra-response criteria such as reward model scores, perplexity, or majority voting. Due to performing reasoning, these methods provide a more accurate evaluation of response quality compared to relying solely on prior information.

However, both categories of methods encounter challenges when scaling the number of LLMs and tasks. For MCS based on prior LLM selection, they either require end-to-end router training (Chen et al., 2024d) for each individual LLM, making it difficult to continuously incorporate new LLMs, or rely on limited and discrete capability labels (Chen et al., 2025), which are insufficient for comprehensive analysis on a given question and hard to handle unseen questions. For MCS based on posterior response enhancement, these methods typically rely on a single posterior criterion, which can introduce bias and lead to inaccurate quality assessments. Moreover, they mainly focus on selecting from an existing pool of responses, lacking the ability to generate new and diverse high-quality responses,



(a) Comparisons with closed-source LLMs.

(b) Comparisons with open-source LLMs.

Figure 1: Results on eight mainstream benchmarks. The proposed SMCS orchestrates fifteen open-source LLMs, surpassing both open-source and closed-source LLMs and pushing the upper bound of a single LLM.

limiting their performance upper bound. Besides the above limitations, current MCS methods often fail to effectively integrate prior and posterior methods in a coupled manner, which causes unfiltered low-quality responses as bottlenecks, which significantly hinders the overall performance and scalability of the collaboration system.

To enhance the scalability and break through the performance upper bound of MCS, we propose a novel framework called Scalable Multi-LLM Collaboration System (SMCS). Specifically, we first construct a question bank comprising diverse questions from multiple domains, along with an LLM pool containing plentiful heterogeneous LLMs. Each LLM in the pool is evaluated on the question bank to record its response, representing its capacity across diverse domains. Further, inspired by Retrieval-Augmented Generation (RAG) (Lewis et al., 2020; Chen et al., 2024a), we design a retrieval-based prior selection (RPS) strategy: given any question, we retrieve similar questions from the question bank. A weighted score is computed for each LLM based on its performance on the retrieved questions, which serves as the prior information for selecting high-scoring LLMs. After that, we introduce exploration–exploitation-driven posterior enhancement (EPE): in the exploration phase, these responses are dropped via prior scores to form multiple answer subsets, which are independently aggregated by the selected LLM aggregator; in the exploitation phase, the aggregating responses are evaluated using a hybrid posterior scores of mean pairwise similarity and perplexity. The aggregated response with the highest score is selected as the final response.

We conduct extensive experiments to validate the effectiveness of the proposed framework across eight datasets. Notably, by jointly leveraging fifteen mid-sized open-source LLMs, SMCS significantly surpasses the current flagship closed-source models, such as GPT-4.1(**+5.36%**) and GPT-o3-mini(**+5.28%**). Moreover, SMCS also exceeds both the average performance of the ensemble of the open-source best baselines(**+2.86%**). This demonstrates the strong capability of SMCS and its potential to break through the upper bound of performance. Besides, SMCS can consistently obtain gains without remarkable saturation by progressively increasing the number of LLMs, demonstrating excellent scalability. Our contributions are summarized as follows:

- We first present a comprehensive analysis of existing multi-LLM collaboration systems from prior and posterior perspectives, and identify several key limitations hindering the development of scalable and high-performance MCS frameworks.
- We propose SMCS, a scalable multi-LLM collaboration framework. It jointly considers prior and posterior information, where a retrieval-based prior selection strategy is proposed to recruit suitable LLMs at the instance level, and an exploration–exploitation-driven posterior enhancement strategy is designed to generate higher-quality responses.
- Extensive experiments across diverse datasets validate the scalability and effectiveness of SMCS, demonstrating its ability to enable continuous expansion of LLMs while harness-

ing open-source models to surpass prevailing closed-source models.

2 Related works

Prior-based LLM Collaboration. Prior-based methods focus on dynamically selecting or routing LLMs before generating responses. Recent research explores LLM routing, where a selector determines the most suitable model for a given question without integrating all LLMs. The preliminary work (Shnitzer et al., 2023) proposes binary classifiers to predict the correctness of individual LLMs, while ZOOTER (Lu et al., 2023) aligns a router with reward-model supervision. RouterDC (Chen et al., 2024d) utilizes dual contrastive learning for improved accuracy. While GraphRouter (Feng et al., 2025) constructs model selection as a dynamic link prediction problem by constructing heterogeneous task-query-LLM graphs with GNNs, MODEL-SAT (Zhang et al., 2025) focuses on performance-based capability representations. The latter specifically leverages a lightweight LLM to predict the most effective candidate for a given task. Most relevant to our work, Symbolic_MoE (Chen et al., 2025) proposes a Mixture-of-Experts framework that dynamically selects and combines LLMs based on skill-specific expertise.

Posterior-based LLM Collaboration. Posterior based methods aggregate outputs from multiple LLM executions to derive an improved response. Simple but effective techniques such as Voting (Li et al., 2024; Wang et al., 2022) and advanced ranking-based approaches such as LLM-Blender (Jiang et al., 2023), demonstrate the benefits of ensemble refinement. Besides, techniques like majority voting (Chen et al., 2024c), self-consistency (Wang et al., 2022; Chen et al., 2023b), and best-of-n sampling (Gui et al., 2024) could enhance reliability in tasks lacking verification tools. Mixture of Agents (MoA) (Wang et al., 2024a) introduces a framework for combining LLM agents into ensembles, relying on a fixed set of agents across tasks. Similarly, Self-MoA (Li et al., 2025) argues that invoking a single high-performing model multiple times, paired with an optimal aggregator, can achieve competitive performance without leveraging diverse LLMs.

While existing MCS demonstrate effectiveness, they suffer from two critical limitations: (1) scalability constraints that hinder seamless integration of new LLMs, and (2) suboptimal performance due to inefficient utilization and limited exploration of different LLMs' responses. In this work, we propose SMCS that incorporates the advantages of prior and posterior approaches. It enables scalable instance-level LLM selection via RPS strategy, and extends the diversity of responses while making full use of them via designed EPE.

3 Method

In this section, we first provide an overview of SMCS in Sec. 3.1. Then, the construction of the question bank is stated in Sec. 3.2. Next, we present the retrieval-based prior selection (RPS) and exploration–exploitation-driven posterior enhancement (EPE) in Sec. 3.3 and Sec. 3.4. A visual comparison between proposed techniques and existing methods is shown in Fig. 2.

3.1 Overall Framework

As shown in Fig. 3, for scalable and generalizable capability assessment for each LLM, SMCS constructs a unified question bank by integrating questions from multiple domains with their labels. Each LLM is evaluated on the unified question bank to obtain a fine-grained assessment of its capability distribution, which contains prior information of each LLM. During inference, SMCS consists of two stages: (1) Retrieval-based Prior Selection (2) Exploration-exploitation-driven Posterior Enhancement. In the first stage, given a question, SMCS retrieves related questions from the question bank to obtain prior information of each LLM and selects suitable expert LLMs as referencers. Then, all referencers are forwarded to collect their responses as references. Meanwhile, SMCS selects the LLM with the strongest instruction-following capability to serve as the aggregator. In the second stage, the references are dropped based on the distribution of the prior information of the corresponding referencers to generate multiple reference subsets. Each subset is aggregated by the aggregator, resulting in multiple candidates to explore high-quality responses. Finally, SMCS evaluates each candidate using a hybrid posterior score that incorporates both intra-response and inter-response criteria, serving as an exploitation over the output space of the aggregator. The candidate with the highest score is selected as the final response.

3.2 Unified Question Bank

Due to the heterogeneity of LLMs from various sources, it is infeasible to extract prior informa-

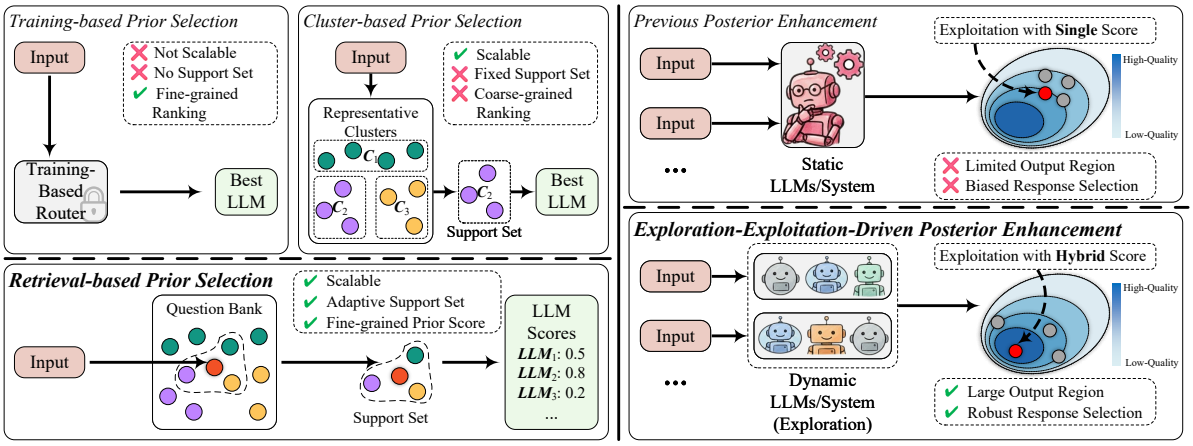


Figure 2: The illustration of two core innovations in proposed SMCS. SMCS adopts different and more advanced paradigms for prior selection and posterior enhancement, achieving significant scalability and performance.

tion by directly analyzing their architectures or parameters. To guarantee the generalization of prior information extraction across diverse LLMs and tasks, SMCS adopts a black-box evaluation strategy that analyzes the responses generated by each LLM to specific inputs. Specifically, given an LLM bank $\mathcal{A} = \{A_1, A_2, ...A_R\}$ containing R LLMs, SMCS constructs a unified question bank $\mathcal{B} = \{(x_i^{qb}, y_i^{qb})|i \in [1, N]\}$ by sampling N questions from the validation sets of diverse tasks for a comprehensive capability assessment for each LLM. x_i^{qb} and y_i^{qb} are the i_{th} question and the corresponding label in the question bank, respectively. After constructing the unified question bank, each LLM A_i is forwarded to answer all questions in the question bank, obtaining a capability vector $V_i^{qb} \in \{0, 1\}^{N\times 1}$ that represents its capabilities across diverse tasks,

$$V_i^{qb} = \left[\mathbf{1}_{\{A_i(x_0^{qb})=y_0^{qb}\}}, \mathbf{1}_{\{A_i(x_1^{qb})=y_1^{qb}\}}, ..., \mathbf{1}_{\{A_i(x_N^{qb})=y_N^{qb}\}}\right]^\top \quad (1)$$

where $\mathbf{1}_{\{\cdot\}}$ is the indicator. It is worth noting that for notational simplicity, the parameters θ_i of A_i are omitted, and we use "=" to represent verifying the correctness of a response. Moreover, a pre-trained embedding model $\mathcal{M}_{emb}$ is introduced to embedded each question x_i^{qb} into latent space for the later retrieval, denoted as $e_i^{qb} = Norm(\mathcal{M}_{emb}(x_i^{qb})) \in \mathbb{R}^{d\times 1}$, where d is the embedding dimension of $\mathcal{M}_{emb}$ and $Norm(\cdot)$ is normalization function. The capacity vector V_i^{qb} records the historical performance of each LLM at the instance level, providing fine-grained prior information.

3.3 Retrieval-based Prior Selection

The key to selecting the optimal LLMs is establishing the relevance between the given question and the collected prior information. Existing methods typically introduce a preprocessing procedure and assign the given question to an explicit or implicit category based on unsupervised clustering (Jitkrittum et al., 2025; Srivatsa et al., 2024) or supervised learning (Shnitzer et al., 2023; Chen et al., 2024d). The prior information associated with that category is used to estimate the capabilities of different LLMs for the given question. However, the complex preprocessing introduces noise and bias, potentially incorporating irrelevant prior information. To address these issues, inspired by the Retrieval-Augmented Generation (RAG) (Lewis et al., 2020; Chen et al., 2024a) paradigm, we design a retrieval-based prior selection without complex preprocessing. The core idea is to retrieve questions similar to the given question as support questions and then utilize the weighted scores on the support questions as a prior representation of LLMs' capabilities. Specifically, given a question x^{in}, a embedding model $\mathcal{M}_{emb}$ transfer it into an embedding vector $e^{in} = Norm(\mathcal{M}_{emb}(x^{in})) \in \mathbb{R}^{d\times 1}$. Then, e^{in} is computed cosine similarity with all e^{qb} to obtain similarity vector $S^{in} \in [0, 1]^{N\times 1}$, denoted as

$$S^{in} = [e_1^{qb}, e_2^{qb}, ..., e_N^{qb}]^\top e^{in}. \quad (2)$$

To adaptively retrieval the support questions, a base number N^{sup_base} is defined to ensure sufficient evaluation coverage. Moreover, a tolerance threshold coefficient $\gamma \in [0, 1]$ is introduced to obtain a relatively threshold to select the support questions. The index of support questions is denoted as

$$I^{sup} = \{i|S^{in}[i] \geq \gamma \max_{N^{sup\_base}}(S^{in})\} \quad (3)$$

where $\max_k(\cdot)$ refers to the k_{th} largest element in a vector. The number of support questions is

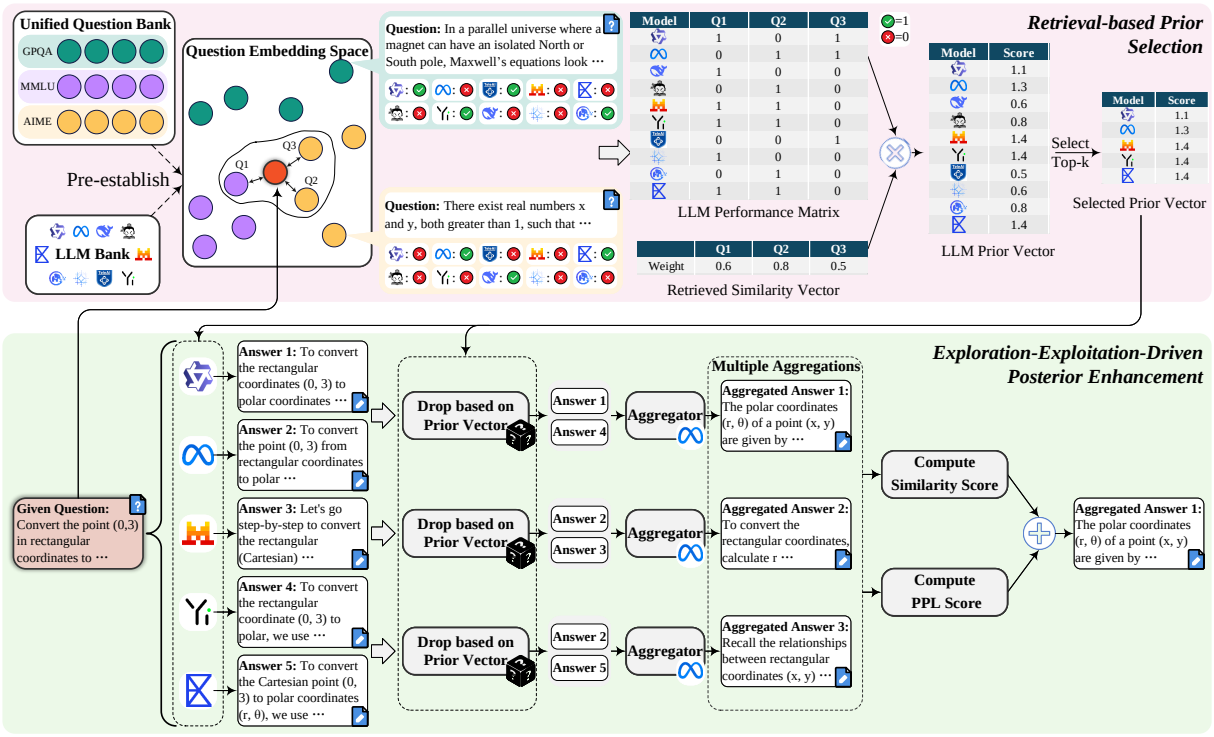


Figure 3: Overview of our SMCS framework. It dynamically selects Top-K expert LLMs from the predefined LLM bank through RPS module, then optimizes responses via EPE module to generate high-quality outputs.

$N^{sup} = |I|$. Then, according to I^{sup}, the retrieved similarity vector $\hat{S}^{in} \in [0,1]^{N^{sup}\times 1}$ can be indexed from S^{in}, denoted as $\hat{S}^{in} = S^{in}_I$, and LLM performance matrix $M^{qb} \in \{0,1\}^{R\times N_{sup}}$ can be indexed from LLM capability vector V^{qb}, denoted as $M^{qb} = [V^{qb}_{I,1}, V^{qb}_{I,2}, ..., V^{qb}_{I,R}]^\top$, where R is the number of LLMs in LLM bank. The LLM prior vector $V^{ref} \in \mathbb{R}^{R\times 1}$ can be computed by

$$V^{ref} = M^{qb}\hat{S}^{in}. \tag{4}$$

Given the number of selected referencers K, the selected prior vector can be denoted as

$$\hat{V}^{ref} = V^{ref}_{I^{ref}}, I^{ref} = \text{argtop}_K(V^{ref}), \tag{5}$$

where $\text{argtop}_K(\cdot)$ refers to obtaining the indices of the largest K elements of a vector. The LLMs with the indices I^{ref} are selected as referencers in the inference, denoted as $\mathcal{A}^{ref} = \{A_i | i \in I^{ref}\}$.

3.4 Exploration-Exploitation-Driven Posterior Enhancement

After prior selection, SMCS is required to further evaluate and organize references to filter out inferior information and generate higher-quality responses. Due to differences in training data and architectures, reference responses differ significantly in patterns and distributions, making direct posterior evaluation challenging. To address these issues, we adopt an exploration-exploitation-driven posterior enhancement strategy. It explore diverse and high-quality aggregations by dropping some inferior references and aggregating multi times based on prior information, and exploits the aggregations by introducing a hybrid posterior score to select the optimal aggregation as final response. Specifically, given the referencers LLMs $\mathcal{A}^{ref}$ from prior selection, the references can be collected by forwarding all referencers, denoted as $O^{all} = \{A_i(x^{in}) | A_i \in \mathcal{A}^{ref}\}$. For exploration, given a dropping number K_{drop}, the references are dropped following the prior-based discrete sampling distribution $\mathcal{D}$, which is denoted as

$$\mathcal{D} = \left[\frac{e^{\widetilde{\hat{V}^{ref}}[1]}}{\sum_{j=1}^{K} e^{\widetilde{\hat{V}^{ref}}[j]}}, \frac{e^{\widetilde{\hat{V}^{ref}}[2]}}{\sum_{j=1}^{K} e^{\widetilde{\hat{V}^{ref}}[j]}}, ..., \frac{e^{\widetilde{\hat{V}^{ref}}[K]}}{\sum_{j=1}^{K} e^{\widetilde{\hat{V}^{ref}}[j]}}\right],$$
$$\widetilde{\hat{V}^{ref}} = \left[\frac{\hat{V}^{ref}[1] - \overline{\hat{V}^{ref}}}{std(\hat{V}^{ref})}, \frac{\hat{V}^{ref}[2] - \overline{\hat{V}^{ref}}}{std(\hat{V}^{ref})}, ..., \frac{\hat{V}^{ref}[K] - \overline{\hat{V}^{ref}}}{std(\hat{V}^{ref})}\right] \tag{6}$$

where $std(\cdot)$ refers to obtaining the standard deviation, and we use a renormalize-after-each-draw rule (Panahbehagh et al., 2021) to achieve successive unequal-probability sampling (YU, 2012), which can be seen as sampling $K - K_{drop}$ references from O^{all} following $\mathcal{D}$ without replacement. After prior dropping n times, multiple subsets O^{sub} of O^{all} are obtained, and each O^{sub} is aggregated by an aggregator A_{agg} to generate a aggregation set, denoted as $G_i = A_{agg}(cat(O^{sub}_i))$ where $cat(\cdot)$

Model	AIME	MATH-500	MBPP	LiveCodeBench	GPQA-Diamond	MMLU-PRO	IFEval	MedMCQA	Avg
Close-source LLMs									
GPT-o3-mini(2025-01-31) (OpenAI, 2024)	73.33	84.40	62.00	54.70	66.67	74.00	82.00	74.92	71.50
Claude-3.7-Sonnet(2025-02-19) (Anthropic, 2025b)	26.70	73.20	75.40	41.30	63.64	69.43	88.00	74.75	64.05
GPT-4o(2024-08-06) (Achiam et al., 2023)	10.00	74.60	74.20	29.80	52.53	73.83	82.30	76.17	59.18
Claude-3.5-Sonnet(2024-06-20) (Anthropic, 2025a)	16.70	74.20	75.80	34.30	61.62	78.34	80.30	76.00	62.16
GPT-4.1(2025-04-14) (OpenAI, 2025)	50.00	85.80	79.20	42.20	67.17	80.43	86.00	80.58	71.42
Close-source Average	35.34	78.44	73.32	40.46	62.33	75.21	83.72	76.48	65.66
Open-source LLMs									
GLM-Z1-32B-0414 (GLM et al., 2024)	66.70	90.00	74.40	44.40	59.60	76.76	83.00	70.50	70.67
Qwen-2.5-72B-Instruct (Team, 2024b)	16.70	78.80	75.80	26.10	45.45	72.16	86.30	69.08	58.80
DeepSeek-R1-Distill-Llama-70B (DeepSeek-AI, 2025)	60.00	82.80	76.40	40.70	60.10	74.75	80.30	72.50	68.44
QwQ-32B (Team, 2024c)	46.70	87.80	81.80	38.60	57.07	74.67	81.70	69.83	67.27
Gemma-3-27b-it (Team et al., 2024)	30.00	84.00	70.40	27.70	50.51	65.47	81.00	64.58	59.21
Qwen2.5-32b-Instruct (Team, 2024a)	20.00	75.60	76.00	24.00	40.91	69.15	78.70	62.92	55.91
TeleChat2-35B-32K (Wang et al., 2024c)	10.00	70.00	70.00	19.50	33.33	67.98	82.00	57.08	51.24
InternLM2.5-20B-Chat (Cai et al., 2024)	3.30	55.20	55.00	14.90	34.85	44.23	64.70	51.92	40.51
Llama-3.3-70B-Instruct (Grattafiori et al., 2024)	30.00	73.00	70.40	30.10	46.97	69.87	90.00	72.25	60.32
EXAONE-Deep-32B (LG AI Research, 2025)	33.30	84.38	72.80	31.60	58.59	54.76	76.30	59.17	58.86
Qwen2.5-Coder-32B-Instruct (Hui et al., 2024)	16.70	73.60	78.00	27.70	41.92	61.79	80.30	57.25	54.66
Qwen3-32B (Team, 2025)	53.30	88.00	50.60	33.40	65.15	77.76	83.70	72.17	65.51
Llama-3.3-Nemotron-Super-49B-v1 (Grattafiori et al., 2024)	16.70	75.20	65.40	28.00	48.48	67.47	82.70	70.92	56.86
DeepSeek-R1-Distill-Qwen-32B (DeepSeek-AI, 2025)	56.70	85.60	81.00	44.70	60.10	75.17	73.70	67.25	68.03
HuatuoGPT-o1-72B (Chen et al., 2024b)	16.70	73.00	78.00	27.40	50.00	74.16	74.00	75.25	58.56
Open-source Average	31.79	78.47	71.73	30.59	50.20	68.41	79.89	66.18	59.66
Other Methods									
Symbolic-MoE* (Chen et al., 2025)	50.00	90.40	82.60	43.01	62.63	80.60	89.00	74.88	71.64
MoA (Wang et al., 2024a)	53.33	87.80	82.00	40.12	58.80	79.6	89.33	73.08	70.51
Self-MoA (Li et al., 2025)	76.67	93.00	83.20	29.39	64.41	69.89	86.00	74.88	72.18
Self Consistency (Best on Validation) (Chen et al., 2023b)	60.00	90.40	82.40	40.12	63.64	78.01	90.39	74.83	72.47
Majority Voting (Chen et al., 2024c)	56.67	90.2	80.4	34.65	26.26	80.85	80.67	73.33	65.38
Simple Router	46.70	88.00	81.80	33.40	60.10	72.50	90.00	72.50	68.40
Ours v.s. Strong Baselines									
Open-source Upper Bound	66.70	90.00	81.80	44.70	65.15	77.76	90.00	75.25	73.92
SMCS(ours)	**73.33**	**92.60**	**82.80**	52.58	65.15	**82.02**	**90.00**	75.75	**76.78**
- v.s. Self Consistency (Best on Validation)	↑13.33	↑2.20	↑0.40	↑12.46	↑1.51	↑4.01	↓0.39	↑0.92	↑4.31
- v.s. MoA	↑20.00	↑4.80	↑0.8	↑12.46	↑6.35	↑2.42	↑0.67	↑2.67	↑6.27
- v.s. GPT-o3-mini	0	↑8.20	↑20.80	↓2.12	↓1.52	↑8.02	↑8.00	↑0.83	↑5.28

Table 1: Main Results of our SMCS framework with fifteen open-source LLMs on eight mainstream benchmarks.

refers to concatenating the references and injecting prompts for aggregating. Then, the mean pairwise similarity of an aggregation G_i is computed as a similarity score $\mathcal{S}_i^{sim}$, denoted as $\mathcal{S}_i^{sim} = \frac{1}{n}\sum_{j=1}^{n} sim(G_i, G_j)$, where $sim(\cdot,\cdot)$ is computing cosine similarity using embedding model same as Formulation 2. Meanwhile, the perplexity score $\mathcal{S}_i^{PPL}$ is computed as $\mathcal{S}_i^{PPL} = 1 - PPL(G_i)$, where $PPL(\cdot)$ refers to computing the perplexity (Parsing, 2009; Hu et al., 2024) of a response. Finally, the total score of an aggregation can be denoted as

$$\mathcal{S}^{total} = \mathcal{S}^{sim} + \lambda \mathcal{S}^{PPL}, \tag{7}$$

where λ is the balance coefficient. Finally, the aggregation G with the highest $\mathcal{S}^{total}$ is regarded as the final response of SMCS.

4 Experiments

4.1 Experimental Setting

Datasets. We establish a multi-domain evaluation comprising eight mainstream benchmarks spanning four key task categories: (1) Mathematical Problem Solving (MATH-500 (Hendrycks et al., 2021), AIME2024 (MAA, 2024)), (2) Complex Reasoning (GPQA (Rein et al., 2024), MMLU-PRO (Wang et al., 2024b), MedMCQA (Pal et al., 2022)), (3) Instruction Following (IFEval (Zhou et al., 2023)), and (4) Code Generation (MBPP (Austin et al., 2021), LiveCodeBench (Jain et al., 2024)). Each dataset is split into non-overlapping validation and test sets, with all validation sets combined to form the unified question bank for all benchmarks. See Appendix A.1 for more details.

LLM Bank. To achieve a balance between model diversity and efficiency, we carefully curate a collection of fifteen mid-sized open-source LLMs (from 20B to 72B) from various architectural families. SMCS framework employs a two-tiered model utilization strategy: reference models are dynamically selected from the full LLM bank during inference via task requirements, while the critical aggregator model is handled by Llama-3.3-70B-Instruct due to its exceptional instruction-following performance. See Appendix A.2 for more details.

4.2 Main Results

As demonstrated in Table 1, our proposed SMCS framework establishes new state-of-the-art results across eight diverse benchmarks. Through comprehensive comparisons with (1) five leading close-source models (including GPT-o3-mini (OpenAI, 2024), GPT-4.1 (OpenAI, 2025), GPT-4o (Achiam et al., 2023), Claude-3.5-Sonnet (Anthropic, 2025a), Claude-3.7-Sonnet (Anthropic, 2025b)), (2) fifteen representative open-source models, and (3) six existing collaboration methods, our approach demonstrates consistent and sub-

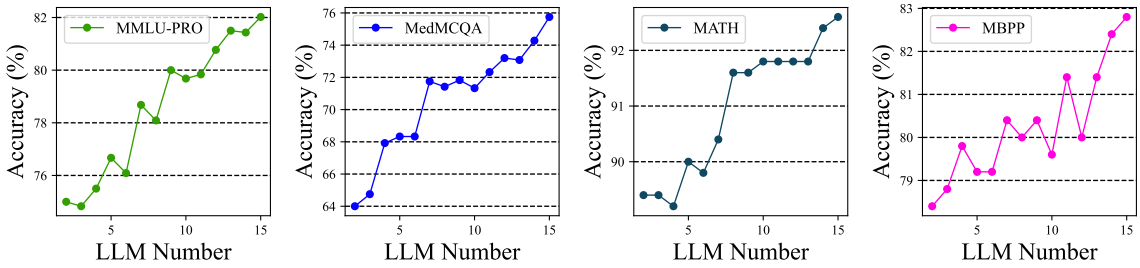


Figure 4: The scalability curve of SMCS. It can increasingly incorporate more LLMs for higher performance.

stantial improvements across all evaluation dimensions. For example, our SMCS framework achieves 76.78% average accuracy on eight benchmarks, representing substantial gains of +11.12% and +17.12% over the average closed-source (65.66%) and open-source (59.66%) baselines, respectively. Compared to existing collaboration approaches, SMCS outperforms Symbolic-MoE* (Chen et al., 2025) by +5.14%, MoA (Wang et al., 2024a) by +6.27%, Self-MoA (Li et al., 2025) by +4.6%. Remarkably, our solution even exceeds open-source upper bounds (+2.86%), while significantly surpassing individual leading models, including GPT-4.1 (+5.36%), GPT-4o (+17.60%), and Claude-3.5-Sonnet (+12.73%). It demonstrates that SMCS can effectively combine the strengths of multiple LLMs to achieve unprecedented performance.

4.3 Efficiency Analysis

Although SMCS focuses on exploring the maximum performance boundary of multi-LLM collaboration rather than optimizing efficiency, we further report the API cost and average query latency of multi-LLM methods and leading closed-source LLMs. As shown in Table 2, SMCS achieves remarkable performance superiority, e.g., +5.14% and +5.28% compared with Symbolic-MoE and GPT-o3-mini, with a competitive cost and inference time. It verifies the feasibility and economy of SMCS in practical implementation. The efficiency of SMCS comes from 1) APIs of mid-sized open-source LLMs are dramatically cheaper than closed-source LLMs; 2) Although SMCS requires more LLM forward passes, most of these forward passes, e.g., the inferences of different referencers and aggregating multiple times, are independent and can be parallelized, making the overall inference time only determined by the slowest LLM.

4.4 Scaling Ability

To empirically validate the scalability of SMCS framework, we conducted experiments measuring performance improvements with increasing numbers of input LLMs. Fig. 4 shows key findings across four standard benchmarks, revealing a clear positive correlation between the scale of input LLMs and overall performance. For instance, on MMLU-PRO, our SMCS achieves approximately 77% accuracy with five LLMs. When scaled to ten LLMs, performance improves to nearly 80%, and with 15 LLMs, the final accuracy approaches 82%. It not only demonstrates the capability of SMCS to leverage diverse LLMs effectively but also shows that SMCS has the potential to obtain sustainable performance gains as the number of LLMs continually scales up.

Method/Model	Avg Acc(%)	Cost($)	Avg Latency(s)
GPT-o3-mini	71.50	15.36	10.42
Claude-3.7-Sonnet	64.05	20.38	17.92
GPT-4.1	71.42	11.98	**10.11**
MoA	70.51	9.42	18.64
Self MoA	72.18	9.14	12.82
Symbolic-MoE	71.64	**7.86**	12.47
Self Consistency(Best on Validation)	72.47	8.39	12.78
SMCS(ours)	**76.78**	8.11	12.32

Table 2: Cost and average latency of different methods.

Question Bank	AIME	MBPP	GPQA-Diamond	MedMCQA
Single(MMLU-PRO)	73.33	82.2	64.14	75.17
Unified(Eight Datasets)	73.33	82.80	65.15	75.75
w/o Question Bank(Random Select)	56.67	82.20	56.56	74.25

Table 3: Comparison with different question banks.

4.5 Out-of-Distribution Performance

To demonstrate the generalization of the proposed prior selection, we conduct out-of-distribution retrieval experiments. Specifically, we build a question bank using 5,512 questions only from MMLU-PRO and evaluate SMCS on the other four datasets. As shown in Table 3, the results demonstrate that even with a question bank using an out-of-distribution dataset, our retrieval-based prior selection surpasses random selection significantly, while with only marginal performance drops compared with using a multi-dataset question bank. It verifies the strong generalization of our prior selection mechanism when facing out-of-distribution questions. Moreover, SMCS can introduce more diverse questions to further refine LLM capability assessments and boost performance.

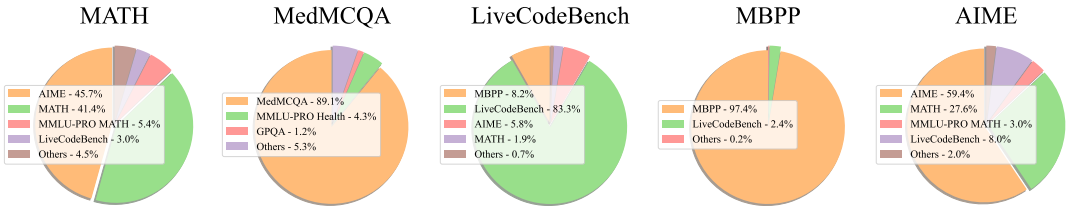


Figure 5: The proportion of support questions retrieved from different source datasets for a given question.

4.6 Analysis on Prior Selection

As shown in Fig. 5, we display the proportion of support questions retrieved from different source datasets. For a given question, the retrieved support questions are mostly from subjects with similar capability requirements. Specifically, a substantial portion of the support questions are retrieved from the same dataset as the given question, while others are from other datasets with similar subjects. For instance, in the case of MATH, nearly half of the retrieved support questions are from AIME, and for MedMCQA, several are retrieved from the "Health" category in MMLU-PRO. It not only verifies the effectiveness of our proposed method in bridging the given question with relevant prior information but also demonstrates its ability to perform cross-dataset retrieval. This capability significantly increases the amount of relevant prior information, enhancing model assessment and suggesting a potential for scalability. Additionally, we observe a retrieval connection between mathematics and code-generation tasks, e.g., LiveCodeBench and AIME retrieve questions from each other, indicating that solving coding and mathematical problems may require similar capabilities. Moreover, to verify the correlation between prior LLM evaluation and practical performance, we introduce a pairwise ranking score inspired by the ranking loss (Hu et al., 2021; Xu et al., 2021) in Neural Architecture Search (NAS), denoted as

$$Sco_{rank} = \frac{\sum_{i \in I^{test}} \sum_{j \in P} \sum_{k \in N} \mathbf{1}\{V_i^{ref}[j] > V_i^{ref}[k]\}}{\sum_{i \in I^{test}} |\{j|V_i^{test}[j]=0\}| \cdot |\{k|V^{test}[k]=1\}|}, \quad (8)$$

where I^{test} is the index of test questions, V_i^{ref} is the LLM prior vector in Formulation 4 for i_{th} test question. $V_i^{test} \in \{0,1\}^R$ represents the correctness of all LLMs on the test set, where R is the number of LLMs, 0 and 1 indicate an incorrect and correct answer, respectively. As shown in Table 4, compared with Symbolic-MoE, our retrieval-based method consistently obtains a higher score, suggesting our method provides a more accurate prior evaluation of LLMs.

	MATH	MBPP	MedMCQA	LiveCodeBench	AIME
Symbolic-MoE	73.54	70.2	64.62	48.64	60.64
SMCS(ours)	74.46	70.61	65.17	68.19	86.5

Table 4: Pairwise ranking scores of different methods.

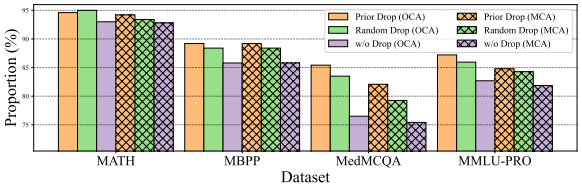


Figure 6: The comparison of different posterior enhancement methods. OCA: One Correct Answer proportion; MCA: Multiple Correct Answers proportion.

4.7 Analysis on Posterior Enhancement

To verify the effectiveness of the proposed posterior exploration and exploitation, we analyze the proportion of correct answers within multiple aggregation responses using different strategies, including the proposed prior drop, random drop, and vallina aggregating without drop. We use the existing one correct answer proportion (OCA) and the existing multiple correct answers proportion (MCA) to indicate the diversity and quality of multiple aggregating responses, respectively. As shown in Fig. 6, compared with aggregating without drop and random drop, our method can consistently obtain both higher OCA and MCA, suggesting our method can explore a more optimal output region by aggregating multiple times. Thus, there are abundant high-quality candidate responses for exploitation.

5 Conclusion

In this paper, to boost the scalability and performance of multi-LLM collaboration systems, we propose SMCS by prior selection and posterior enhancement. Specifically, based on a unified question bank, we propose a retrieval-based prior selection to select the optimal LLMs. Moreover, we propose an exploration–exploitation-driven posterior enhancement, which aggregates references multiple times based on prior information to explore high-quality responses. To select the final output, we propose a hybrid score that combines perplexity and mean pairwise similarity. Extensive experiments demonstrate the effectiveness of SMCS.

Limitations

In this section, we discuss the limitations of the proposed SMCS to provide an underlying advance in the field of multi-LLM collaboration systems and to point out promising directions for future research.

Lack of Efficiency Optimization. To maximize performance upper bounds, SMCS framework does not set constraints on the computational cost of selected LLMs. Thus, the system requires sufficient computational resources and inference time, making it hard to deploy on resource-constrained edge devices. A promising direction for future work is to design multi-LLM systems that optimally balance performance and efficiency.

Lack of Optimization in Inference Configuration. In SMCS, all LLMs are queried using the same sampling parameters and prompts. However, a uniform configuration may not be optimal for heterogeneous LLMs within the system. A potential future direction is to tailor prompts and configurations for each LLM individually, which can maximize their capabilities and improve overall system performance.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Anthropic. 2025a. Claude-3.5-sonnet. *URL https://www.anthropic.com/news/claude-3-5-sonnet*.

Anthropic. 2025b. Claude-3.7-sonnet. *URL https://www.anthropic.com/news/claude-3-7-sonnet*.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and 1 others. 2021. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*.

Akhiad Bercovich, Itay Levy, Izik Golan, Mohammad Dabbah, Ran El-Yaniv, Omri Puny, Ido Galil, Zach Moshe, Tomer Ronen, Najeeb Nabwani, and 1 others. 2025. Llama-nemotron: Efficient reasoning models. *arXiv preprint arXiv:2505.00949*.

Zheng Cai, Maosong Cao, Haojiong Chen, Kai Chen, Keyu Chen, Xin Chen, Xun Chen, Zehui Chen, Zhi Chen, Pei Chu, Xiaoyi Dong, Haodong Duan, Qi Fan, Zhaoye Fei, Yang Gao, Jiaye Ge, Chenya Gu, Yuzhe Gu, Tao Gui, and 81 others. 2024. Internlm2 technical report. *Preprint*, arXiv:2403.17297.

Jiawei Chen, Hongyu Lin, Xianpei Han, and Le Sun. 2024a. Benchmarking large language models in retrieval-augmented generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 17754–17762.

Junying Chen, Zhenyang Cai, Ke Ji, Xidong Wang, Wanlong Liu, Rongsheng Wang, Jianye Hou, and Benyou Wang. 2024b. Huatuogpt-o1, towards medical complex reasoning with llms. *Preprint*, arXiv:2412.18925.

Justin Chih-Yao Chen, Sukwon Yun, Elias Stengel-Eskin, Tianlong Chen, and Mohit Bansal. 2025. Symbolic mixture-of-experts: Adaptive skill-based routing for heterogeneous reasoning. *arXiv preprint arXiv:2503.05641*.

Lingjiao Chen, Jared Quincy Davis, Boris Hanin, Peter Bailis, Ion Stoica, Matei Zaharia, and James Zou. 2024c. Are more llm calls all you need? towards scaling laws of compound inference systems. *arXiv preprint arXiv:2403.02419*.

Lingjiao Chen, Matei Zaharia, and James Zou. 2023a. Frugalgpt: How to use large language models while reducing cost and improving performance. *arXiv preprint arXiv:2305.05176*.

Shuhao Chen, Weisen Jiang, Baijiong Lin, James Kwok, and Yu Zhang. 2024d. Routerdc: Query-based router by dual contrastive learning for assembling large language models. *Advances in Neural Information Processing Systems*, 37:66305–66328.

Xinyun Chen, Renat Aksitov, Uri Alon, Jie Ren, Kefan Xiao, Pengcheng Yin, Sushant Prakash, Charles Sutton, Xuezhi Wang, and Denny Zhou. 2023b. Universal self-consistency for large language model generation. *arXiv preprint arXiv:2311.17311*.

Sanjiban Choudhury. 2025. Process reward models for llm agents: Practical framework and directions. *arXiv preprint arXiv:2502.10325*.

DeepSeek-AI. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

Tao Feng, Yanzhen Shen, and Jiaxuan You. 2025. Graphrouter: A graph-based router for llm selections. *Preprint*, arXiv:2410.03834.

Team GLM, Aohan Zeng, Bin Xu, Bowen Wang, Chenhui Zhang, Da Yin, Diego Rojas, Guanyu Feng, Hanlin Zhao, Hanyu Lai, Hao Yu, Hongning Wang, Jiadai Sun, Jiajie Zhang, Jiale Cheng, Jiayi Gui, Jie Tang, Jing Zhang, Juanzi Li, and 37 others. 2024. Chatglm: A family of large language models from glm-130b to glm-4 all tools. *Preprint*, arXiv:2406.12793.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Lin Gui, Cristina Gârbacea, and Victor Veitch. 2024. Bonbon alignment for large language models and the sweetness of best-of-n sampling. *arXiv preprint arXiv:2406.00832*.

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*.

Chi Hu, Chenglong Wang, Xiangnan Ma, Xia Meng, Yinqiao Li, Tong Xiao, Jingbo Zhu, and Changliang Li. 2021. Ranknas: Efficient neural architecture search by pairwise ranking. *arXiv preprint arXiv:2109.07383*.

Yutong Hu, Quzhe Huang, Mingxu Tao, Chen Zhang, and Yansong Feng. 2024. Can perplexity reflect large language model's ability in long text understanding? *arXiv preprint arXiv:2405.06105*.

Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Kai Dang, and 1 others. 2024. Qwen2. 5-coder technical report. *arXiv preprint arXiv:2409.12186*.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. 2024. Livecodebench: Holistic and contamination free evaluation of large language models for code. *arXiv preprint arXiv:2403.07974*.

Dongfu Jiang, Xiang Ren, and Bill Yuchen Lin. 2023. Llm-blender: Ensembling large language models with pairwise ranking and generative fusion. *arXiv preprint arXiv:2306.02561*.

Wittawat Jitkrittum, Harikrishna Narasimhan, Ankit Singh Rawat, Jeevesh Juneja, Zifeng Wang, Chen-Yu Lee, Pradeep Shenoy, Rina Panigrahy, Aditya Krishna Menon, and Sanjiv Kumar. 2025. Universal model routing for efficient llm inference. *arXiv preprint arXiv:2502.08773*.

Jihoon Kwon Sangmo Gu Yejin Kim, Minkyung Cho Jy-yong Sohn Chanyeol, Choi Junseong Kim, and Seolhwa Lee. 2024. Linq-embed-mistral: Elevating text retrieval with improved gpt data through task-specific control and quality refinement. linq ai research blog.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, and 1 others. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474.

LG AI Research. 2025. Exaone deep: Reasoning enhanced language models. *arXiv preprint arXiv:2503.12524*.

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

Wenzhe Li, Yong Lin, Mengzhou Xia, and Chi Jin. 2025. Rethinking mixture-of-agents: Is mixing different large language models beneficial? *arXiv preprint arXiv:2502.00674*.

Keming Lu, Hongyi Yuan, Runji Lin, Junyang Lin, Zheng Yuan, Chang Zhou, and Jingren Zhou. 2023. Routing to the expert: Efficient reward-guided ensemble of large language models. *arXiv preprint arXiv:2311.08692*.

MAA. 2024. American invitational mathematics examination. *https://maa.org/math-competitions/american-invitational-mathematics-examination-aime.*

OpenAI. 2024. Gpt-o3-mini [online]. *Available: https://platform.openai.com/docs/models*.

OpenAI. 2025. Introducing gpt-4.1 in the api. *Accessed: 2025-05-07*.

Ankit Pal, Logesh Kumar Umapathi, and Malaikannan Sankarasubbu. 2022. Medmcqa: A large-scale multi-subject multi-choice dataset for medical domain question answering. In *Conference on health, inference, and learning*, pages 248–260. PMLR.

Bardia Panahbehagh, Raphaël Jauslin, and Yves Tillé. 2021. Sequential unequal probability sampling for stream population. *arXiv preprint arXiv:2111.08433*.

Constituency Parsing. 2009. Speech and language processing. *Power point slides*, page 20.

Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. 2023. Yarn: Efficient context window extension of large language models. *arXiv preprint arXiv:2309.00071*.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. 2024. Gpqa: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*.

Tal Shnitzer, Anthony Ou, Mírian Silva, Kate Soule, Yuekai Sun, Justin Solomon, Neil Thompson, and Mikhail Yurochkin. 2023. Large language model routing with benchmark datasets. *arXiv preprint arXiv:2309.15789*.

KV Srivatsa, Kaushal Kumar Maurya, and Ekaterina Kochmar. 2024. Harnessing the power of multiple minds: Lessons learned from llm routing. *arXiv preprint arXiv:2405.00467*.

Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, and 1 others. 2024. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*.

Qwen Team. 2024a. Qwen2.5-32b-instruct.

Qwen Team. 2024b. Qwen2.5: A party of foundation models.

Qwen Team. 2024c. Qwq: Reflect deeply on the boundaries of the unknown.

Qwen Team. 2025. Qwq-32b: Embracing the power of reinforcement learning.

Junlin Wang, Jue Wang, Ben Athiwaratkun, Ce Zhang, and James Zou. 2024a. Mixture-of-agents enhances large language model capabilities. *arXiv preprint arXiv:2406.04692*.

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*.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, and 1 others. 2024b. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Zihan Wang, Xinzhang Liu, Shixuan Liu, Yitong Yao, Yuyao Huang, Zhongjiang He, Xuelong Li, Yongxiang Li, Zhonghao Che, Zhaoxi Zhang, Yan Wang, Xin Wang, Luwen Pu, Huihan Xu, Ruiyu Fang, Yu Zhao, Jie Zhang, Xiaomeng Huang, Zhilong Lu, and 17 others. 2024c. Telechat technical report. *Preprint*, arXiv:2401.03804.

Yixing Xu, Yunhe Wang, Kai Han, Yehui Tang, Shangling Jui, Chunjing Xu, and Chang Xu. 2021. Renas: Relativistic evaluation of neural architecture search. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4411–4420.

YAMING YU. 2012. On the inclusion probabilities in some unequal probability sampling plans without replacement. *Bernoulli*, pages 279–289.

Yi-Kai Zhang, De-Chuan Zhan, and Han-Jia Ye. 2025. Capability instruction tuning: A new paradigm for dynamic llm routing. *Preprint*, arXiv:2502.17282.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. 2023. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*.

A Appendix

A.1 Dataset Details

In the experimental section, we evaluate our proposed SMCS framework across eight diverse benchmarks spanning mathematical reasoning, complex question answering, instruction following, and code generation tasks. Specifically, we construct a balanced test set of 1,196 college-level multidisciplinary questions in MMLU-Pro (Wang et al., 2024b) through stratified sampling from the original test set, with 5,512 remaining questions allocated to validation. For GPQA (Rein et al., 2024), the diamond subset (graduate-level science questions) serves as our test set, while the remaining data forms the validation set. For MedMCQA (Pal et al., 2022), 1,200 medical professional questions are randomly selected for testing, with 1,000 questions reserved for validation. MATH-500 (Hendrycks et al., 2021) subset is used for testing, complemented by 1,000 randomly sampled validation questions from the original dataset. We employ AIME2024 (MAA, 2024) as our test set and historical problems (1983-2023) for validation. For IFEval (Zhou et al., 2023) dataset, 300 instruction-following instances are randomly selected for testing, with 241 instances for validation. The original test set of MBPP (Austin et al., 2021) is preserved for evaluation, while the training and validation sets are combined to form validation. LiveCodeBench (Jain et al., 2024) v5 serves as test set, with v6 reserved for validation purposes.

A.2 LLM Bank Details

To achieve an optimal balance between model diversity and computational efficiency, we carefully curate a collection of 15 mid-sized open-source LLMs (from 20B to 72B) from various architectural families. Specifically, the selected LLMs include: Qwen2.5-32B-Instruct (Team, 2024b), Qwen-2.5-72B-Instruct (Team, 2024b), Qwen2.5-Coder-32B-Instruct (Hui et al., 2024), Qwen3-32B (Team, 2025), GLM-Z1-32B-0414 (GLM et al., 2024), DeepSeek-R1-Distill-Qwen-32B (DeepSeek-AI, 2025), DeepSeek-R1-Distill-Llama-70B (DeepSeek-AI, 2025), QwQ-32B (Team, 2024c), Gemma-3-27b-it (Team et al., 2024), TeleChat2-35B-32K (Wang et al., 2024c), InternLM2.5-20B-Chat (Cai et al., 2024), Llama-3.3-70B-Instruct (Grattafiori et al., 2024), Llama-3.3-Nemotron-Super-49B-v1 (Bercovich et al., 2025), HuatuoGPT-o1-72B (Chen et al., 2024b), EXAONE-Deep-32B (LG AI Research, 2025). As shown in Table 5, our selection encompasses: (1) instruction-tuned variants, and (2) deep thinking models. This strategic composition ensures comprehensive coverage of different capabilities while maintaining manageable computational requirements.

Name	Size	Type
GLM-Z1-32B-0414 (GLM et al., 2024)	32B	Deep Thinking
Qwen-2.5-72B-Instruct (Team, 2024b)	72B	Instruction-tuned
DeepSeek-R1-Distill-Llama-70B (DeepSeek-AI, 2025)	70B	Deep Thinking
QwQ-32B (Team, 2024c)	32B	Deep Thinking
Gemma-3-27b-it (Team et al., 2024)	27B	Instruction-tuned
Qwen2.5-32b-Instruct (Team, 2024a)	32B	Instruction-tuned
TeleChat2-35B-32K (Wang et al., 2024c)	35B	Instruction-tuned
Llama-3.3-70B-Instruct (Grattafiori et al., 2024)	70B	Instruction-tuned
EXAONE-Deep-32B (LG AI Research, 2025)	32B	Deep Thinking
Qwen2.5-Coder-32B-Instruct (Hui et al., 2024)	32B	Instruction-tuned
Qwen3-32B (Team, 2025)	32B	Deep Thinking
Llama-3.3-Nemotron-Super-49B-v1 (Grattafiori et al., 2024)	49B	Deep Thinking
DeepSeek-R1-Distill-Qwen-32B (DeepSeek-AI, 2025)	32B	Deep Thinking
HuatuoGPT-o1-72B (Chen et al., 2024b)	72B	Deep Thinking
InternLM2.5-20B-Chat (Cai et al., 2024)	20B	Instruction-tuned

Table 5: The details of the used LLM bank.

A.3 Implementation Details

Inference Configs. For a fair comparison, we adopt the same inference configs for all experiments. Specifically, we utilize VLLM (Kwon et al., 2023) as the framework for LLM inference. For the sampling parameters of LLM inference, we set the temperature to 0.7. The maximum length of output tokens is 8,192 to avoid extremely long responses. We also set the presence penalty to 1.05 to avoid endless repetition. If the length of context tokens exceeds the limitation of an LLM, the YaRN (Peng et al., 2023) method is used to extend the context window. Moreover, we use Linq-Embed-Mistral (Kim et al., 2024) as the embedding model in all experiments, and the embedding dimension is 8,192.

Hyperparameters. For all SMCS experiments, we use nearly the same hyperparameters to ensure consistency and fair comparison. Specifically, we set the number of referencers K as 7. The base retrieval number N^{sup_base} is 400 while the tolerance threshold coefficient $\gamma = 0.95$. The dropping number K_{drop} is 1. The number of aggregating $n = 8$. The balance coefficient of PPL score λ is 1.0. **Compared Methods.** In the experiment, in addition to comparing the performance of single LLMs, we also compared six popular multi-LLMs collaboration methods, and the experimental settings are as follows: Symbolic_MOE* (Chen et al., 2025) retains its original model profiling and LLM selection framework while employing Llama-3.3-70B-Instruct for final response aggregation.

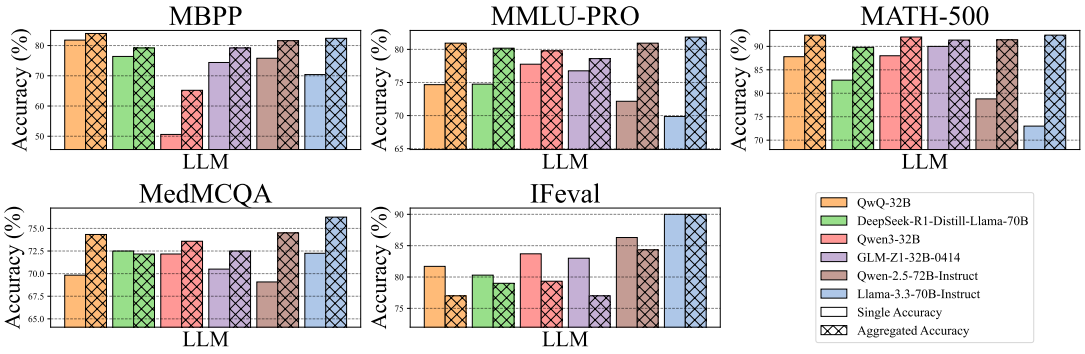


Figure 7: Analysis on aggregator selection with six LLMs across five standard benchmarks

MoA (Wang et al., 2024a) employs 15 LLMs as references, also utilizing Llama-3.3-70B-Instruct as the aggregator. For both Self-MoA (Li et al., 2025) and Self-Consistency (Chen et al., 2023b), we utilize each dataset's best LLM to generate 8 responses per query. Simple Router directly employs the best-performing LLM from each dataset's question bank for response generation. Majority Voting (Chen et al., 2024c) determines the final output through voting among 15 reference LLMs.

A.4 Aggregator Selection

In our SMCS framework, the aggregator plays a pivotal role in consolidating responses from multiple LLMs to generate optimal outputs. To identify the most effective aggregator, we conducted systematic experiments evaluating 6 LLMs as potential aggregators across five diverse benchmarks and the results are shown in Figure 7. Our analysis revealed that Llama-3.3-70B-Instruct demonstrated consistently superior performance across all datasets, leading to its adoption as our default aggregator. In addition, two key insights emerged from this experiment: First, we observed a dissociation between single-LLM performance and its aggregation capability on common benchmarks. For instance, while Qwen3-32B outperformed Llama-3.3-70B-Instruct by +8% on MMLU-PRO, the latter showed significantly better aggregation performance (+4% over Qwen3-32B). (2) However, we also identified a positive correlation between single-LLM performance and aggregation capability on the IFEval benchmark. This correlation stems from IFEval's focus on instruction-following tasks, suggesting that optimal aggregator selection should prioritize LLMs with strong instruction-following abilities to maximize MACS performance.

A.5 Experiments on More Output Tokens

To further explore the potential of the proposed SMCS with the existing open-source LLMs, we only extend the maximum length of output tokens of referencers and aggregators from 8,192 to 32,768 while retaining the other experiment settings for complex reasoning questions. For a fair and accurate comparison, we also extend the maximum length of output tokens of other LLMs to 32,768. It is worth noting that because non-deep-thinking LLMs respond to questions with fewer output tokens (fewer than 8,192 tokens), we directly utilize the results with 8,192 output tokens for non-deep-thinking LLMs as a comparison. Besides, different from the experimental settings in the manuscript, in coding tasks including MBPP and LiveCodeBench, QwQ-32B is adopted as the aggregator for better performance, while Llama-3.3-70B-Instruct is utilized in other tasks. As shown in Table 6, with more output tokens, SMCS also maintains remarkable superiority compared with other open-source and closed-source single LLMs. Specifically, based on fifteen mid-sized open-source LLMs, SMCS can surpass the flagship closed-source LLMs GPT-4.1 by 9.59% and GPT-o3-mini by 9.51%, respectively. Moreover, under the setting of more output tokens, SMCS can consistently break through the challenging open-source upper bound by 4.68% and closed-source upper bound by 6.27%, which demonstrates that SMCS has the potential to push the upper bound of intelligence using multi-LLM collaboration.

A.6 Statistical Analysis

To provide statements about statistical significance, we conduct repetitive experiments on four datasets, including LiveCodeBench, MMLU-Pro, GPQA-Diamond, and MedMCQA. Each setting is run

Model	AIME	MATH-500	MBPP	LiveCodeBench	GPQA-Diamond	MMLU-PRO	IFEval	MedMCQA	Avg
Close-source LLMs									
GPT-o3-mini(2025-01-31) (OpenAI, 2024)	73.33	84.40	62.00	54.70	66.67	74.00	82.00	74.92	71.50
Claude-3.7-Sonnet(2025-02-19) (Anthropic, 2025b)	26.70	73.20	75.40	41.30	63.64	69.43	88.00	74.75	64.05
GPT-4o(2024-08-06) (Achiam et al., 2023)	10.00	74.60	74.20	29.80	52.53	73.83	82.30	76.17	59.18
Claude-3.5-Sonnet(2024-06-20) (Anthropic, 2025a)	16.70	74.20	75.80	34.30	61.62	78.34	80.30	76.00	62.16
GPT-4.1(2025-04-14) (OpenAI, 2025)	50.00	85.80	79.20	42.20	67.17	80.43	86.00	80.58	71.42
Close-source Average	35.34	78.44	73.32	40.46	62.33	75.21	83.72	76.48	65.66
Open-source LLMs									
GLM-Z1-32B-0414 (GLM et al., 2024)	73.33	92.2	76.20	56.50	62.12	77.84	84.30	71.08	74.20
Qwen-2.5-72B-Instruct (Team, 2024b)	16.70	80.8	77.40	26.10	46.97	72.16	86.30	69.08	59.43
DeepSeek-R1-Distill-Llama-70B (DeepSeek-AI, 2025)	76.70	86.00	78.40	57.80	60.60	76.09	80.70	73.17	73.68
QwQ-32B (Team, 2024c)	80.00	91.80	85.20	59.60	62.63	78.34	82.30	70.75	76.33
Gemma-3-27b-it (Team et al., 2024)	30.00	84.00	70.40	27.70	50.51	65.47	81.00	64.58	59.21
Qwen2.5-32b-Instruct (Team, 2024a)	20.00	75.60	76.00	24.00	40.91	69.15	78.70	62.92	55.91
TeleChat2-35B-32K (Wang et al., 2024c)	10.00	70.00	70.00	19.50	33.33	67.98	82.00	57.08	51.24
InternLM2.5-20B-Chat (Cai et al., 2024)	3.30	55.20	55.00	14.90	34.85	44.23	64.70	51.92	40.51
Llama-3.3-70B-Instruct (Grattafiori et al., 2024)	30.00	73.00	70.40	30.10	46.97	69.87	90.00	72.25	60.32
EXAONE-Deep-32B (LG AI Research, 2025)	73.33	92.20	80.60	58.10	63.13	70.48	76.30	60.83	71.87
Qwen2.5-Coder-32B-Instruct (Hui et al., 2024)	16.70	73.60	78.00	27.70	41.92	61.79	80.30	57.25	54.66
Qwen3-32B (Team, 2025)	80.00	92.80	53.20	64.10	64.65	77.76	83.00	70.92	73.30
Llama-3.3-Nemotron-Super-49B-v1 (Grattafiori et al., 2024)	16.70	75.20	65.40	28.00	48.48	67.47	82.70	70.92	56.86
DeepSeek-R1-Distill-Qwen-32B (DeepSeek-AI, 2025)	70.00	85.60	83.40	58.10	57.58	75.17	74.30	67.75	71.49
HuatuoGPT-o1-72B (Chen et al., 2024b)	13.33	73.20	78.00	24.30	52.53	74.16	74.00	76.00	58.19
Open-source Average	40.67	80.11	73.09	38.37	51.45	69.86	80.04	66.18	62.47
Ours v.s. Strong Baselines									
Open-source Upper Bound	80.00	92.80	85.20	64.10	64.65	78.34	90.00	76.00	76.33
SMCS(ours)	**86.67**	**94.50**	**87.00**	**65.65**	66.16	**81.61**	**90.00**	76.50	**81.01**
- *v.s. GPT-4.1*	↑36.67	↑8.70	↑7.80	↑23.45	↓1.01	↑1.18	↑4.00	↓4.08	↑9.59
- *v.s. GPT-o3-mini*	↑13.34	↑10.10	↑25.00	↑10.95	↓0.51	↑7.61	↑8.00	↑1.58	↑9.51
- *v.s. GPT-4o*	↑76.67	↑19.90	↑12.80	↑35.85	↑13.63	↑7.78	↑7.70	↑0.33	↑21.83
- *v.s. Claude-3.7-Sonnet*	↑59.97	↑21.30	↑11.60	↑24.35	↑2.52	↑12.18	↑2.00	↑1.75	↑16.96

Table 6: Main Results on eight mainstream benchmarks using 32,768 maximum output tokens.

Datasets	LiveCodeBench	MMLU-PRO	GPQA-Diamond	MedMCQA
SMCS	52.17±0.46	82.05±0.13	64.81±0.58	75.69±0.42

Table 7: The statistical analysis of the proposed SMCS on four datasets. Each setting is run three times under different random seeds.

RPS	MPS	PPL	Prior Drop	MMLU-PRO	MedMCQA	MATH	MBPP
✗	✗	✗	✗	79.60	73.08	87.8	82.00
✗	✓	✗	✗	80.27	74.00	90.00	82.2
✓	✗	✗	✗	81.10	75.08	91.80	82.40
✓	✓	✗	✗	81.43	75.16	91.80	82.40
✓	✓	✓	✗	81.52	75.42	92.40	82.60
✓	✓	✓	✓	82.02	75.75	92.60	82.80

Table 8: Component ablation on four standard datasets. RPS: Retrieval-based Prior Selection; MPS: Mean Pairwise Similarity; PPL: Perplexity.

three times using the hyperparameters in A.3 under different random seeds. As shown in Table 7, SMCS achieves high mean performance across all four datasets, comparable to the results in Table 1, which demonstrates its ability to deliver consistently strong performance. Furthermore, it can be observed that the standard deviation of SMCS is below 0.6, indicating its superior stability across various settings.

A.7 Component Ablation

We perform a comprehensive component-wise ablation study on four standard benchmarks to quantify the contribution of each component in our SMCS framework. As shown in Table 8, the baseline achieves 79.60% accuracy on MMLU-PRO. Adding the Major Similarity and RPS modules improves performance by +0.67% and +1.5%, respectively, reaching 81.43% when combined. Further gains come from PPL Filtering and Prior Drop, each contributing an additional +0.5%. Similar improvements are observed on MedMCQA, MATH, and MBPP, confirming the effectiveness of each component in enhancing multi-agent collaboration.

A.8 Prompts

To maximize task-specific performance across diverse benchmarks, we developed customized prompt designs for each of the eight evaluation benchmarks, aligning with their distinctive characteristics, as illustrated in Fig. 8. In addition, we elaborated on the prompt design for the aggregator within our SMCS framework by drawing inspiration from the aggregator prompt strategy proposed in MOA (Wang et al., 2024a), as shown in Fig. 9.

Prompt Design for AIME benchmark

System Prompt: "Please reason step by step, and put your final answer within \\boxed{}."
User Prompt: "Question: {question}."

Prompt Design for MATH benchmark

System Prompt: "You are a math problem solver. Please solve the following math problem. Be sure to explain your solution in detail. The numerical values in the answer should be surrounded by \\boxed. The final answer should start with 'The answer is' and give the conclusion directly. Do not add any extra content."
User Prompt: "Question: {question}."

Prompt Design for MBPP benchmark

System Prompt: "You are an exceptionally intelligent coding assistant that consistently delivers accurate and reliable responses to user instructions."
User Prompt: "Question: {question}."

Prompt Design for LiveCodeBench benchmark

System Prompt: "You are an expert Python programmer. You will be given a question (problem specification) and will generate a correct Python program that matches the specification and passes all tests."
User Prompt: "Question: {question}."

Prompt Design for GPQA benchmark

System Prompt: "You are a very intelligent assistant, who follows instructions directly."
User Prompt: "Question: {question}."

Prompt Design for MMLU-PRO benchmark

System Prompt: "The following are multiple choice questions (with answers) about . Think step by step and then output the answer in the format of "The answer is (X)" at the end."
User Prompt: "Question: {question}."

Prompt Design for IFEval benchmark

User Prompt: "Instruction: {question}."

Prompt Design for MedMCQA benchmark

System Prompt: "Provide your step-by-step reasoning first, and then print "The answer is (X)", where X is the answer choice (one capital letter), at the end of your response."
User Prompt: "Question: {question}."

Figure 8: Prompt Design for eight diverse benchmarks within our SMCS framework.

Prompt Design for Aggregator

System Prompt: "You have been provided with a set of responses from various open-source models to the latest user query. Your task is to synthesize these responses into a single, high-quality response. It is crucial to critically evaluate the information provided in these responses, recognizing that some of it may be biased or incorrect. Your response should not simply replicate the given answers but should offer a refined, accurate, and comprehensive reply to the instruction. Ensure your response is well-structured, coherent, and adheres to the highest standards of accuracy and reliability.
Responses from models:
1.{Response1}
2.{Response2}
... "
User Prompt: "Question: {question}."

Figure 9: Prompt Design for Aggregator within our SMCS, inspired by MoA (Wang et al., 2024a).