CoLM: Collaborative Large Models via A Client-Server Paradigm

Siqi Huang[1,2], **Sida Huang**[1,2], **Hongyuan Zhang**[1,3] *

[1]Institute of Artificial Intelligence (TeleAI), China Telecom
[2] School of Artificial Intelligence, OPtics and ElectroNics (iOPEN), Northwestern Polytechnical University
[3]The University of Hong Kong
{4777huang, sidahuang2001, hyzhang98}@gmail.com

Abstract

Large models have achieved remarkable performance across a range of reasoning and understanding tasks. Prior work often utilizes model ensembles or multi-agent systems to collaboratively generate responses, effectively operating in a server-to-server paradigm. However, such approaches do not align well with practical deployment settings, where a limited number of server-side models are shared by many clients under modern internet architectures. In this paper, we introduce **CoLM** (**Co**llaboration in **L**arge-**M**odels), a novel framework for collaborative reasoning that redefines cooperation among large models from a client-server perspective. Unlike traditional ensemble methods that rely on simultaneous inference from multiple models to produce a single output, CoLM allows the outputs of multiple models to be aggregated or shared, enabling each client model to independently refine and update its own generation based on these high-quality outputs. This design enables collaborative benefits by fully leveraging both client-side and shared server-side models. We further extend CoLM to vision-language models (VLMs), demonstrating its applicability beyond language tasks. Experimental results across multiple benchmarks show that CoLM consistently improves model performance on previously failed queries, highlighting the effectiveness of collaborative guidance in enhancing single-model capabilities.

Introduction

Large language models (LLMs) (Brown et al. 2020; Achiam et al. 2023; Liu et al. 2024a; Yang et al. 2025) and vision-language models (VLMs) (Hurst et al. 2024; Li et al. 2023b; Bai et al. 2025) have demonstrated impressive capabilities across a wide range of tasks, including language understanding, logical reasoning, code generation, and multimodal question answering. Their performance often improves with scale, making them a foundation for many state-of-the-art systems in artificial intelligence. However, growing evidence from recent empirical studies and deployment experiences suggests that no single model can consistently dominate across all domains, task types, or input distributions (Labrak, Rouvier, and Dufour 2024; Gretz et al. 2023; Xu et al. 2024). Even state-of-the-art models may struggle with out-of-distribution inputs or domain-specific requirements. To further enhance the reasoning ability of large models, prior work has explored collaborative approaches, such as model ensembles, which aggregate outputs from multiple models, and multi-agent frameworks, where multiple specialized or redundant agents communicate to solve a problem jointly (Wang et al. 2024a; Li et al. 2025; Wang et al. 2025). These methods have shown promising results in boosting accuracy, robustness, and coverage, particularly in tasks where a single model might fail due to uncertainty or lack of context.

However, most existing collaborative paradigms rely on a server-to-server collaboration assumption, where multiple large models can communicate freely and synchronously during inference. While these methods are effective in controlled or offline environments, it becomes impractical for large-scale deployment of LLMs and VLMs over the internet, where computational resources are limited and user access is typically routed through shared servers. These practical constraints call for a fundamental reevaluation of how collaborative reasoning should be designed in real-world systems. In many deployment scenarios such as mobile applications, edge devices, or shared computing clusters, a vast number of clients must interact with only a small number of centralized, high-capacity server models. Under such conditions, these methods are incompatible with the nature of client-server architectures, limiting their practicality.

To address these challenges, we propose **CoLM** (**Co**llaboration in **L**arge-**M**odels), a novel framework that redefines collaborative reasoning from a client-server perspective. Unlike traditional ensemble methods that directly produce joint outputs, CoLM leverages intermediate outputs generated by multiple models running on distributed clients, which are then aggregated and refined by a more capable server-side model. The resulting guidance is sent back to the client, where typically lightweight or privacy-constrained target models utilize this information to generate the final responses. By separating heavy reasoning from local generation, CoLM enables efficient collaboration that improves client-side performance without incurring the computational cost of ensemble inference or requiring full server-side decoding.

For language tasks, we adopt a three-stage client-server paradigm. Domain-specialized client models generate refer-

*: Corresponding authors.

ence responses independently, which are then synthesized by a central server model to produce a global answer. This answer is subsequently returned to the clients as guidance, enabling them to refine their final responses.

Vision-language models are often trained on diverse multimodal datasets and exhibit varied strengths and biases across tasks. Rather than forcing ensemble decoding, CoLM for VLMs uses a prompt-based collaboration strategy: the outputs from multiple VLMs are concatenated as contextual input to guide a model. This design naturally supports task-level diversity and allows each model to contribute complementary perspectives, resulting in more robust and accurate final outputs. Our experiments demonstrate that models with complementary strengths can collaborate to guide a target vision-language model, significantly improving accuracy especially on challenging queries where standalone models often fail. This demonstrates the potential of client-server collaboration to advance the capabilities and applicability of large models.

Related Work

Ensemble and Collaborative Reasoning in Large Models LLMs have made significant progress in reasoning tasks with prompting techniques such as Chain-of-Thought (CoT)(Kojima et al. 2022; Wei et al. 2022) and Self-Consistency(Wang et al. 2022), which promote step-by-step thinking and improve answer reliability through path sampling. In parallel, recent studies have explored multi-agent collaboration to enhance LLM reasoning. A common direction involves debate-style frameworks, where multiple models interact through iterative discussion or voting to arrive at better answers (Du et al. 2023; Liang et al. 2023). Several works have shown that introducing constructive noise into the model inputs or intermediate representations can enhance model robustness and generalization (Li 2022; Zhang et al. 2025, 2024a; Huang et al. 2025a; Jiang et al. 2025). Other approaches, such as Multi-Agent (MoA) and Self-MoA (Wang et al. 2024a; Li et al. 2025), improve prediction by aggregating responses from multiple rounds of model interaction. While these systems rely on iterative collaboration, they do not aim to improve or guide a specific target model.

Routing and Cascading Inference for Cost Efficiency To reduce the cost of LLM deployment, routing and cascading methods have been widely explored. Routing methods like RouteLLM (Ong et al. 2024) and Eagle (Zhao, Jin, and Mao 2024) aim to dynamically select the most appropriate model per input query. Cascading methods instead involve sequential invocation of models based on response confidence or quality thresholds. FrugalGPT (Chen, Zaharia, and Zou 2024), for example, uses a judging model to determine if the current model's output is sufficient, invoking stronger models only when needed. Other works explore policy learning (Zhang et al. 2024b) or structured representations (Yue et al. 2024a) to optimize cascade decisions. While effective in saving cost, these approaches still incur multiple inference calls and primarily focus on model switching, not on improving a given model's capability.

Client-Server and Distributed Inference Some efforts have explored distributed inference strategies to balance latency and computation in real-world systems. Neurosurgeon (Kang et al. 2017) and DDNN (Teerapittayanon, McDanel, and Kung 2017) propose splitting models between edge and cloud for collaborative computation. Autosplit (Banitalebi-Dehkordi et al. 2021) further generalizes this concept into a practical framework for edge-cloud AI deployments. A comprehensive survey (Wang et al. 2024b) reviews recent advances in end-edge-cloud collaborative deep learning, highlighting challenges and system-level design considerations. These approaches align with our CoLM design, which leverages limited server-side interaction to guide lightweight client-side inference, enabling scalable, efficient collaboration in practical deployments.

Collaborative Reasoning in Multimodal Settings Some works have explored using multiple agents with distinct capabilities to tackle complex multi-modal tasks. Multi-Agent VQA (Jiang et al. 2024) employs a cooperative setup where a central vision-language model offloads subtasks like object detection or counting to specialized models. Similarly, BuboGPT (Zhao et al. 2023) integrates an off-the-shelf grounding module into a multimodal LLM to enhance fine-grained object localization during response generation. MMCTAgent (Kumar et al. 2024) further advances this approach by introducing a critic module and iterative reasoning loops, mimicking human critical thinking to refine complex visual answers. Recent works demonstrate that introducing beneficial noise into multimodal representations can improve alignment and generation quality (Huang, Zhang, and Li 2025; Huang et al. 2025b; Wang, Zhang, and Yuan 2025; Fu et al. 2025). More recently, MAMMQA (Rajput et al. 2025) generalizes multi-agent collaboration to handle text, tables, and images jointly, where dedicated VLM and LLM agents sequentially decompose, synthesize, and integrate modality-specific insights.

Method

Our proposed **CoLM** is designed to facilitate efficient and structured collaboration between models deployed in a client-server architecture. This design closely aligns with practical deployment scenarios where users interact with AI models running on resource-constrained client devices such as smartphones, tablets, or edge computing nodes.

Motivation

In real-world deployment scenarios, client-side models are highly customized, either through domain-specific or long-term adaptation to particular user behaviors. These models run on personal devices or localized environments and accumulate specialized knowledge and exhibit strong domain preferences. They distinct from one another not just in scale, but also in perspective and reasoning habits.

Unlike traditional ensemble or agent systems which typically rely on multiple models to reach a consensus, our setting embraces the diversity among client models. This diversity is not a source of noise but rather a valuable feature.

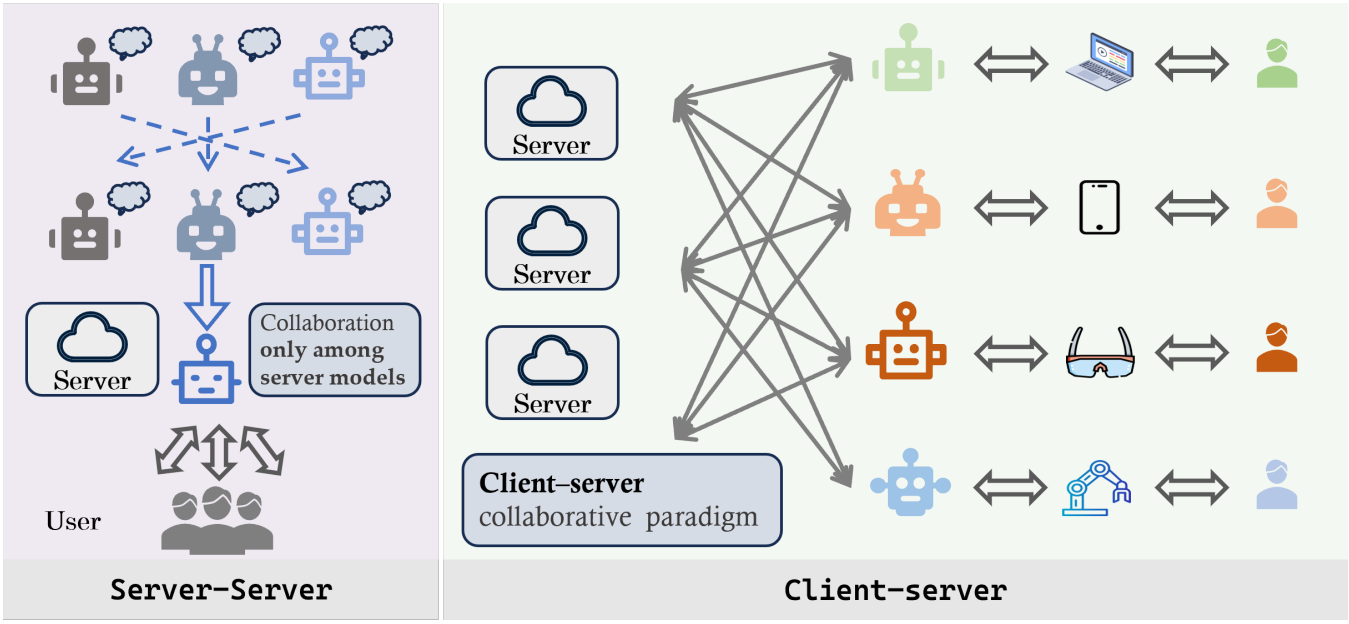

Figure 1: **Left:** Traditional server-to-server collaboration paradigm, where multiple large models interact directly during inference. These approaches often rely on interactions among general-purpose models, lacking specialization structure. **Right:** Our proposed client-server collaboration paradigm, where lightweight client models receive guidance from shared server-side models. This design allows each client to maintain long-lived, domain-specific expertise while improving response quality through collaboration.

Each model reflects a unique domain expertise or personalization history. In our setup, client models first independently produce responses based on their specialized understanding. These responses are then sent to a central server model, which integrates them to generate guidance. Then each client model can use to revise and refine its own answer.

This interaction loop encourages models to not only contribute their strengths but also evolve through exposure to alternative perspectives. It allows underperforming models to benefit from others' knowledge, while still preserving their personalized traits. In doing so, CoLM enables a richer form of collaboration that improves robustness and generalization. As shown in our experiments (Section), CoLM achieves a strong balance between collaboration efficiency and performance, demonstrating its practicality and broad potential for real-world deployment.

The CoLM Inference Pipeline

CoLM supports collaborative reasoning in both language and vision-language scenarios. While both share the same guiding principle, the actual inference pipelines differ due to architectural differences between LLMs and VLMs.

For LLM models, CoLM organizes models into two asymmetric roles: lightweight client models that generate reference responses, and a central server model responsible for synthesizing and producing the final output.

Given a user query q, the inference process begins by identifying which models in a larger pool are most relevant to the query. Let $\mathcal{C} = \{M_1, M_2, \ldots, M_K\}$ denote the complete set of available client models. Each client M_i is associated with a specialization prompt $\mathcal{P}(M_i)$ that describes its intended domain or behavioral role (e.g., "You are an expert in math"). We use a strong general-purpose language model (e.g., GPT-4o) to estimate the semantic similarity between q and each $\mathcal{P}(M_i)$, and select the top-k most relevant models to form a task-specific subset $\mathcal{C}^* \subseteq \mathcal{C}$.

Each selected model $M_i \in \mathcal{C}^*$ is then queried independently to generate a domain-specific response $M_i(q)$. These client responses are treated as expert contributions offering diverse knowledge perspectives. These experts may include (1) real, task-optimized models like Qwen-Math and Qwen-Coder, or (2) simulated pseudo-experts created by prompt-based role conditioning of general models. The latter approach allows us to instantiate specialists in domains without further training. Once all expert responses are collected, the server model M_s is tasked with synthesizing them into a final answer. Formally, the answer a is produced as:

$$a = M_s\left(q,\ \{M_i(q) \mid M_i \in \mathcal{C}^*\}\right)$$

Here, $\{M_i(q)\}_{i \in \mathcal{C}^*}$ denotes the collection of expert responses, and the server model M_s conditions on both the original query and these expert outputs to generate a. The aggregation is guided by a prompt that encourages consistency, factual accuracy.

In final stage, every client receives the server's aggregated output, and revises its response accordingly. Rather than producing a single unified answer, this process enables each model to benefit from the shared insights while still tailoring its output to its specialized domain or user preference.

Model	MME-P	MME-R	SEEDBench	MMBench	OCRBench	AI2D	MMMU-Val	MMMU-Dev	Avg. Score
Qwen2.5-VL-7B	**1693.53**	611.43	0.771	**0.831**	**881**	0.809	0.444	0.433	61.88
Qwen2.5-VL-7B*	1656.04	**614.64**↑	**0.772**↑	0.819	865	0.782	**0.532**↑	**0.513**↑	**63.30**↑
Janus-Pro-7B	**1509.38**	270.71	0.701	0.665	584	0.679	0.380	0.373	47.55
Janus-Pro-7B*	1482.28	**434.64**↑	**0.747**↑	**0.773**↑	**800**↑	**0.782**↑	**0.499**↑	**0.460**↑	**58.06**↑
LLaVA-1.5-7B	1340.31	302.14	0.601	0.629	308	0.519	0.323	0.273	38.62
LLaVA-1.5-7B*	**1349.79**↑	**408.93**↑	**0.751**↑	**0.767**↑	**678**↑	**0.775**↑	**0.489**↑	**0.487**↑	**56.13**↑
GPT-4o	1618.96	672.86	0.755	0.813	806	**0.737**	0.569	0.567	63.51
GPT-4o*	**1704.77**↑	**688.93**↑	**0.766**↑	**0.825**↑	**824**↑	0.734	**0.574**↑	**0.580**↑	**64.53**↑

Table 1: Evaluation results of VLMs on multiple benchmarks. Models marked with an asterisk (*) and highlighted rows represent responses generated using our collaborative method. ↑ indicates improved performance compared to the original model. Avg. Score is the average of all scores scaled to a 0–100 range.

For VLMs, we consider a realistic scenario where a single user interacts through multiple devices, each representing a distinct view. Correspondingly, multiple client VLMs are employed, each independently reasoning on the same multimodal query and generating their own answers. These answers often provide complementary perspectives and capture different aspects of the query. Unlike language models, VLMs generally do not possess chain-of-thought reasoning capabilities. Server-side integration is not suitable.

Therefore our collaborative inference strategy involves a two-step process: first, the query is distributed to multiple client VLMs, yielding diverse responses. Then, all responses are concatenated into a structured prompt and fed back to previous VLM models. These models integrate and refine the aggregated information to produce a final, more comprehensive answer. This approach effectively leverages the complementary strengths of multiple VLMs, enabling more accurate and multimodal understanding.

Experiment

We conduct comprehensive experiments to evaluate the effectiveness of the proposed framework across both LLM and VLM tasks. We aim to answer the following key research questions:

- **Q1:** *Can collaboration improve overall response quality compared to standalone generation?*
- **Q2:** *To what extent does each expert model contribute to performance gains?*
- **Q3:** *How does the performance change as the scale of client model becomes larger?*

Experimental Setup

Benchmarks For VLMs, we evaluate our model on widely recognized image-based vision-language benchmarks to assess multimodal understanding capabilities: MME (Fu et al. 2024), SEED Bench (Li et al. 2023a), MMBench (Liu et al. 2024b), AI2D (Kembhavi et al. 2016), OCRBench (Liu et al. 2024c) and MMMU (Yue et al. 2024b).

For LLMs, we adopt three prominent alignment and instruction-following datasets: AlpacaEval 2.0 (Dubois et al. 2024), Arena-Hard (Li et al. 2024), and MT-Bench (Zheng et al. 2023).

Models In our experiments, we employ a selection of widely used, publicly available open-source language models, focusing on those with strong performance across a variety of tasks. Specifically, for the VLM models, we leverage four different models with diverse architectures and training paradigms: **GPT-4o**, **Qwen2.5-VL-7B-Instruct**(Bai et al. 2025), **Janus-Pro-7B**(Chen et al. 2025), and **LLaVA-v1.5-7B** (Liu et al. 2023). All inferences are performed via official APIs or direct model reference. Specifically, GPT-4o is accessed through OpenAI's API, while the remaining models are downloaded from Hugging Face and run locally using open-source inference frameworks.

For the LLM models, we select five models, each representing a distinct domain expertise. Three are expert-tuned variants from the Qwen and DeepSeek teams (Yang et al. 2025; Liu et al. 2024a): **Qwen-Math**, optimized for mathematical reasoning; **Qwen-Coder**, specialized in code generation; and **Deepseek-Math-7B**, another model focused on mathematical tasks. In addition to these, we simulate expert behaviors in general-purpose models through prompt-based role conditioning. For example, **DeepSeek-Creative** (Liu et al. 2024a) is prompted to adopt a creative writing role, while GPT-4o (Achiam et al. 2023) is guided to emulate empathetic dialogue which we called **GPT-Conversational**. To support centralized response synthesis, we use **GPT-4o** as the server model. We select it for its strong reasoning capabilities and consistent cross-domain performance. All model inferences are conducted via official APIs, adhering strictly to licensing terms and usage policies.

Figure 2: Examples of Janus-Pro-7B responses on VQA tasks. Our method enables the model to produce more accurate answers through collaborative inference.

Main Results

VLM Results Table 1 presents the performance of several representative vision-language models (VLMs) across a wide range of benchmarks, covering both perception and reasoning capabilities. Our method leads to **consistent improvements across most tasks**.

A general pattern emerges when grouping models by their original capability. **Relatively weaker models**, such as LLaVA-1.5-7B and Janus-Pro-7B, tend to gain the most. For example, Janus improves notably on reasoning-intensive benchmarks like MMBench and MMMU-Val, while LLaVA shows strong gains on OCRBench and SEEDBench. These models likely benefit from richer, multi-perspective context that helps compensate for their limited perception or reasoning skills. **Stronger models**, such as GPT-4o and Qwen2.5-VL-7B, also benefit, though the improvements are more modest. Since these models already perform near the ceiling on many tasks, the collaboration of models acts more as a refinement than a correction. Still, GPT-4o sees consistent gains on benchmarks like MME and MMMU, suggesting that even high-capacity models can profit from added contextual diversity.

Benchmarks that require complex reasoning such as MMBench, SEEDBench, and MMMU show the most consistent gains. This supports the idea that collaboration serves as a lightweight form of **externalized reasoning**, enabling stronger inference without altering the model itself. For example, the collaboration helps models like LLaVA and Janus focus on semantically important regions or concepts that they might otherwise miss. This is further illustrated in Figure 2, where Janus-Pro-7B fails on several VQA examples, while the enhanced version produces more accurate and grounded answers with collaboration. These cases show that our method improves not only overall scores but also answer quality at the instance level.

LLM Results. Table 2 reports the performance of several large language models (LLMs) on three benchmarks: **MT-Bench**, **AlpacaEval 2.0**, and **Arena-Hard**, which respectively evaluate multi-turn dialogue capability, alignment with human preferences, and challenging reasoning ability. Each model is evaluated both in its original form and an enhanced version (marked with *), where the enhancement corresponds to applying our proposed collaborative client-server mechanism. To further contextualize performance, we also include comparisons against **MoA** (Wang et al. 2024a), a representative collaborative framework, and our centralized **Server Output**, which aggregates responses from all collaborating users. The server output achieves the best overall performance across most metrics, demonstrating the upper bound of collaborative reasoning under full information sharing.

On MT-Bench, which focuses on multi-turn conversational ability, all models benefit from the enhancement, with particularly notable improvements in the second turn scores. This consistent pattern suggests that the collaborative context effectively helps models maintain dialogue coherence across turns, especially enhancing weaker baselines. Interestingly, the average turn score across all models increases significantly, confirming the robustness of the approach. Regarding AlpacaEval 2.0, our approach consistently enhances all evaluated models, improving their alignment with human preferences and producing more fluent, locally consistent outputs. On the more challenging Arena-Hard benchmark, all enhanced models show significant performance boosts. Particularly, models with initially modest results experience substantial improvements, illustrating how our client-server

Model	MT-Bench			AlpacaEval 2.0		Arena-Hard	Avg. Score
	1st Turn	2nd Turn	Avg.	LC Win	Win	Score	
Qwen2.5-Math-7B-Instruct	4.35	3.19	3.77	4.33	3.98	3.02	15.02
Qwen2.5-Math-7B-Instruct*	**6.30**↑	**4.34**↑	**5.34**↑	**14.26**↑	**14.48**↑	**12.48**↑	**26.71**↑
Qwen2.5-Coder-7B-Instruct	3.66	2.34	2.99	15.64	7.26	8.72	18.09
Qwen2.5-Coder-7B-Instruct*	**5.03**↑	**2.83**↑	**3.99**↑	**21.74**↑	**8.28**↑	**54.80**↑	**38.81**↑
Deepseek-Math-7B-Instruct	4.54	3.16	3.85	4.61	2.81	3.48	15.53
Deepseek-Math-7B-Instruct*	**7.36**↑	**5.42**↑	**6.40**↑	**45.08**↑	**32.23**↑	**59.76**↑	**56.28**↑
GPT-Conversation	5.98	5.71	5.84	33.59	45.41	17.00	36.33
GPT-Conversation*	**7.14**↑	**7.24**↑	**7.19**↑	**42.90**↑	**57.56**↑	**67.57**↑	**60.79**↑
DeepSeek-Creative	7.90	7.78	7.84	60.04	**54.56**	61.25	66.56
DeepSeek-Creative*	**8.20**↑	**7.86**↑	**8.03**↑	**60.88**↑	54.43	**80.73**↑	**73.97**↑
MoA	8.36	7.42	7.90	76.78	81.90	**92.96**	82.91
Server output	**8.85**	**7.49**	**8.17**	**77.72**	**82.31**	92.37	**83.93**

Table 2: Results on MT-Bench, AlpacaEval 2.0, and Arena-Hard. Rows with background shading indicate outputs generated by our collaborative method. ↑ denotes improved performance compared to MoA. "Server output" refers to model outputs generated using our server-side method, while "MoA" refers to outputs generated by the same model using the MoA architecture. The comparison highlights the effectiveness of our server-side approach.

collaboration empowers weaker models by effectively leveraging external expertise.

In summary, our **client-server collaborative framework** systematically enhances diverse models, particularly empowering weaker models to better leverage information and guidance from server-side counterparts, validating the practical advantage of distributed collaboration in large models.

Ablation Study

In this section, we analyze three key factors affecting our collaborative framework: individual client contributions, the number of collaborating users, andcollaboration rounds.

Influence of Individual Client Models in Collaboration

To better understand the contribution of each client model, we evaluate a simplified scenario where the server collaborates with only one client model at a time. The results are shown in Table 3. Overall, on most benchmarks, the best performance is achieved only when all client models collaborate together, as single models alone struggle to fully cover the diverse demands of multimodal tasks. Among individual clients, GPT-4o consistently achieves the best standalone performance, especially on complex reasoning tasks. Qwen-VL performs well on MME and SEEDBench. LLaVA performs relatively weaker when used alone, highlighting the necessity of collaborative synergy in more challenging tasks.

These findings demonstrate that collaboration within CoLM is not merely compensatory but synergistic. Each model contributes unique strengths, yet no single client matches the performance of the fully collaborative CoLM setup. This underscores the importance of our client-server architecture and motivates further exploration of adaptive client selection strategies tailored to task requirements.

Model	MME	SEEDBench	MMBench	AI2D
Janus-Pro-7B	270.71	0.701	0.665	0.679
Only Qwen2.5-VL-7B	364.29	0.731	0.685	0.733
Δ	+93.58	+0.030	+0.020	+0.054
Only LLaVA-1.5-7B	276.79	0.653	0.597	0.483
Δ	+6.08	−0.048	−0.068	−0.196
Only GPT-4o	310.71	0.721	0.701	0.742
Δ	+40.00	+0.020	+0.036	+0.063
Janus-Pro-7B*	**434.64**	**0.747**	**0.773**	**0.782**
Δ	**+163.93**	**+0.046**	**+0.108**	**+0.103**

Table 3: Performance comparison of Janus-Pro-7B under different collaboration settings across multimodal benchmarks. Janus-Pro-7B runs without collaboration. Rows labeled "Only [Model]" denote collaboration between Janus-Pro-7B and an additional client model. Janus-Pro-7B* uses all models in collaboration. Δ indicates the absolute performance change relative to Janus-Pro-7B.

Impact of Collaborative User Scale on LLM

We investigate how the number of collaborative users affects the performance of large language models on three representative benchmarks. As shown in Figure 3, increasing the number of participating client models consistently improves

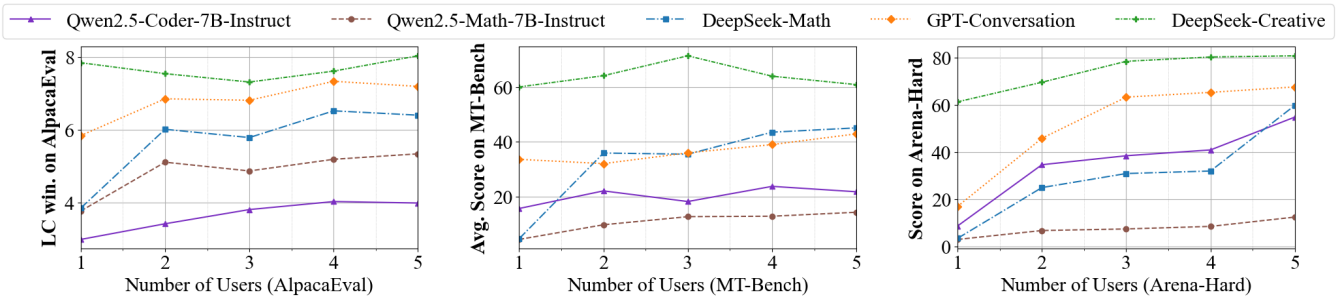

Figure 3: Ablation study on the effect of collaborative user scale on LLM performance. Experiments conducted on three benchmarks show that increasing the number of collaborative clients leads to consistent performance improvement.

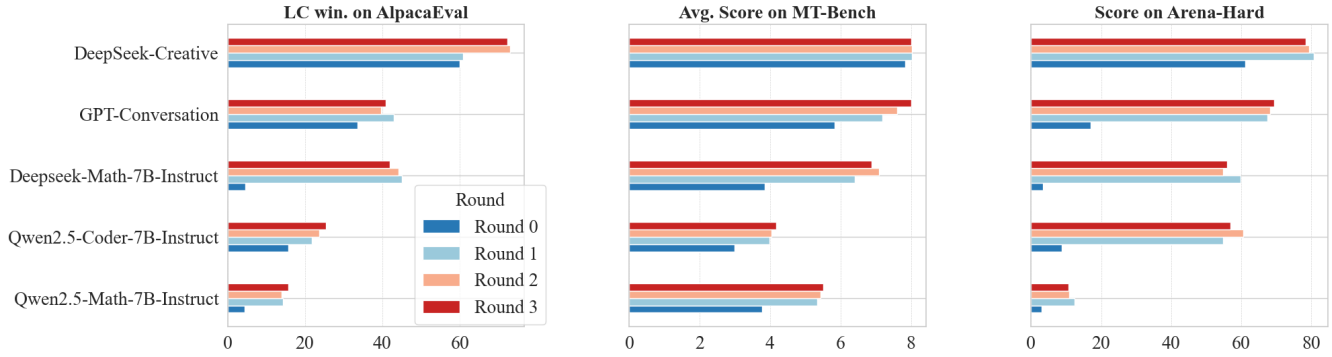

Figure 4: Performance improvements with increasing collaboration rounds across multiple datasets and models. Iterative interaction consistently enhances results, especially for models with weaker initial outputs, but shows diminishing returns after several rounds.

performance across all evaluated tasks. This demonstrates that diversity among expert LLMs provides richer insights and better reasoning capabilities, mirroring the benefits of real-world collaboration where multiple perspectives lead to stronger outcomes.

However, these performance gains exhibit diminishing returns as the number of collaborating clients increases. Beyond a certain point, additional client responses may introduce redundant or conflicting information, making it challenging for the server to extract further useful knowledge. Therefore future work could explore more selective or adaptive integration strategies that prioritize high-quality or complementary inputs, thus improving the efficiency and effectiveness of the aggregation process.

Effect of Collaboration Rounds

Figure 4 illustrates how model performance improves progressively with an increasing number of collaboration rounds across various datasets and models. Iterative interaction allows models, especially those with weaker initial predictions, to refine their outputs by correcting mistakes and integrating complementary knowledge.

Domain-specific models often exhibit rapid gains in the early rounds, leveraging their specialized knowledge effectively. However, the performance improvements tend to plateau after several iterations, indicating diminishing returns. This saturation highlights the practical trade-off between accuracy improvements and additional computational overhead, suggesting the importance of choosing an optimal number of collaboration rounds in real-world applications.

Conclusion

In this work, we introduce CoLM, a client-server paradigm for collaboration in large models. By shifting from traditional server-to-server ensembles to a more practical client-server architecture, CoLM better reflects real-world deployment constraints, where resource-limited user-side models can still benefit from server-side expertise. Moreover, we extended CoLM to vision-language models, showing that collaborative guidance remains effective in multimodal settings. Overall, CoLM provides an efficient and deployment-friendly framework for improving model robustness and performance in both language and vision-language tasks. Currently, our approach is limited by the availability of truly specialized client-side models, as such dedicated expert models are not yet widely deployed. This restricts the diversity and effectiveness of model selection in practice. However, we are optimistic that as more specialized and personalized models become accessible on client devices in the future, the potential and benefits of our method will be greatly enhanced.

References

Achiam, J.; Adler, S.; Agarwal, S.; Ahmad, L.; Akkaya, I.; Aleman, F. L.; Almeida, D.; Altenschmidt, J.; Altman, S.; Anadkat, S.; et al. 2023. GPT-4 Technical Report.

Bai, S.; Chen, K.; Liu, X.; Wang, J.; Ge, W.; Song, S.; Dang, K.; Wang, P.; Wang, S.; Tang, J.; et al. 2025. Qwen2.5-VL technical report. *arXiv preprint arXiv:2502.13923*.

Banitalebi-Dehkordi, A.; Vedula, N.; Pei, J.; Xia, F.; Wang, L.; and Zhang, Y. 2021. Auto-split: A general framework of collaborative edge-cloud AI. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 2543–2553.

Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language Models are Few-Shot Learners.

Chen, L.; Zaharia, M.; and Zou, J. 2024. FrugalGPT: How to Use Large Language Models While Reducing Cost and Improving Performance. *Transactions on Machine Learning Research*.

Chen, X.; Wu, Z.; Liu, X.; Pan, Z.; Liu, W.; Xie, Z.; Yu, X.; and Ruan, C. 2025. Janus-Pro: Unified Multimodal Understanding and Generation with Data and Model Scaling. *arXiv preprint arXiv:2501.17811*.

Du, Y.; Li, S.; Torralba, A.; Tenenbaum, J. B.; and Mordatch, I. 2023. Improving factuality and reasoning in language models through multiagent debate. In *Forty-first International Conference on Machine Learning*.

Dubois, Y.; Galambosi, B.; Liang, P.; and Hashimoto, T. B. 2024. Length-Controlled AlpacaEval: A Simple Way to Debias Automatic Evaluators. *arXiv preprint arXiv:2404.04475*.

Fu, C.; Chen, P.; Shen, Y.; Qin, Y.; Zhang, M.; Lin, X.; Yang, J.; Zheng, X.; Li, K.; Sun, X.; Wu, Y.; and Ji, R. 2024. MME: A Comprehensive Evaluation Benchmark for Multimodal Large Language Models. arXiv:2306.13394.

Fu, Y.; Si, R.; Wang, H.; Zhou, D.; Sun, J.; Luo, P.; Hu, D.; Zhang, H.; and Li, X. 2025. Object-AVEdit: An Object-level Audio-Visual Editing Model. *arXiv preprint arXiv:2510.00050*.

Gretz, S.; Halfon, A.; Shnayderman, I.; Toledo-Ronen, O.; Spector, A.; Dankin, L.; Katsis, Y.; Arviv, O.; Katz, Y.; Slonim, N.; and Ein-Dor, L. 2023. Zero-shot Topical Text Classification with LLMs - an Experimental Study. In Bouamor, H.; Pino, J.; and Bali, K., eds., *Findings of the Association for Computational Linguistics: EMNLP 2023*, 9647–9676. Singapore: Association for Computational Linguistics.

Huang, S.; Xu, Y.; Zhang, H.; and Li, X. 2025a. Learn beneficial noise as graph augmentation. In *Forty-two International Conference on Machine Learning*.

Huang, S.; Zhang, H.; and Li, X. 2025. Enhance vision-language alignment with noise. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, 17449–17457.

Huang, Z.; Qiu, X.; Ma, Y.; Zhou, Y.; Chen, J.; Zhang, H.; Zhang, C.; and Li, X. 2025b. Nfig: Autoregressive image generation with next-frequency prediction. *The Thirty-Ninth Annual Conference on Neural Information Processing Systems (NeurIPS)*.

Hurst, A.; Lerer, A.; Goucher, A. P.; Perelman, A.; Ramesh, A.; Clark, A.; Ostrow, A.; Welihinda, A.; Hayes, A.; Radford, A.; et al. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.

Jiang, B.; Zhuang, Z.; Shivakumar, S. S.; Roth, D.; and Taylor, C. J. 2024. Multi-Agent VQA: Exploring Multi-Agent Foundation Models in Zero-Shot Visual Question Answering. arXiv:2403.14783.

Jiang, K.; Shi, Z.; Zhang, D.; Zhang, H.; and Li, X. 2025. Mixture of Noise for Pre-Trained Model-Based Class-Incremental Learning. In *The Thirty-Ninth Annual Conference on Neural Information Processing Systems (NeurIPS)*.

Kang, Y.; Hauswald, J.; Gao, C.; Rovinski, A.; Mudge, T.; Mars, J.; and Tang, L. 2017. Neurosurgeon: Collaborative intelligence between the cloud and mobile edge. *ACM SIGARCH Computer Architecture News*, 45(1): 615–629.

Kembhavi, A.; Salvato, M.; Kolve, E.; Seo, M.; Hajishirzi, H.; and Farhadi, A. 2016. A Diagram Is Worth A Dozen Images. arXiv:1603.07396.

Kojima, T.; Gu, S. S.; Reid, M.; Matsuo, Y.; and Iwasawa, Y. 2022. Large Language Models are Zero-Shot Reasoners. In *Advances in Neural Information Processing Systems*, volume 35, 22199–22213.

Kumar, S.; Gadhia, Y.; Ganu, T.; and Nambi, A. 2024. Mmctagent: Multi-modal critical thinking agent framework for complex visual reasoning. *arXiv preprint arXiv:2405.18358*.

Labrak, Y.; Rouvier, M.; and Dufour, R. 2024. A Zero-shot and Few-shot Study of Instruction-Finetuned Large Language Models Applied to Clinical and Biomedical Tasks. In Calzolari, N.; Kan, M.-Y.; Hoste, V.; Lenci, A.; Sakti, S.; and Xue, N., eds., *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, 2049–2066. Torino, Italia: ELRA and ICCL.

Li, B.; Wang, R.; Wang, G.; Ge, Y.; Ge, Y.; and Shan, Y. 2023a. SEED-Bench: Benchmarking Multimodal LLMs with Generative Comprehension. arXiv:2307.16125.

Li, J.; Li, D.; Savarese, S.; and Hoi, S. 2023b. BLIP-2: Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models. arXiv:2301.12597.

Li, T.; Chiang, W.-L.; Frick, E.; Dunlap, L.; Wu, T.; Zhu, B.; Gonzalez, J. E.; and Stoica, I. 2024. From crowdsourced data to high-quality benchmarks: Arena-hard and benchbuilder pipeline. *arXiv preprint arXiv:2406.11939*.

Li, W.; Lin, Y.; Xia, M.; and Jin, C. 2025. Rethinking Mixture-of-Agents: Is Mixing Different Large Language Models Beneficial? *arXiv preprint arXiv:2502.00674*.

Li, X. 2022. Positive-incentive noise. *IEEE Transactions on Neural Networks and Learning Systems*, 35(6): 8708–8714.

Liang, T.; He, Z.; Jiao, W.; Wang, X.; Wang, Y.; Wang, R.; Yang, Y.; Shi, S.; and Tu, Z. 2023. Encouraging divergent thinking in large language models through multi-agent debate. *arXiv preprint arXiv:2305.19118*.

Liu, A.; Feng, B.; Xue, B.; Wang, B.; Wu, B.; Lu, C.; Zhao, C.; Deng, C.; Zhang, C.; Ruan, C.; et al. 2024a. Deepseek-v3 technical report.

Liu, H.; Li, C.; Wu, Q.; and Lee, Y. J. 2023. Visual Instruction Tuning. arXiv:2304.08485.

Liu, Y.; Duan, H.; Zhang, Y.; Li, B.; Zhang, S.; Zhao, W.; Yuan, Y.; Wang, J.; He, C.; Liu, Z.; Chen, K.; and Lin, D. 2024b. MMBench: Is Your Multi-modal Model an All-around Player? arXiv:2307.06281.

Liu, Y.; Li, Z.; Huang, M.; Yang, B.; Yu, W.; Li, C.; Yin, X.-C.; Liu, C.-L.; Jin, L.; and Bai, X. 2024c. OCRBench: on the hidden mystery of OCR in large multimodal models. *Science China Information Sciences*, 67(12).

Ong, I.; Almahairi, A.; Wu, V.; Chiang, W.-L.; Wu, T.; Gonzalez, J. E.; Kadous, M. W.; and Stoica, I. 2024. Routellm: Learning to route llms with preference data. *arXiv preprint arXiv:2406.18665*.

Rajput, K. S.; Anvekar, T.; Baral, C.; and Gupta, V. 2025. Rethinking Information Synthesis in Multimodal Question Answering A Multi-Agent Perspective. *arXiv preprint arXiv:2505.20816*.

Teerapittayanon, S.; McDanel, B.; and Kung, H.-T. 2017. Distributed deep neural networks over the cloud, the edge and end devices. In *2017 IEEE 37th international conference on distributed computing systems (ICDCS)*, 328–339. IEEE.

Wang, J.; Wang, J.; Athiwaratkun, B.; Zhang, C.; and Zou, J. 2024a. Mixture-of-Agents Enhances Large Language Model Capabilities. *arXiv preprint arXiv:2406.04692*.

Wang, J.; Zhang, H.; and Yuan, Y. 2025. Adv-cpg: A customized portrait generation framework with facial adversarial attacks. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, 21001–21010.

Wang, J.; Zhu, S.; Saad-Falcon, J.; Athiwaratkun, B.; Wu, Q.; Wang, J.; Song, S. L.; Zhang, C.; Dhingra, B.; and Zou, J. 2025. Think Deep, Think Fast: Investigating Efficiency of Verifier-free Inference-time-scaling Methods. *arXiv preprint arXiv:2504.14047*.

Wang, X.; Wei, J.; Schuurmans, D.; Le, Q.; Chi, E.; Narang, S.; Chowdhery, A.; and Zhou, D. 2022. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

Wang, Y.; Yang, C.; Lan, S.; Zhu, L.; and Zhang, Y. 2024b. End-edge-cloud collaborative computing for deep learning: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, 26(4): 2647–2683.

Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Xia, F.; Chi, E.; Le, Q. V.; Zhou, D.; et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35: 24824–24837.

Xu, Z.; Zhu, Y.; Deng, S.; Mittal, A.; Chen, Y.; Wang, M.; Favaro, P.; Tighe, J.; and Modolo, D. 2024. Benchmarking zero-shot recognition with vision-language models: Challenges on granularity and specificity. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 1827–1836.

Yang, A.; Li, A.; Yang, B.; Zhang, B.; Hui, B.; Zheng, B.; Yu, B.; Gao, C.; Huang, C.; Lv, C.; et al. 2025. Qwen3 technical report.

Yue, M.; Zhao, J.; Zhang, M.; Du, L.; and Yao, Z. 2024a. Large Language Model Cascades with Mixture of Thought Representations for Cost-Efficient Reasoning. In *The Twelfth International Conference on Learning Representations*.

Yue, X.; Ni, Y.; Zhang, K.; Zheng, T.; Liu, R.; Zhang, G.; Stevens, S.; Jiang, D.; Ren, W.; Sun, Y.; et al. 2024b. Mmmu: A massive multi-discipline multimodal understanding and reasoning benchmark for expert agi. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 9556–9567.

Zhang, H.; Huang, S.; Guo, Y.; and Li, X. 2025. Variational positive-incentive noise: How noise benefits models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Zhang, H.; Xu, Y.; Huang, S.; and Li, X. 2024a. Data augmentation of contrastive learning is estimating positive-incentive noise. *arXiv preprint arXiv:2408.09929*.

Zhang, X.; Huang, Z.; Taga, E. O.; Joe-Wong, C.; Oymak, S.; and Chen, J. 2024b. Efficient contextual llm cascades through budget-constrained policy learning. *Advances in Neural Information Processing Systems*, 37: 91691–91722.

Zhao, Y.; Lin, Z.; Zhou, D.; Huang, Z.; Feng, J.; and Kang, B. 2023. BuboGPT: Enabling Visual Grounding in Multi-Modal LLMs. arXiv:2307.08581.

Zhao, Z.; Jin, S.; and Mao, Z. M. 2024. Eagle: Efficient training-free router for multi-llm inference. *arXiv preprint arXiv:2409.15518*.

Zheng, L.; Chiang, W.-L.; Sheng, Y.; Zhuang, S.; Wu, Z.; Zhuang, Y.; Lin, Z.; Li, Z.; Li, D.; Xing, E. P.; Zhang, H.; Gonzalez, J. E.; and Stoica, I. 2023. Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena. arXiv:2306.05685.