MoCo: A One-Stop Shop for Model Collaboration Research

Anonymous authors
Paper under double-blind review

Abstract

Advancing beyond single monolithic language models (LMs), recent research increasingly recognizes the importance of *model collaboration*, where multiple LMs collaborate, compose, and complement each other. Existing research on this topic has mostly been disparate and disconnected, from different research communities, and lacks rigorous comparison. To consolidate existing research and establish model collaboration as a school of thought, we present MoCo: a one-stop Python library of executing, benchmarking, and comparing model collaboration algorithms at scale. MoCo features 26 model collaboration methods, spanning diverse levels of cross-model information exchange such as routing, text, logit, and model parameters. MoCo integrates 25 evaluation datasets spanning reasoning, QA, code, safety, and more, while users could flexibly bring their own data. Extensive experiments with MoCo demonstrate that most collaboration strategies outperform models without collaboration in 61.0% of (model, data) settings on average, with the most effective methods outperforming by up to 25.8%. We further analyze the scaling of model collaboration strategies, the training/inference efficiency of diverse methods, highlight that the collaborative system solves problems where single LMs struggle, and discuss future work in model collaboration, all made possible by MoCo. We envision MoCo as a valuable toolkit to facilitate and turbocharge the quest for an open, modular, decentralized, and collaborative AI future.[1]

1 Introduction

Language models (LMs) are increasingly not used in isolation but in *collaboration*: multiple LMs discuss (Feng et al., 2024b), debate (Du et al., 2023), divide and conquer (Yu et al., 2025) to solve complex problems; multiple LMs form routing systems to select the best model for each user query (Ong et al., 2025; Feng et al., 2025e); multiple LMs interact and exchange information in the logits (Liu et al., 2024a) and model parameter (Yadav et al., 2024; Wang et al., 2025a) space for collaborative decoding, generation, and deriving new models.

These efforts, despite being previously disconnected, unrelated, and underappreciated only as ad-hoc solutions to a narrow problem, together demonstrate the promise of *model collaboration* (Feng et al., 2025a): multiple language models, trained by different people, on different data, and thus possessing diverse skills and strengths, collaborate, compose, and complement each other. Model collaboration has the unique potential to unlock a collaborative and decentralized AI future, featuring modular and compositional AI systems built from the bottom up with everyone everywhere's models and contribution.

However, existing research on this topic are mostly disconnected and lack rigorous comparison. To consolidate existing research progress, evaluate diverse methods, motivate future work, and facilitate model collaboration's potential as compositional AI systems built by the many, we propose MoCo: a one-stop Python library and framework to build, execute, and compare diverse model collaboration systems:

- MoCo features a wide range of 26 model collaboration algorithms, spanning four levels of collaboration defined by the level of information exchange: API-level (e.g., routing (Ong et al., 2025) and switching (Feng et al., 2025d; Huang et al., 2026)), text-level (e.g., debate (Du et al., 2023) and cooperate (Yu et al.,

[1]To facilitate reviewing, an anonymous copy of MoCo is available at https://anonymous.4open.science/r/moco_anon-204B/.

2025)), logit-level (e.g., collective decoding (Liu et al., 2024a)), and weight-level (e.g., merging (Yadav et al., 2024) and parameter-space search (Feng et al., 2025c)).

- MoCo provides flexible implementations of model collaboration strategies, supporting their execution and evaluation with any amount of any LMs with any hardware setting (e.g., any amount of GPUs), democratizing model collaboration research for small models and less compute.
- MoCo comes with 25 built-in evaluation datasets (and growing), spanning reasoning, math, QA, knowledge, science, instruction following, safety, coding, computational social science, and more. Users could also flexibly bring their own prompts and datasets for evaluation and comparing model collaboration algorithms.
- Like model collaboration, we envision MoCo as a collaborative initiative: we provide detailed documentation and annotated code templates so everyone everywhere could contribute their model collaboration research to MoCo. We commit to providing continuous support for external contributors, even after publication.

Extensive experiments with MoCo demonstrate that model collaboration is a promising path towards modular and compositional AI systems. Model collaboration outperforms individual models in 61.0% of cases across diverse (model, data) settings, with the most successful algorithms outperforming in almost every evaluation domain by up to 25.8%. These results also enable reflection on existing methods and progress: text-level and weight-level collaboration generally work best across the board, reasoning tasks could be sensitive to model choice, and model collaboration algorithms benefit most from the diversity of language models. We further analyze the potential of scaling up these model collaboration systems, the training/inference efficiency of diverse methods, quantitative evidence where collaboration solves problems where individual models struggle, and motivate future work. We envision MoCo as a valuable framework to spearhead a new generation of modular, bottom-up, and compositional AI systems.

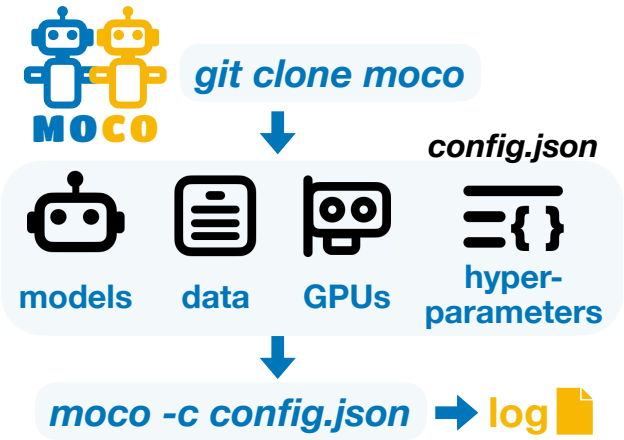

Figure 1: MoCo is a comprehensive library for model collaboration research. Download MoCo, write a config file specifying model collaboration setups (models, data, hardware, etc.), execute and compare diverse model collaboration algorithms with MoCo.

2 MoCo

We introduce MoCo, a comprehensive Python library for model collaboration research. MoCo supports 26 diverse approaches spanning four levels of collaboration, incorporates 25 datasets and benchmarks for evaluation, and features great flexibility and extensibility.

2.1 Methods

We categorize the 26 model collaboration algorithms in MoCo into four collaboration levels, depending on the level of information exchange across LLMs.

API-level collaboration approaches aim to select the best LLM in the pool for response generation, through paradigms such as routing, cascading, and switching.

Method #1: Prompt Routing Given an instruction, we prompt an LLM to select the best-fitting model in the pool based on their model descriptions provided by the user.

Method #2: Nudging A base LLM generates responses guided by one or more nudging LLMs: if the base LLM is uncertain about the next token (token probability lower than a threshold), then the nudging models

generate these tokens (often stylistic/discourse markers) to guide the base model's generation. (Fei et al., 2024)

Method #3: Switch Generation Given a pool of LLMs, we train a selector LM to govern how multiple LMs take turns to generate token patches as part of the full response. (Feng et al., 2025d)

Method #4: Trained Router Given a pool of LLMs, we train a routing LM by evaluating models on the dev set, identifying the best model for each data point, and supervised fine-tuning the routing LM to select the best model. At inference-time, the trained router conducts routing and the selected models generate. (Ong et al., 2025)

Method #5: Graph Router Similar to Method #4, but we employ a graph neural network operating on a task-query-model graph as the routing mechanism. (Feng et al., 2025e)

Method #6: Cascade Given an ordered list of language models, we let each model generate first and defer to the next model in line if the current model is uncertain, in terms of token probabilities. (Chen et al., 2023a; Gupta et al., 2024)

Method #7: Co-LLM We train a small deferral model for a pair of LLMs to decide when a model should defer generation to another model. During inference, the deferral model and two LLMs are jointly employed to collaborative generate responses. (Shen et al., 2024)

Method #8: Mentor Collab Given a generator model and a mentor model, we stop the generator model at random token positions, inspect whether the next predicted token differs between the two models. If yes, the generator model or an additionally trained classifier decides which model to generate the immediate following text patch.

Text-level collaboration approaches feature exchanges of generated texts among models, through paradigms such as "debate", "feedback", and "discuss".

Method #9: Multiagent Debate Given a pool of LLMs, each model first independently generates an answer, then refine their answer based on the answers of other LLMs. Repeats for a few iterations and an LLM summarizes the final responses. (Du et al., 2023)

Method #10: Multiagent Feedback Each model first independently generates an answer, then generates feedback for the answers of other models, and refines their own answer based on received feedback. Repeats for a few iterations and an LLM summarizes the final responses. (Feng et al., 2024b)

Method #11: LLM Blender Each model first generates an answer, then a ranker LLM, optionally trained with pairwise preferences on the dev set, conducts re-ranking of the responses. The top-k ranked responses are merged with a fuser LLM, optionally trained with gold answers on the dev set, to derive a final answer. (Jiang et al., 2023)

Method #12: Knowledge Card Given a question, each LLM generates a paragraph of related knowledge and information. An LLM then answers the question based on the aggregated knowledge of all LLMs. (Feng et al., 2024a)

Method #13: Majority Vote Each model generates an answer, then a majority/plurality vote is employed to select the final answer. Only works for objective tasks with a definitive answer.

Method #14: Heterogeneous Swarms Multiple LLMs form a directed acyclic graph to collaboratively generate responses, where one LLM's output becomes part of another LLM input based on directed edges. The graph structure is optimized with particle swarm optimization on the dev set. (Feng et al., 2025b)

Method #15: Multiagent Finetuning Each LLM first generates an initial response, then perform majority vote to get a consensus, and build generation and critic agent fine-tuning datasets for adaptation. Repeat for a few iterations for training. At inferece-time, the finetuned agents perform debate and the final answer is decided by majority vote. (Subramaniam et al., 2025)

Method #16: Structured Interaction Multiple LLMs interact and update their responses based on a specified graph structure, where each model receives the responses from models within its 1-hop neighborhood to update their answer. (Yu et al., 2025)

Method #17: BBMAS Multiple LLMs collaborate through a shared blackboard to iteratively solve problems. Specifically, models/agents take turns to contribute to the blackboard through a set of five actions. In the end, all models vote on the best final conclusion.

Method #18: Sparta Alignment Multiple LLMs collectievly self-align through competition and mutual evaluation. Specifically, two models are sampled to compete in fulfilling an instruction, while other LLMs judge the contest. The winning model gains in reputation and vice versa, which effects how much say does it have in evaluating other LLMs. We then perform preference optimization on the collected preferences, where the winning response is preferred over the losing response. (Jiang et al., 2025c)

Method #19: AggLM Each model first generates an response, we then train an aggregator model with reinforcement learning to aggregate their responses. Specifically, we employ GRPO (Shao et al., 2024) with verifiable rewards and balances training on "hard" examples (where majority voting fails) and "easy" examples (where majority voting succeeds) to learn both minority-answer recovery and reliable aggregation. (Zhao et al., 2025)

Logit-level collaboration approaches feature employing and transforming the token probability distributions among multiple LLMs for collaboration.

Method #20: Logit Fusion We average the next-token probabilities across multiple LLMs and decode from the joint distribution for text generation. These LLMs need to share the same tokenization/vocabulary.

Method #21: Logit Contrastive We first evaluate the pool of LLMs on the dev set of the dataset. We then retain the top-k and bottom-k models, and decode with joint distribution $p_1 + \alpha(p_1 + \cdots + p_k - p_{n-k+1} - \cdots - p_n)$, where p_i denotes the token probabilities of the i-th ranked model and α is a hyperparameter. (Liu et al., 2024a)

Weight-level collaboration approaches feature arithmetic and merging in the model parameter space for collaboration. These methods often require the LLM pool to share the same architecture.

Method #22: Greedy Soup We first evaluate the pool of LLMs on the dev set and sort them in descending performance. Starting from the best model, we iteratively add one model at a time to the soup, the parameter averaging of all selected models, and retains it if it improves performance on the dev set. (Wortsman et al., 2022)

Method #23: Dare Ties Models in the pool are merged with the DARE random pruning (Yu et al., 2024) and the TIES sign consensus of parameter values (Yadav et al., 2023). We incorporate the MergeKit implementation (Goddard et al., 2024) and gratefully acknowledge their valuable contribution.

Method #24: Model Swarms Multiple LLMs collaboratively search in the model weight space to find better parameter values based on dev set performance. The search is instanstiated with particle swarm optimization. (Feng et al., 2025c)

Method #25: LoraHub We use gradient-free optimization to learn the best scalar weights to linearly compose multiple language models, specifically LoRA adapters. (Huang et al., 2024)

Method #26: ExPO We employ model **ex**tra**po**lation in the parameter space over the pool of LLMs. Specifically, we first evaluate all LLMs on the dev set and rank by performance. We then merge the top-k models and bottom-k models, and extrapolate with $\mathbf{x}_{expo} = \mathbf{x}_{top\text{-}k} + \alpha(\mathbf{x}_{top\text{-}k} - \mathbf{x}_{bottom\text{-}k})$. (Zheng et al., 2024)

While MoCo provides a comprehensive slate of diverse model collaboration algorithms, it is not an exhaustive list: we encourage readers who work on model collaboration to get in touch and incorporate their method in MoCo.

2.2 Evaluation

To facilitate evaluations and fair comparisons, MoCo comes with 25 (and growing) evaluation datasets/benchmarks built-in, spanning diverse model capabilities.

- General-purpose QA: AGIEval (Zhong et al., 2024), ARC-challenge (Clark et al., 2018), MMLU-redux (Gema et al., 2025), GPQA (Rein et al., 2024)
- Math: GSM8k (Cobbe et al., 2021), MATH (Hendrycks et al., 2021a)
- Reasoning: BigBench-hard (Suzgun et al., 2023), TableMWP (Lu et al., 2023), TheoremQA (Chen et al., 2023b)
- Knowledge and factuality: WikiDYK (Zhang et al., 2025b), PopQA (Mallen et al., 2023)
- Human diversity: BLEND (Myung et al., 2024), CultureBench (Chiu et al., 2024)
- Science: Sciencemeter (Wang et al., 2025d), Sciriff (Wadden et al., 2025)
- Safety: TruthfulQA (Lin et al., 2022), CocoNot (Brahman et al., 2024)
- Coding: mbpp (Austin et al., 2021), HumanEval (Chen et al., 2021)
- Medical: MedQA (Jin et al., 2021; Li et al., 2024), MedMCQA (Pal et al., 2022), PubMedQA (Jin et al., 2019)
- Instruction following: AlpacaEval (Dubois et al., 2023), Wildchat (Zhao et al., 2024), human interest (Feng et al., 2025c)

We by default downsample to 1k for both the dev and test sets, if the original dataset is large. These tasks and datasets provide a comprehensive test bed for diverse model collaboration algorithms in MoCo, while users could flexibly bring their own data/benchmarks for evaluation with MoCo.

2.3 Design principles of MoCo

- **Flexibility**: We provide flexible implementations for diverse model collaboration strategies, so that they could be executed and evaluated with any amount of any LLMs with any hardware setting (e.g., any amount of common GPUs). We employ native Huggingface infrastructure for model training and inference, maximizing compatibility with diverse models. The commitment to democratize model collaboration research is core to our mission in MoCo, where small models and less compute are adequately supported.
- **Extensibility**: MoCo will not stop at 26 methods and 25 datasets. New methods from new/existing research could be flexibly added to MoCo: we provide blank code templates to guide contributors, and we commit to providing continuous support for external contributors, even after publication. Users could also flexibly bring their own data for generation and evaluation, or contributing evaluations important to them as part of MoCo.
- **Foundational**: In addition to being an artifact building on top of a lot of amazing research, we envision that MoCo also serves as a solid foundation and opens up sweeping new research avenues about model collaboration, compositional AI, collaborative development, and more. How do we incorporate the models/contributions across diverse parties to jointly build an AI system? What is the scalability of diverse model collaboration approaches in terms of the number of models and model diversity? How would malicious models impact the performance/integrity of different collaboration systems? How do we increase the efficiency and reconcile the strengths/weaknesses of diverse collaboration algorithms? These critical research questions are all made possible to pursue based on the unified MoCo framework and infrastructure.

Table 1: Performance of API-level, text-level, logit-level, and weight-level model collaboration algorithms over two model pool settings and six evaluation domains. Improvements over the best single model without collaboration are highlighted in orange. IF stands for instruction following and is evaluated with a reward model, where scores could be negative. When calculating the average column, we normalize IF scores with min-max standardization to 0-1 and calculate the macro-average across evaluation domains. Best in **bold** and second-best in underline. * indicates that these methods operate with objective tasks: the open-ended generation CocoNot dataset is not included in their performance on the safety domain. / indicates that the collaboration method is not compatible with this model/data setting (e.g., model merging requires models to share the same architecture and only works with model pool 1, open-ended tasks do not have a definitive answer for majority vote). Results show that diverse model collaboration approaches improve over individual models in 61.0% (model, evaluation) settings, with text-level and weight-level collaboration algorithms as generally stronger.

	Model Pool 1: Specialized LMs							Model Pool 2: General-Purpose LMs						
	QA	math	reason	safety	code	IF	avg.	QA	math	reason	safety	code	IF	avg.
Best Single	0.588	0.774	0.418	0.586	0.553	1.387	0.549	0.504	0.834	0.413	0.495	0.614	8.402	0.582
Cascade	0.597	0.805	0.387	0.631	0.597	1.385	0.565	0.540	0.831	0.423	0.457	0.632	9.289	0.591
Graph Routing	0.610	0.780	0.387	0.638	0.579	1.676	0.565	0.592	0.836	0.487	0.567	0.790	7.516	0.644
Prompt Routing	0.596	0.803	0.411	0.571	0.597	1.381	0.558	0.579	0.837	0.461	0.557	0.754	10.276	0.649
Switch Generation	0.587	0.733	0.382	0.652	0.412	-1.060	0.493	0.491	0.860	0.481	0.521	0.395	**17.234**	0.625
Trained Router	0.589	0.802	0.424	0.547	0.588	1.268	0.552	0.490	0.815	0.320	0.474	0.544	7.552	0.539
Mentor Collab	0.524	0.599	0.238	0.526	0.009	-3.552	0.316	0.396	0.748	0.416	0.543	0.412	-6.609	0.419
Nudging	0.595	0.789	0.395	0.583	0.596	0.994	0.550	0.545	0.814	0.481	0.508	0.763	8.640	0.625
Heterogeneous Swarms	0.633	0.801	0.448	0.587	0.614	0.331	0.563	0.610	0.884	0.528	0.537	0.728	9.764	0.662
Knowledge Card	0.494	0.761	0.438	0.488	0.386	-0.157	0.471	0.492	0.814	0.454	0.438	0.640	2.100	0.534
LLM Blender	**0.657**	**0.859**	0.450	0.558	0.605	-1.267	0.550	0.635	0.867	**0.530**	0.478	0.772	14.375	**0.694**
Majority Vote	0.622	0.608	0.382	0.601*	/	/	0.553	0.528	0.658	0.411	0.637*	/	/	0.558
Multiagent Feedback	0.496	0.696	0.423	0.419	0.439	-3.298	0.415	0.470	0.725	0.386	0.429	0.658	-2.284	0.475
Multiagent Finetuning	0.653	0.739	0.431	0.703*	/	/	0.631	0.567	0.882	0.503	**0.679***	/	/	0.658
Multiagent Refine	0.553	0.816	0.424	0.448	0.553	-2.946	0.473	0.516	0.799	0.480	0.471	0.649	1.224	0.541
Structure	0.571	0.763	0.372	0.590	0.526	-5.289	0.448	0.602	0.841	0.496	0.452	0.711	-3.163	0.541
Agg-LM	0.648	0.743	0.376	0.624*	/	/	0.598	**0.692**	**0.892**	0.524	0.635*	/	/	0.686
Sparta	0.562	**0.859**	0.478	0.551	**0.789**	**9.664**	0.707	0.590	0.843	0.469	0.556	**0.798**	9.883	0.658
Logit Fusion	0.499	0.587	0.350	0.563	0.482	-0.621	0.450	/	/	/	/	/	/	/
Logit Contrastive	0.557	0.442	0.291	0.602	0.114	-0.369	0.374	/	/	/	/	/	/	/
Dare Ties	0.625	0.814	0.438	0.608	0.675	1.848	0.595	/	/	/	/	/	/	/
Greedy Soup	0.621	0.795	0.420	0.701	0.675	1.822	0.603	/	/	/	/	/	/	/
LoraHub	0.454	0.844	0.409	0.575	0.395	-0.691	0.482	/	/	/	/	/	/	/
Model Swarms	0.628	0.853	**0.497**	**0.724**	0.763	8.493	**0.729**	/	/	/	/	/	/	/
Weight Expo	0.604	0.775	0.391	0.708	0.728	1.180	0.594	/	/	/	/	/	/	/

3 Experiment Settings

Models and Implementation We employ the two most representative model pool settings, specialized and general-purpose, to fairly benchmark diverse model collaboration approaches. Model pool #1 features 3 specialized LLMs (Jiang et al., 2025c) fine-tuned on different domains of data in Tulu-v3 (Lambert et al., 2024); model pool #2 features 3 general-purpose LLMs, specifically Qwen-2.5-7B (Yang et al., 2024), Llama-3.1-8B (Grattafiori et al., 2024), and Olmo-3-7B (Olmo et al., 2025). We focus on open models for reproducible research, while MoCo users could freely explore the collaboration of closed models. Note that some model collaboration algorithms might be designed for other specific settings (e.g., switch generation for pretrained-aligned collaboration (Feng et al., 2025d)): we encourage users to experiment with diverse model settings and choose collaboration strategies based on their need. We by default employ 512 max new tokens, 1024 for code generation, temperature $\tau = 0.7$, and top-p $p = 0.9$ sampling for text generation. We employ the default hyperparameters provided in MoCo for diverse model collaboration approaches.

Data and Evaluation We evaluate collaboration strategies across six domains of 11 datasets: QA (AGIeval, MMLU-redux), math (MATH, GSM8k), reasoning (BigBench-hard, TheoremQA), safety (CocoNot, TruthfulQA), coding (HumanEval), and instruction following (Alpaca, Human Interest). We employ task accuracy, generative verifiers (Ma et al., 2025), and reward models (Liu et al., 2024b) included in MoCo to evaluate their corresponding tasks, and report the macro-average of tasks within a domain. Though all other

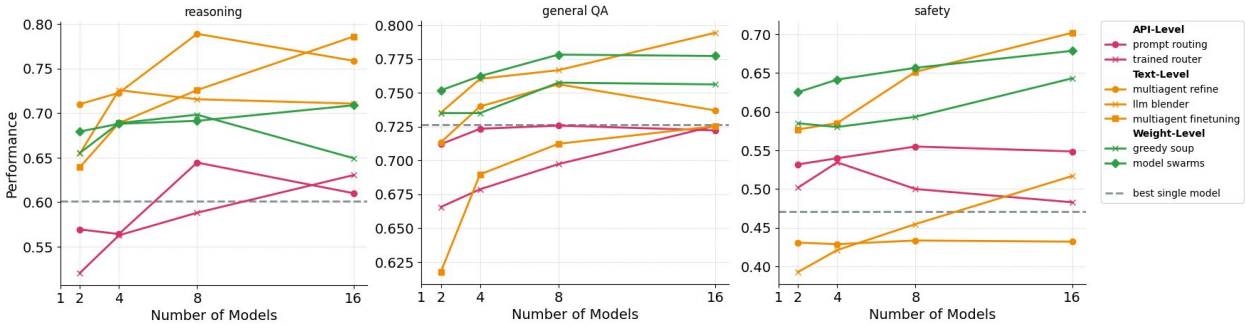

Figure 2: Scaling the number of models in model collaboration systems and evaluating on reasoning, QA, and safety domains. We observe a consistent upward trend that further improves over the best single model, with text-level and weight-level methods being more scalable and benefiting from a larger pool of diverse models. This indicates that by scaling up model collaboration, we could build bottom-up compositional AI systems where the components are small but the system is large.

datasets employ evaluation metrics bound in 0 to 1, instruction following evaluated by reward models is not: as a result, we use min-max normalization for instruction following before aggregating it with other domains.

Our main contribution is the MoCo tool and framework. In the following, we showcase some of the benchmarking, experiments, and analyses made possible by MoCo, while readers could use it for diverse settings not covered in this paper (diverse baselines, adding extra collaboration methods, additional model pools, datasets from other domains, etc.).

4 Results

Table 1 shows the performance of model collaboration approaches on two model pools and 6 evaluation domains.

Model collaboration is broadly effective. We highlight improvements over initial models without collaboration in orange cells: diverse model collaboration strategies bring performance gains across six evaluation domains, demonstrating their general effectiveness across varying models, settings, and tasks.

MoCo offers fair comparison and new insights about diverse collaboration strategies. Weight-level is in general the most effective, achieving an average performance of 60.1 compared to the global average of 53.5. These approaches operate with assumptions that the participating models share the same architecture, in order to perform merging or arithmetic directly with model parameters. Among all approaches, Model Swarms, Sparta Alignment, LLM Blender, and Agg-LM are among the best with three of them being text-level collaboration methods. This indicates that collaboration by models exchanging generated texts with each other is both broadly applicable and strong.

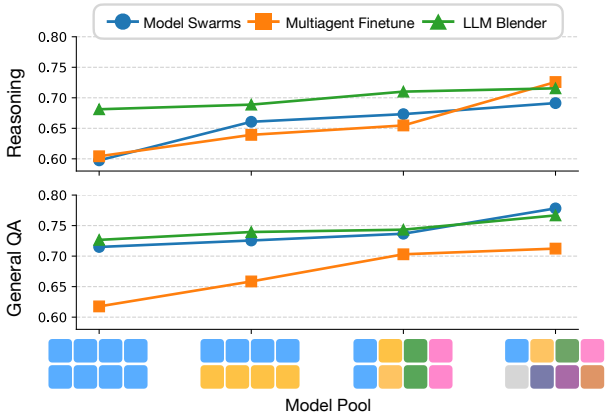

Figure 3: Impact of model diversity on collaboration performance. The x-axis shows configurations of model pool diversity: $1 \times 8, 2 \times 4, 4 \times 2$ and 8×1. Results demonstrate that model collaboration benefits from increased diversity among participating models, indicating the need for model specialization.

MoCo reveals the synergy among collaboration strategies, application domains, and model settings. Trained router works better with specialized LMs (pool #1) than general LMs (pool #2), potentially due to the artificial hivemind phenomenon (Jiang et al., 2025a) where general-purpose LMs generate similar responses to certain types of queries, reducing the effectiveness of routing. Multiagent

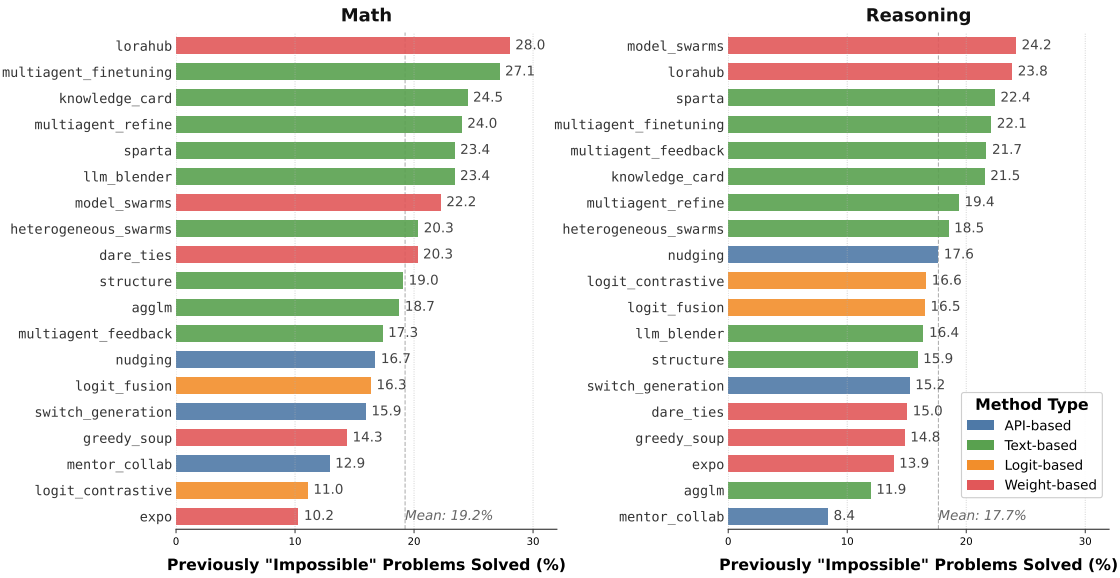

Figure 4: For problems where none of the LLMs could solve individually, what percentage of them are solvable with the model collaboration system, across diverse tasks and collaboration strategies. We observe consistent *collaborative emergence* across settings with an average of 18.5%, indicating that many model collaboration algorithms do not merely offer a union of existing capabilities: new skills emerge in the collaborative system of multiple models that solve problems where individual models struggle to.

Refine works great with math and reasoning tasks, but refining generation is challenging for safety and refusal scenarios in the CocoNot dataset. Approaches such as Sparta Alignment and Model Swarms bring consistent improvements to almost all model and dataset settings, thanks to their methodology of tailoring models and collaboration to diverse tasks and applications. By running, evaluating, and comparing diverse model collaboration algorithms with MoCo, we could derive a treasure trove of insights to reflect on existing methods and motivate future work.

5 Analysis

Scaling the number of models We experiment with scaling up model collaboration systems by scaling the number of participating models in collaboration algorithms. We experiment with 2, 4, 8, and 16 LLMs sourced from academic research artifacts (details in Appendix C) and evaluate with strong methods across collaboration levels. Figure 2 demonstrates that model collaboration algorithms are generally scalable: from 2 models to 16 models, there are consistent upward trends across reasoning, QA, and safety evaluation domains. Among them, text-level and weight-level methods are more successful than API-level routing approaches: scaling the number of models increases the candidate pool of routing and might introduce noise, while having more models collaborate via generated texts or model parameters offer deeper integration and stronger synergy.

Scaling the diversity of models The benefit of model collaboration not only comes from more compute, but also from the diversity and complementary strength of multiple LLMs. We experiment with $a \times b$ settings: a unique LLMs each repeated for b times, so that the total pool size is fixed as ab. We experiment with 1×8, 2×4, 4×2, and 8×1 settings and present results in Figure 3, showing a consistent upward trend with the increase of model diversity. This suggests that employing a pool of diverse language models is key to the success of model collaboration, and motivates future research on studying model diversity, training diverse language models, and more.

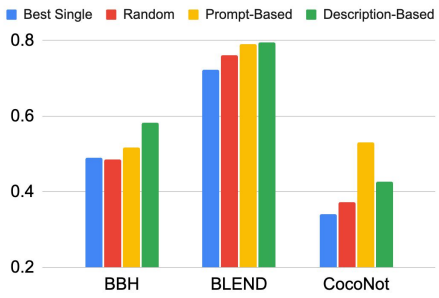

Figure 5: Employing random, prompt, or description-based strategies to select 3 models out of 8. Both strategies outperform the random baseline and no collaboration, indicating the importance of model selection strategies and highlighting the need for future research.

Collaborative emergence For a pool of multiple LLMs, the most complex and difficult problems might be beyond their capability, and none could individually solve them. However, we observe that model collaboration systems built with these LLMs could sometimes solve these problems "impossible" for the individual models, a phenomenon we term *collaborative emergence*. We quantify what percentage of these previously "impossible" problems are now solvable with the collaborative system with model pool #1 across diverse tasks and collaboration strategies. Figure 4 demonstrates that collaborative emergence is a consistent phenomenon across various settings, with an average of 18.5% problems now solvable with model collaboration algorithms. We present additional collaborative emergence results on more evaluation domains in Appendix B.

How to select models In section 4, we experiment with two most popular settings: a pool of specialized LLMs fine-tuned on different domains of data, and a pool of general-purpose LLMs released by different entities. However, for diverse applications, how to dynamically select models that offer a diverse set of related expertise for collaboration remains an open research question. MoCo empowers this investigation: we take an initial step with two model selection strategies. 1) *prompt-based selection*, where an LLM (Qwen-2.5-7B) is given the task description, the description of all candidate LLMs, and asked to select a subset of them for collaboration. 2) *similarity-based selection*, employing an encoder LM (RoBERTa-base) to encode model descriptions, calculate pairwise distances of description embeddings, and select a subset with the most intra-group distances. We experiment with the first 8 models in Figure 2 and select 3, comparing with baselines of the best single model and average performance over 5 random selections. Figure 5 shows that both strategies outperform random selection and no collaboration. We envision future research on dynamic model selection in model collaboration systems uniquely supported by MoCo.

6 Related Work

Section 2 provides a comprehensive overview of model collaboration methods and algorithms. In addition to individual methods, here we focus on high-level related works from both conceptual and engineering standpoints.

Du & Kaelbling (2024) argues that "a single is not all you need" and advocates for compositional generative systems, discussing their benefits across vision, reinforcement learning, robotics, and a brief mention of language. Feng et al. (2025a) later presents a deepdive on model collaboration specifically to language and language models, characterizing diverse collaboration strategies based on four levels of information exchange across models. Raffel (2023) proposes to "build machine learning models like open source software" and advocates for community-based training and updating of AI models: the community and decentralized aspects are important points underpinning the need for model collaboration. Many survey papers also summarize collaborative and modular approaches towards building AI models and systems (Yadav et al., 2024; Cai et al., 2025; Wang et al., 2025c).

From an engineering standpoint, MoCo is related to a few resources for modular and collaborative AI. MergeKit (Goddard et al., 2024) features implementations of diverse model merging approaches and is widely employed: for some of the weight-level collaboration algorithms in MoCo we employ MergeKit, and we gratefully acknowledge their contribution. LLMRouter (Feng et al., 2025f) and RouteLLM (Ong et al., 2025) are open-source libraries that support diverse methods for routing queries among multiple LLMs, overlapping with some of the API-level methods. In contrast, MoCo focuses on the whole spectrum of model collaboration algorithms, from API-level routing and switching, to text-level collaboration where models exchange generated texts, to logit-level arithematic on the token probabilities of multiple LMs, to weight-level merging and parameter-space methods. We encourage readers to check out the related valuable resources that inspired MoCo.

7 Conclusion

We present MoCo, a comprehensive toolkit and resource for model collaboration research. MoCo integrates 26 model collaboration algorithms spanning four levels of cross-LLM information exchange, 25 evaluation datasets spanning diverse application domains, and is seamlessly extensible for new novel methods and evaluation datasets. Extensive experiments with MoCo demonstrate that model collaboration approaches boost participating LLMs in 61.0% of cases, showing gains across a wide range of domains such as reasoning,

safety, coding, and more. Further analysis showcases the scalability of model collaboration, the important benefits of model diversity, and highlights collaborative emergence — how model collaboration systems solve challenging problems where the individual models can not. We envision MoCo as uniquely enabling and empowering novel and diverse research questions about model collaboration, compositional AI, and collaborative development.

Broader Impact Statement

As an open resource to facilitate model collaboration research, it is possible that malicious actors attempting to influence AI models/systems with a certain agenda would also investigate how a malicious model/component could impact/jailbreak compositional AI systems. As such, we envision important future work on the safety of model collaboration systems: studying the impact of malicious models in decentralized model collaboration systems, designing strategies to identify and mitigate their impact, and more. MoCo empowers this endeavor: by allowing the stress-testing and red-teaming of compositional AI systems and implementing diverse guardrail strategies, we will be ready to defend the integrity of future decentralized AI systems from the outset.

References

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

Faeze Brahman, Sachin Kumar, Vidhisha Balachandran, Pradeep Dasigi, Valentina Pyatkin, Abhilasha Ravichander, Sarah Wiegreffe, Nouha Dziri, Khyathi Chandu, Jack Hessel, et al. The art of saying no: Contextual noncompliance in language models. *Advances in Neural Information Processing Systems*, 37: 49706–49748, 2024.

Weilin Cai, Juyong Jiang, Fan Wang, Jing Tang, Sunghun Kim, and Jiayi Huang. A survey on mixture of experts in large language models. *IEEE Transactions on Knowledge and Data Engineering*, 2025.

Lingjiao Chen, Matei Zaharia, and James Zou. Frugalgpt: How to use large language models while reducing cost and improving performance. *Transactions on Machine Learning Research*, 2023a.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code. 2021.

Wenhu Chen, Ming Yin, Max Ku, Pan Lu, Yixin Wan, Xueguang Ma, Jianyu Xu, Xinyi Wang, and Tony Xia. Theoremqa: A theorem-driven question answering dataset. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 7889–7901, 2023b.

Yu Ying Chiu, Liwei Jiang, Bill Yuchen Lin, Chan Young Park, Shuyue Stella Li, Sahithya Ravi, Mehar Bhatia, Maria Antoniak, Yulia Tsvetkov, Vered Shwartz, et al. Culturalbench: A robust, diverse, and challenging cultural benchmark by human-ai culturalteaming. *arXiv preprint arXiv:2410.02677*, 2024.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Yilun Du and Leslie Pack Kaelbling. Position: Compositional generative modeling: A single model is not all you need. In *Forty-first International Conference on Machine Learning*, 2024.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B Tenenbaum, and Igor Mordatch. Improving factuality and reasoning in language models through multiagent debate. In *Forty-first International Conference on Machine Learning*, 2023.

Yann Dubois, Chen Xuechen Li, Rohan Taori, Tianyi Zhang, Ishaan Gulrajani, Jimmy Ba, Carlos Guestrin, Percy S Liang, and Tatsunori B Hashimoto. Alpacafarm: A simulation framework for methods that learn from human feedback. *Advances in Neural Information Processing Systems*, 36:30039–30069, 2023.

Yu Fei, Yasaman Razeghi, and Sameer Singh. Nudging: Inference-time alignment of llms via guided decoding. *arXiv preprint arXiv:2410.09300*, 2024.

Shangbin Feng, Weijia Shi, Yuyang Bai, Vidhisha Balachandran, Tianxing He, and Yulia Tsvetkov. Knowledge card: Filling llms' knowledge gaps with plug-in specialized language models. In *The Twelfth International Conference on Learning Representations*, 2024a.

Shangbin Feng, Weijia Shi, Yike Wang, Wenxuan Ding, Vidhisha Balachandran, and Yulia Tsvetkov. Don't hallucinate, abstain: Identifying llm knowledge gaps via multi-llm collaboration. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 14664–14690, 2024b.

Shangbin Feng, Wenxuan Ding, Alisa Liu, Zifeng Wang, Weijia Shi, Yike Wang, Zejiang Shen, Xiaochuang Han, Hunter Lang, Chen-Yu Lee, et al. When one llm drools, multi-llm collaboration rules. *arXiv preprint arXiv:2502.04506*, 2025a.

Shangbin Feng, Zifeng Wang, Palash Goyal, Yike Wang, Weijia Shi, Huang Xia, Hamid Palangi, Luke Zettlemoyer, Yulia Tsvetkov, Chen-Yu Lee, et al. Heterogeneous swarms: Jointly optimizing model roles and weights for multi-llm systems. *arXiv preprint arXiv:2502.04510*, 2025b.

Shangbin Feng, Zifeng Wang, Yike Wang, Sayna Ebrahimi, Hamid Palangi, Lesly Miculicich, Achin Kulshrestha, Nathalie Rauschmayr, Yejin Choi, Yulia Tsvetkov, et al. Model swarms: Collaborative search to adapt llm experts via swarm intelligence. In *Forty-second International Conference on Machine Learning*, 2025c.

Shangbin Feng, Wenhao Yu, Yike Wang, Hongming Zhang, Yulia Tsvetkov, and Dong Yu. Don't throw away your pretrained model. *arXiv preprint arXiv:2510.09913*, 2025d.

Tao Feng, Yanzhen Shen, and Jiaxuan You. Graphrouter: A graph-based router for llm selections. In *The Thirteenth International Conference on Learning Representations*, 2025e.

Tao Feng, Haozhen Zhang, Zijie Lei, Haodong Yue, Chongshan Lin, and Jiaxuan You. Llmrouter: An open-source library for llm routing. `https://github.com/ulab-uiuc/LLMRouter`, 2025f. GitHub repository.

Aryo Pradipta Gema, Joshua Ong Jun Leang, Giwon Hong, Alessio Devoto, Alberto Carlo Maria Mancino, Rohit Saxena, Xuanli He, Yu Zhao, Xiaotang Du, Mohammad Reza Ghasemi Madani, et al. Are we done with mmlu? In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 5069–5096, 2025.

Charles Goddard, Shamane Siriwardhana, Malikeh Ehghaghi, Luke Meyers, Vladimir Karpukhin, Brian Benedict, Mark McQuade, and Jacob Solawetz. Arcee's MergeKit: A toolkit for merging large language models. In Franck Dernoncourt, Daniel Preoţiuc-Pietro, and Anastasia Shimorina (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, November 2024.

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

Geyang Guo, Tarek Naous, Hiromi Wakaki, Yukiko Nishimura, Yuki Mitsufuji, Alan Ritter, and Wei Xu. Care: Aligning language models for regional cultural awareness. *arXiv preprint arXiv:2504.05154*, 2025.

Neha Gupta, Harikrishna Narasimhan, Wittawat Jitkrittum, Ankit Singh Rawat, Aditya Krishna Menon, and Sanjiv Kumar. Language model cascades: Token-level uncertainty and beyond. In *The Twelfth International Conference on Learning Representations*, 2024.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021a.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In *The Thirty-fifth Annual Conference on Neural Information Processing Systems*, 2021b.

Chengsong Huang, Qian Liu, Bill Yuchen Lin, Tianyu Pang, Chao Du, and Min Lin. Lorahub: Efficient cross-task generalization via dynamic lora composition. In *First Conference on Language Modeling*, 2024.

Chengsong Huang, Tong Zheng, Langlin Huang, Jinyuan Li, Haolin Liu, and Jiaxin Huang. Relayllm: Efficient reasoning via collaborative decoding. *arXiv preprint arXiv:2601.05167*, 2026.

Dongfu Jiang, Xiang Ren, and Bill Yuchen Lin. Llm-blender: Ensembling large language models with pairwise ranking and generative fusion. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 14165–14178, 2023.

Liwei Jiang, Yuanjun Chai, Margaret Li, Mickel Liu, Raymond Fok, Nouha Dziri, Yulia Tsvetkov, Maarten Sap, and Yejin Choi. Artificial hivemind: The open-ended homogeneity of language models (and beyond). In *The Thirty-ninth Annual Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2025a.

Pengcheng Jiang, Jiacheng Lin, Lang Cao, Runchu Tian, SeongKu Kang, Zifeng Wang, Jimeng Sun, and Jiawei Han. Deepretrieval: Hacking real search engines and retrievers with large language models via reinforcement learning. *arXiv preprint arXiv:2503.00223*, 2025b.

Yuru Jiang, Wenxuan Ding, Shangbin Feng, Greg Durrett, and Yulia Tsvetkov. Sparta alignment: Collectively aligning multiple language models through combat. *arXiv preprint arXiv:2506.04721*, 2025c.

Di Jin, Eileen Pan, Nassim Oufattole, Wei-Hung Weng, Hanyi Fang, and Peter Szolovits. What disease does this patient have? a large-scale open domain question answering dataset from medical exams. *Applied Sciences*, 11(14):6421, 2021.

Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William Cohen, and Xinghua Lu. Pubmedqa: A dataset for biomedical research question answering. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pp. 2567–2577, 2019.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, et al. Tulu 3: Pushing frontiers in open language model post-training. *arXiv preprint arXiv:2411.15124*, 2024.

Stella Li, Vidhisha Balachandran, Shangbin Feng, Jonathan Ilgen, Emma Pierson, Pang Wei W Koh, and Yulia Tsvetkov. Mediq: Question-asking llms and a benchmark for reliable interactive clinical reasoning. *Advances in Neural Information Processing Systems*, 37:28858–28888, 2024.

Zhuofeng Li, Haoxiang Zhang, Seungju Han, Sheng Liu, Jianwen Xie, Yu Zhang, Yejin Choi, James Zou, and Pan Lu. In-the-flow agentic system optimization for effective planning and tool use. In *NeurIPS 2025 Workshop on Efficient Reasoning*, 2025.

Stephanie Lin, Jacob Hilton, and Owain Evans. Truthfulqa: Measuring how models mimic human falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3214–3252, 2022.

Alisa Liu, Xiaochuang Han, Yizhong Wang, Yulia Tsvetkov, Yejin Choi, and Noah A Smith. Tuning language models by proxy. In *First Conference on Language Modeling*, 2024a.

Chris Yuhao Liu, Liang Zeng, Jiacai Liu, Rui Yan, Jujie He, Chaojie Wang, Shuicheng Yan, Yang Liu, and Yahui Zhou. Skywork-reward: Bag of tricks for reward modeling in llms. *arXiv preprint arXiv:2410.18451*, 2024b.

Gabrielle Kaili-May Liu, Bowen Shi, Avi Caciularu, Idan Szpektor, and Arman Cohan. Mdcure: A scalable pipeline for multi-document instruction-following. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 29258–29296, 2025.

Pan Lu, Liang Qiu, Kai-Wei Chang, Ying Nian Wu, Song-Chun Zhu, Tanmay Rajpurohit, Peter Clark, and Ashwin Kalyan. Dynamic prompt learning via policy gradient for semi-structured mathematical reasoning. In *The Eleventh International Conference on Learning Representations*, 2023.

Xueguang Ma, Qian Liu, Dongfu Jiang, Ge Zhang, Zejun Ma, and Wenhu Chen. General-reasoner: Advancing llm reasoning across all domains. *arXiv preprint arXiv:2505.14652*, 2025.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9802–9822, 2023.

Niklas Muennighoff, SU Hongjin, Liang Wang, Nan Yang, Furu Wei, Tao Yu, Amanpreet Singh, and Douwe Kiela. Generative representational instruction tuning. In *The Thirteenth International Conference on Learning Representations*, 2025.

Junho Myung, Nayeon Lee, Yi Zhou, Jiho Jin, Rifki Putri, Dimosthenis Antypas, Hsuvas Borkakoty, Eunsu Kim, Carla Perez-Almendros, Abinew Ali Ayele, et al. Blend: A benchmark for llms on everyday knowledge in diverse cultures and languages. *Advances in Neural Information Processing Systems*, 37:78104–78146, 2024.

Team Olmo, Allyson Ettinger, Amanda Bertsch, Bailey Kuehl, David Graham, David Heineman, Dirk Groeneveld, Faeze Brahman, Finbarr Timbers, Hamish Ivison, et al. Olmo 3. *arXiv preprint arXiv:2512.13961*, 2025.

Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E Gonzalez, M Waleed Kadous, and Ion Stoica. Routellm: Learning to route llms from preference data. In *The Thirteenth International Conference on Learning Representations*, 2025.

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

Chau Minh Pham, Yapei Chang, and Mohit Iyyer. Clipper: Compression enables long-context synthetic data generation. *arXiv preprint arXiv:2502.14854*, 2025.

Cheng Qian, Emre Can Acikgoz, Qi He, Hongru Wang, Xiusi Chen, Dilek Hakkani-Tür, Gokhan Tur, and Heng Ji. Toolrl: Reward is all tool learning needs. *arXiv preprint arXiv:2504.13958*, 2025.

Colin Raffel. Building machine learning models like open source software. *Communications of the ACM*, 66 (2):38–40, 2023.

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

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Zejiang Shen, Hunter Lang, Bailin Wang, Yoon Kim, and David Sontag. Learning to decode collaboratively with multiple language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12974–12990, 2024.

Linxin Song, Taiwei Shi, and Jieyu Zhao. The hallucination tax of reinforcement finetuning. *arXiv preprint arXiv:2505.13988*, 2025.

Vighnesh Subramaniam, Yilun Du, Joshua B Tenenbaum, Antonio Torralba, Shuang Li, and Igor Mordatch. Multiagent finetuning: Self improvement with diverse reasoning chains. In *The Thirteenth International Conference on Learning Representations*, 2025.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed Chi, Denny Zhou, et al. Challenging big-bench tasks and whether chain-of-thought can solve them. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 13003–13051, 2023.

Zhaoxuan Tan, Zheng Li, Tianyi Liu, Haodong Wang, Hyokun Yun, Ming Zeng, Pei Chen, Zhihan Zhang, Yifan Gao, Ruijie Wang, et al. Aligning large language models with implicit preferences from user-generated content. *arXiv preprint arXiv:2506.04463*, 2025.

Liyan Tang, Philippe Laban, and Greg Durrett. Minicheck: Efficient fact-checking of llms on grounding documents. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 8818–8847, 2024.

Vijay Viswanathan, Yanchao Sun, Xiang Kong, Meng Cao, Graham Neubig, and Tongshuang Wu. Checklists are better than reward models for aligning language models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

David Wadden, Kejian Shi, Jacob Morrison, Alan Li, Aakanksha Naik, Shruti Singh, Nitzan Barzilay, Kyle Lo, Tom Hope, Luca Soldaini, et al. Sciriff: A resource to enhance language model instruction-following over scientific literature. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 6083–6120, 2025.

Junlin Wang, WANG Jue, Ben Athiwaratkun, Ce Zhang, and James Zou. Mixture-of-agents enhances large language model capabilities. In *The Thirteenth International Conference on Learning Representations*, 2025a.

Liaoyaqi Wang, Zhengping Jiang, Anqi Liu, and Benjamin Van Durme. Always tell me the odds: Fine-grained conditional probability estimation. *arXiv preprint arXiv:2505.01595*, 2025b.

Xin Wang, Haoyang Li, Haibo Chen, Zeyang Zhang, and Wenwu Zhu. Modular machine learning: An indispensable path towards new-generation large language models. *arXiv preprint arXiv:2504.20020*, 2025c.

Yike Wang, Shangbin Feng, Yulia Tsvetkov, and Hannaneh Hajishirzi. Sciencemeter: Tracking scientific knowledge updates in language models. *arXiv preprint arXiv:2505.24302*, 2025d.

Kangda Wei, Hasnat Md Abdullah, and Ruihong Huang. Mitigating gender bias via fostering exploratory thinking in llms. *arXiv preprint arXiv:2505.17217*, 2025.

Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, et al. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International conference on machine learning*, pp. 23965–23998. PMLR, 2022.

Zhengxuan Wu, Qinan Yu, Aryaman Arora, Christopher D Manning, and Christopher Potts. Improved representation steering for language models. *arXiv preprint arXiv:2505.20809*, 2025.

Emily Xiao, Yixiao Zeng, Ada Chen, Chin-Jou Li, Amanda Bertsch, and Graham Neubig. Prompt-mii: Meta-learning instruction induction for llms. *arXiv preprint arXiv:2510.16932*, 2025.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin A Raffel, and Mohit Bansal. Ties-merging: Resolving interference when merging models. *Advances in Neural Information Processing Systems*, 36:7093–7115, 2023.

Prateek Yadav, Colin Raffel, Mohammed Muqeeth, Lucas Caccia, Haokun Liu, Tianlong Chen, Mohit Bansal, Leshem Choshen, and Alessandro Sordoni. A survey on model moerging: Recycling and routing among specialized experts for collaborative learning. *Transactions on Machine Learning Research*, 2024.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zhihao Fan. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2024.

Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. Language models are super mario: Absorbing abilities from homologous models as a free lunch. In *Forty-first International Conference on Machine Learning*, 2024.

Miao Yu, Shilong Wang, Guibin Zhang, Junyuan Mao, Chenlong Yin, Qijiong Liu, Kun Wang, Qingsong Wen, and Yang Wang. Netsafe: Exploring the topological safety of multi-agent system. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 2905–2938, 2025.

Muru Zhang, Mayank Mishra, Zhongzhu Zhou, William Brandon, Jue WANG, Yoon Kim, Jonathan Ragan-Kelley, Shuaiwen Leon Song, Ben Athiwaratkun, and Tri Dao. Ladder-residual: Parallelism-aware architecture for accelerating large model inference with communication overlapping. In *Forty-second International Conference on Machine Learning*, 2025a.

Yuwei Zhang, Wenhao Yu, Shangbin Feng, Yifan Zhu, Letian Peng, Jayanth Srinivasa, Gaowen Liu, and Jingbo Shang. Bidirectional lms are better knowledge memorizers? a benchmark for real-world knowledge injection. *arXiv preprint arXiv:2505.12306*, 2025b.

Wenting Zhao, Xiang Ren, Jack Hessel, Claire Cardie, Yejin Choi, and Yuntian Deng. Wildchat: 1m chatgpt interaction logs in the wild. In *The Twelfth International Conference on Learning Representations*, 2024.

Wenting Zhao, Pranjal Aggarwal, Swarnadeep Saha, Asli Celikyilmaz, Jason Weston, and Ilia Kulikov. The majority is not always right: Rl training for solution aggregation. *arXiv preprint arXiv:2509.06870*, 2025.

Chujie Zheng, Ziqi Wang, Heng Ji, Minlie Huang, and Nanyun Peng. Weak-to-strong extrapolation expedites alignment. In *ICML 2024 Workshop on Models of Human Feedback for AI Alignment*, 2024.

Wanjun Zhong, Ruixiang Cui, Yiduo Guo, Yaobo Liang, Shuai Lu, Yanlin Wang, Amin Saied, Weizhu Chen, and Nan Duan. Agieval: A human-centric benchmark for evaluating foundation models. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pp. 2299–2314, 2024.

A Discussion

MoCo offers flexible implementations of diverse model collaboration algorithms, uniquely enabling the study and investigation of a wide spectrum of research questions about model collaboration, compositional AI systems, collaborative development, and more.

- How do we scale up model collaboration? Which collaboration strategies are more scalable to large quantities of diverse LLMs, so that we could build compositional AI systems where the modular components are small but the system is large?
- How do we achieve bottom-up collaborative development for AI systems? Specifically, how do models trained by different stakeholders together form decentralized systems where no one has unilateral control over state-of-the-art AI?
- How does the cost/efficiency of diverse collaboration strategies compare?[3] How do we improve efficiency and design novel and cost-effective collaboration algorithms (e.g., through information exchange at the latent space (Wu et al., 2025))?
- What are the risks of having malicious models in model collaboration systems? How do we safeguard decentralized collaborative AI systems from malicious actors and artifacts?
- How do we train models that are not only individually strong, but also *compositionally strong*: models that bring new information, improve on underrepresented skills, and boost existing models when used in collaboration?

We envision the creation of MoCo as turbocharging future research on these and many other important topics for an open, compositional, and decentralized AI future.

B Additional Analysis

Training/Inference Complexity We analyze the training and inference time complexity of different collaboration methods in Table 4. Most collaboration methods require an additional training stage to determine the collaboration structure. Moreover, collaboration at different levels incurs varying computational costs, with text-level methods exhibiting relatively higher time complexity.

Collaborative Emergence on More Domains We extend our analysis of collaborative emergence to additional evaluation domains: General-purpose QA, Safety, and Coding. Figures 6, 7, and 8 present the percentage of previously "impossible" problems—those unsolvable by any individual LLM—that become solvable through model collaboration. For the General-purpose QA domain, we observe a mean collaborative emergence rate of 15.8%, with multiagent_finetuning (26.1%) and llm_blender (21.8%) achieving the highest rates. In the Safety domain, logit_contrastive achieves the highest emergence rate of 27.0%, followed by expo at 24.2%, resulting in a mean of 14.1%. The Coding domain exhibits a strong collaborative emergence with a mean of 17.6%, where sparta alignment achieves a remarkable 38.1% and model swarms follows at 28.6%. These results reinforce that collaborative emergence is a robust phenomenon across diverse tasks, and different collaboration strategies exhibit varying strengths depending on the application domain.

	mmlu-redux	gsm8k	bbh
w/o model 1	0.572	0.729	0.539
w/o model 2	0.581	0.759	0.561
w/o model 3	0.589	0.822	0.565
w/o model 4	0.593	0.828	0.588
w/o model 5	0.598	0.872	0.598
avg	0.587	0.802	0.570
std	0.010	0.057	0.023

Table 2: Leave-one-out analysis to study the sensitivity of model collaboration, specifically the multiagent debate approach, to minor changes in model composition. An average standard deviation of 0.030 shows that it is mostly robust.

[3] We present an analysis of existing algorithms of MoCo in Appendix B.

Dataset	Source	Size	
		dev	test
AGIeval	(Zhong et al., 2024)	1156	1156
Alpaca	(Dubois et al., 2023)	2000	1000
ARC-Challenge	(Clark et al., 2018)	299	1172
BigBench-hard	(Suzgun et al., 2023)	1000	1000
BLEND	(Myung et al., 2024)	1000	1000
CocoNot	(Brahman et al., 2024)	1000	1000
CultureBench	(Chiu et al., 2024)	2454	2454
GPQA (Diamond)	(Rein et al., 2024)	99	99
GPQA (Extended)	(Rein et al., 2024)	273	273
GPQA (Main)	(Rein et al., 2024)	224	224
GSM8k	(Cobbe et al., 2021)	200	1000
Human Interest	(Feng et al., 2025c)	400	400
HumanEval	(Chen et al., 2021)	50	114
MATH	(Hendrycks et al., 2021b)	956	956
MBPP	(Austin et al., 2021)	487	487
MedMCQA	(Pal et al., 2022)	2091	2092
MedQA	(Jin et al., 2021)	636	637
MMLU-redux	(Gema et al., 2025)	1000	1000
PopQA	(Mallen et al., 2023)	1000	1000
PubMedQA	(Jin et al., 2019)	500	500
ScienceMeter	(Wang et al., 2025d)	1000	1000
SciRIFF	(Wadden et al., 2025)	114	139
TableMWP	(Lu et al., 2023)	375	376
TableMWP (MC)	(Lu et al., 2023)	124	125
TheoremQA	(Chen et al., 2023b)	400	400
TruthfulQA	(Lin et al., 2022)	200	617
WikiDyK	(Zhang et al., 2025b)	1000	765
WildChat	(Zhao et al., 2024)	1000	1000

Table 3: Datasets Statistics.

Sensitivity to Model Choice Ideally, model collaboration strategies should be robust to minor changes in participating models. We create a testbed of model choice sensitivity by employing the first 5 models in Figure 2, and conduct leave-one-out analysis with the multiagent debate strategy across three datasets. Results in Table 2 demonstrate that it is mostly robust with an average standard deviation of 0.03 across five model pool settings, with math and reasoning being more sensitive and would thus benefit from tailored model selection strategies.

When Model Collaboration Fails There are cases where model collaboration didn't lead to performance improvements: we observe that there is no one-size-fits-all approach and each collaboration strategy has its pros and cons. For example, multiagent refine excels on reasoning problems, but the debate and refinement process might be counterproductive on safety tasks and could actually "refine" responses towards safe directions. API-level routing methods benefit from having a small and high-quality pool of models: when the number of models increases, they might be susceptible to noise. MoCo uniquely enables the understanding and investigation of when and how specific collaboration strategies fail, therefore informing the selection of collaboration algorithms for diverse use cases.

C Experiment Details

Dataset Details We systematically report datasets details integrated in MoCo, including reasoning, math, QA, knowledge, instruction following, safety, coding and social science. MoCo is easily extensible and users can easily incorporate additional datasets in MoCo. Datasets statistics are summarized in Table 3.

Implementation Details We by default employ 512 max new tokens, with an exception of 1024 for coding tasks, and $\tau = 0.7$ and $p = 0.9$ for temperature and top-p sampling in text generation. We employ the default hyperparameters in MoCo for different model collaboration algorithms. Model pool #1 includes the following three models from (Jiang et al., 2025c): bunsenfeng/yuru_qw_wizardlm, bun-

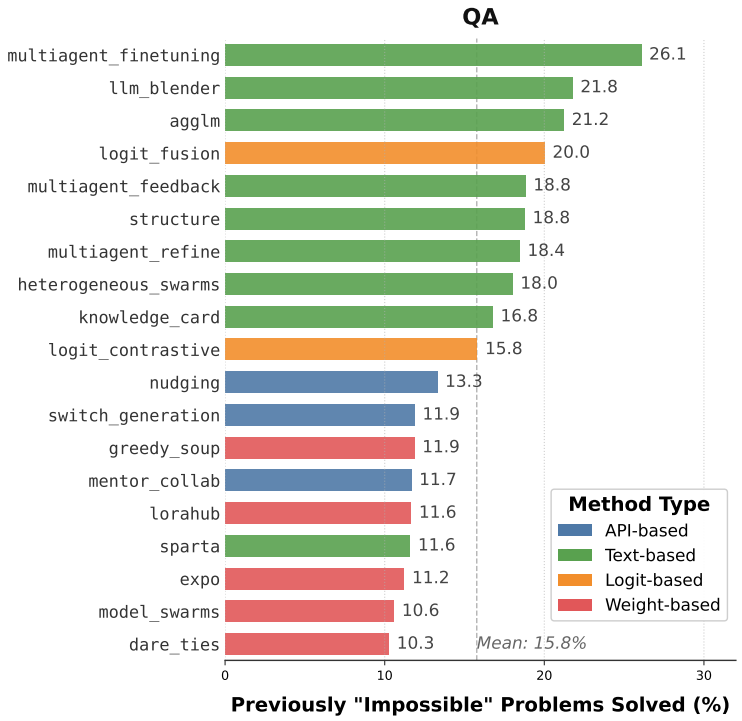

Figure 6: Collaborative emergence on the General-purpose QA domain.

senfeng/yuru_qw_sharegpt, bunsenfeng/yuru_qw_oasst1. Model pool #2 includes the following three models: Qwen/Qwen2.5-7B-Instruct, meta-llama/Llama-3.1-8B-Instruct, allenai/Olmo-3-7B-Instruct. We evaluate instruction following in Table 1 with the Skywork reward model (Skywork/Skywork-Reward-Llama-3.1-8B-v0.2). We evaluate coding tasks with pass@1.

For Figure 2, we employ 16 LLMs in the following order: chtmp223/Qwen2.5-7B-CLIPPER (Pham et al., 2025), chengq9/ToolRL-Qwen2.5-3B (Qian et al., 2025), AgentFlow/agentflow-planner-7b (Li et al., 2025), nanami/ladder-last16L-llama3.1-8binstruct-sft4k-stage2v03-bsize32-rkl8b (Zhang et al., 2025a), viswavi/qwen2.5_rlcf (Viswanathan et al., 2025), milli19/promptmii-llama-3.1-8b-instruct (Xiao et al., 2025), Zhengping/conditional-probability-regression (Wang et al., 2025b), yale-nlp/MDCure-Qwen2-7B-Instruct (Liu et al., 2025), GritLM/GritLM-7B (Muennighoff et al., 2025), lime-nlp/Qwen2.5-7B-Instruct-SUM10 (Song et al., 2025), geyang627/care-chinese-gemma2-9b (Guo et al., 2025), bespokelabs/Bespoke-Stratos-7B (Tang et al., 2024), kangdawei/Llama-3.1-8B-Instruct-GenderNeutral-Finetuned (Wei et al., 2025), DeepRetrieval/DeepRetrieval-PubMed-3B-Llama (Jiang et al., 2025b), yale-nlp/MDCure-Qwen2-1.5B-Instruct (Liu et al., 2025), Zhaoxuan/PUGC-Mistral-DPO (Tan et al., 2025). To run weight-level approaches with these models, we perform distillation with these models as teacher and Qwen-2.5-7B as student using Tulu-v3 data (Lambert et al., 2024) to standardize model architecture.

Releasing MoCo We will make MoCo publicly available upon acceptance: in the meantime, reviewers could check out the anonymous repository[4]. We will 1) release MoCo as a Github repository for directly executing from source, and 2) release a PyPI package based on MoCo for command-line execution. We commit to offer continuous support for MoCo even after publication: working with external contributors to add their collaboration algorithms, add new datasets, update PyPI package versions, and more.

[4]https://anonymous.4open.science/r/moco_anon-204B/

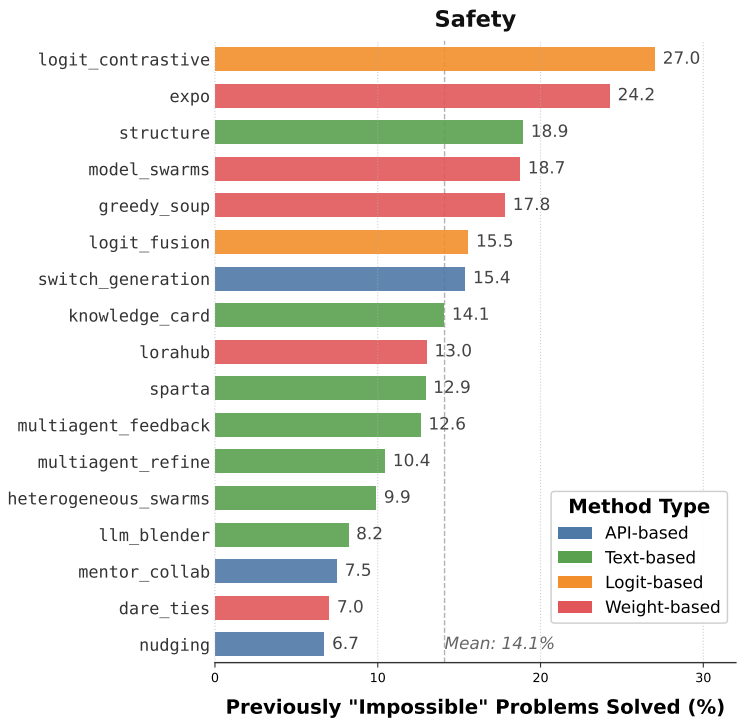

Figure 7: Collaborative emergence on the Safety domain.

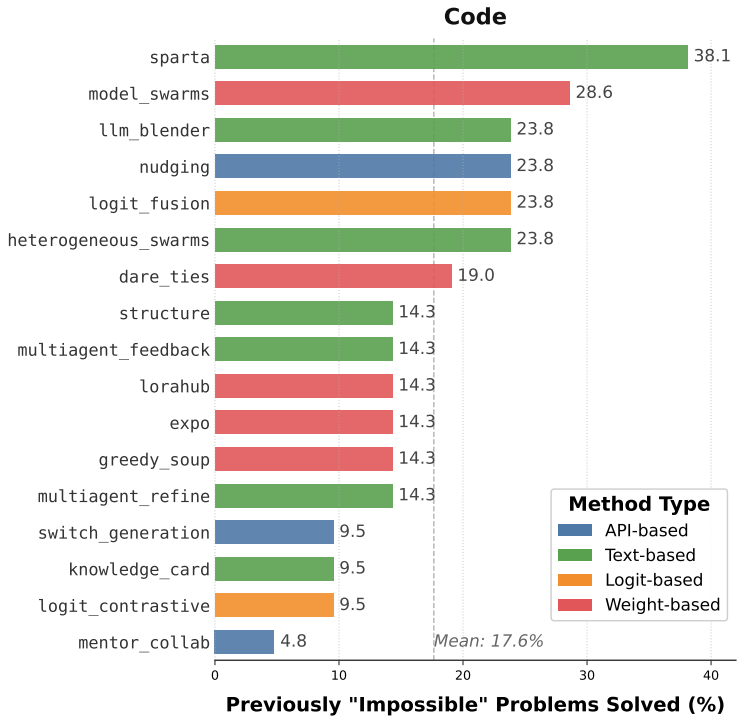

Figure 8: Collaborative emergence on the Coding domain.

Method	Training	Inference	Notes
Cascade	/	$2Dm \cdot \sum_{i=1}^{n} \frac{k_i}{2^{i-1}}$	Each level defer 50%
Graph Router	$2nDm \cdot \max(k_i)$	$2Dm \cdot \max(k_i)$	Omit GNN training
Prompt Router	/	$2Dm \cdot (k_r + \max(k_i))$	Router k_r
LLM Router	$2nDm \cdot \max(k_i) + 6rDM \cdot k_r$	$2Dm \cdot (k_r + \max(k_i))$	Router k_r
Switch Generate	$6rDm \cdot k_s$	$2Dm \cdot \left(k_s/\textit{PATCH} + \max(k_i)\right)$	Switcher k_s, rounds r, path size *patch*
Mentor Collab	$2nDm \cdot \max(k_i)$	$2Dm \cdot \text{mean}(k_i)$	Omit MLP training
Co-llm	$2nDm \cdot \max(k_i)$	$2Dm \cdot \text{mean}(k_i)$	/
Nudging	/	$2Dm \cdot \max(k_i)$	/
Heterogeneous Swarms	$2nDm \cdot \max(k_i)$	$2GrDm \cdot \max(k_i)$	Graph generation G, rounds r
Knowledge Card	$2nDm \cdot \max(k_i)$	$2nDm \cdot \max(k_i)$	/
LLM Blender	$2nDm \cdot \max(k_i) + 6rDM \cdot (n^2 k_r + k_f)$	$2nDM \cdot \max(k_i) + 2DM \cdot (k_r + k_f)$	ranker k_r, fuser k_f, rounds r
Majority Vote	/	$2nDm \cdot \max(k_i)$	/
Multiagent Refine	$2nDm \cdot \max(k_i)$	$2nrDm \cdot \max(k_i)$	rounds r
Multiagent Feedback	$2nDm \cdot \max(k_i)$	$2nrfDm \cdot \max(k_i)$	Feedback f, rounds r
Multiagent Finetuning	$2nDm \cdot \max(k_i) + 6nrDM \cdot 2\max(k_i)$	$2nrDm \cdot \max(k_i)$	rounds r
Structure	$2nDm \cdot \max(k_i)$	$2GrDm \cdot \max(k_i)$	structure G, rounds r
Agg-LM	$2nsDM \cdot \max(k_i) + 6rDM \cdot \min(k_i)$	$2Dm \cdot (n\max(k_i) + \min(k_i))$	sample size s, rounds r
Sparta	$2nDm \cdot \max(k_i) + 6nrDM \cdot \max(k_i)$	$2nDm \cdot \max(k_i)$	training rounds r
Logit Fusion	/	$2nDm \cdot \max(k_i)$	/
Logit Contrastive	$2nDm \cdot \max(k_i)$	$2nDm \cdot \max(k_i)$	/
Dare Ties	/	$2Dm \cdot \max(k_i)$	/
Greedy Soup	$2(2n-1)Dm \cdot \max(k_i)$	$2Dm \cdot \max(k_i)$	
LoraHub	$2nDm \cdot \max(k_i)$	$2Dm \cdot \max(k_i)$	/
Model Swarms	$2nrDm \cdot \max(k_i)$	$2Dm \cdot \max(k_i)$	training rounds r
Weight ExPO	$2nDm \cdot \max(k_i)$	$2Dm \cdot \max(k_i)$	/

Table 4: Collaboration Methods: Training and Inference FLOPs Complexity Analysis. D denotes the dataset size, m the maximum tokens length and the model pool $\mathcal{K} = \{k_i\}_1^n$ contains n models. Each forward pass produces 2 FLOPs per parameter, and each backward pass produces 6 FLOPs per parameter.