Symbolic Mixture-of-Experts: Adaptive Skill-based Routing for Heterogeneous Reasoning

Anonymous authors
Paper under double-blind review

Abstract

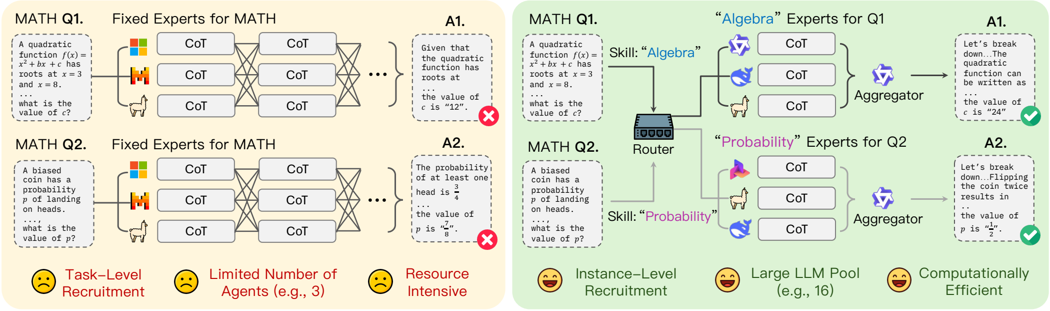
Combining existing pre-trained expert LLMs is a promising avenue for scalably tackling large-scale and diverse tasks. However, selecting experts at the task level is often too coarse-grained, as heterogeneous tasks may require different expertise for each instance. To enable adaptive instance-level mixing of pre-trained LLM experts, we propose Symbolic-MoE, a symbolic, text-based, and gradient-free Mixture-of-Experts framework. Symbolic-MoE takes a fine-grained approach to selection by emphasizing skills, i.e., specialized subcategories such as algebra in mathematics. We propose a skill-based recruiting strategy that dynamically selects the most relevant set of expert LLMs for diverse reasoning tasks based on their strengths. Each selected expert then generates its own reasoning, resulting in k outputs from k experts, which are then synthesized into a final high-quality response by an aggregator, chosen based on its ability to integrate diverse outputs. We show that instance-level expert selection improves performance by a large margin but – when implemented naively – can introduce a high computational overhead due to the need for constant model loading and offloading. To address this, we implement a batch inference strategy that groups instances based on their assigned experts, ensuring each model will only be loaded once. This allows us to integrate 16 models *on a single GPU* with a time cost comparable to prior multi-agent baselines using 4 GPUs. Through extensive evaluations on diverse benchmarks (MMLU-Pro, GPQA, AIME, and MedMCQA), we show that Symbolic-MoE outperforms prior multi-agent approaches, with an absolute average improvement of 8.15% over the best baseline. Moreover, Symbolic-MoE generalizes well to unseen tasks and removes the need for expensive multi-round discussions, outperforming discussion baselines with less computation. [1]

1 Introduction

A core strength of humans is our ability to communicate and coordinate with each other using language (Clark, 1996; Yow & Lim, 2019; Xu et al., 2023). This allows diverse experts to contribute specialized knowledge towards solving a problem. Like humans, large language models (LLMs) often have differing skills and strengths, derived from differences in their architectures and training regimens. For instance, math-specific models like MetaMath (Yu et al., 2023) or QwenMath (Yang et al., 2024) are post-trained with mathematical reasoning data, making them particularly adept at math tasks – often at the cost of performance on out-of-distribution tasks (Kumar et al., 2022; Chu et al., 2025) like commonsense or medical reasoning (Lobo et al., 2024). Even within specialized domains, differences in pre-training data can lead to nuanced variations in expertise: one math-focused model may excel at algebra, while another is better suited for geometry. This motivates our development of an automated, skill-based framework designed to identify and select the most suitable set of expert models *for each problem*.

Indeed, combining multiple "expert" models via Mixture-of-Experts (MoE) is well-studied (Jacobs et al., 1991; Eigen et al., 2013) and has been applied widely for large pre-trained models, enabling better performance at a lower computational cost (Shazeer et al., 2017a; Fedus et al., 2022; Riquelme

[1] Code is provided in the supplement materials.


MATH Q1.
Fixed Experts for MATH
A quadratic function $f(x) = x^2 + bx + c$ has roots at $x = 3$ and $x = 8$. ... what is the value of c?
CoT
A1.
Given that the quadratic function has roots at ... the value of c is "12".
MATH Q2.
Fixed Experts for MATH
A biased coin has a probability p of landing on heads. ..., what is the value of p?
A2.
The probability of at least one head is $\frac{3}{4}$ ... the value of p is "$\frac{7}{8}$".
Task–Level Recruitment
Limited Number of Agents (e.g., 3)
Resource Intensive
MATH Q1.
Skill: "Algebra"
"Algebra" Experts for Q1
Router
Aggregator
A1.
Let's break down…The quadratic function can be written as ... the value of c is "24"
MATH Q2.
Skill: "Probability"
"Probability" Experts for Q2
A2.
Let's break down…Flipping the coin twice results in .. the value of p is "$\frac{1}{2}$".
Instance–Level Recruitment
Large LLM Pool (e.g., 16)
Computationally Efficient

(a) Existing Multi–Agent Work

(b) Symbolic–MoE (Ours)

Figure 1: (a) In prior work, a fixed set of task-level experts is recruited to solve mathematical problems, while heterogeneous questions may differ in the skills required to solve them (e.g., Q1 requires algebra, while Q2 focuses on probability). The recruited experts generate outputs for multiple rounds, making these methods inefficient. (b) In contrast, SYMBOLIC-MOE adaptively recruits instance-level experts based on skills needed ("Algebra" experts for Q1 and a different set of "Probability" experts for Q2). By generating only a single round of responses with an aggregator to synthesize the final output, our approach is both more performant and more efficient.

et al., 2021). However, in the conventional MoE settings, experts are typically sub-models, i.e., subsets of parameters within a larger model, where at test time, they are combined in the model's parameter space. This generally requires end-to-end training from scratch, which is often computationally expensive and precludes the re-use of the vast pool of already-trained LLMs. Building on recent efforts in combining a fixed set of models through multi-agent discussions (Chen et al., 2024b; Du et al., 2023; Liang et al., 2023; Wang et al., 2024a), we propose exploring a new *training-free* paradigm for large-scale MoEs: a symbolic mixture of experts (SYMBOLIC-MOE). Rather than using information encoded in the model's hidden state, SYMBOLIC-MOE uses symbolic structures in two ways: First, SYMBOLIC-MOE infers a set of discrete skills needed to solve a problem, measuring the abilities of each model in a pool of candidate expert models. It then uses skill-based performance as a "router" to recruit a sparse subset of experts *for each problem*. Secondly, SYMBOLIC-MOE combines pre-trained experts through a symbolic channel, i.e., language, which is a common protocol already shared by all LLMs. To take advantage of the diverse set of expert LLMs, we must address two key challenges: **(1) Effective Expert Selection**: Given a large set of LLMs, how can we choose the best experts for each instance? **(2) Scalable Expert Mixing**: How can we serve a large number of experts (e.g. 16) without increasing the demand for GPUs?

(1) Effective Expert Selection: The increasing diversity of benchmarks (Miranda et al., 2024) and the growing number of models means that experts must be selected not at the level of tasks, but at the level of individual queries. Even at the task level, manual selection can be labor-intensive, and the performance of multi-agent frameworks can be sensitive to the agents chosen for a task (Chen et al., 2024b; Liang et al., 2023; Wang et al., 2024b). For instance, as shown in Fig. 1 (a), while a given subset of models may perform well on math tasks on average, their proficiency in specific subfields like algebra or probability might vary – that is, using a fixed subset of models on all math samples might hurt performance on particular subtasks. This underscores the need for an automated, fine-grained selection mechanism, as shown in Fig. 1 (b). **(2) Scalable Expert Mixing:** Past work has often relied on multiple rounds of inference, leading to significant GPU demands. Moreover, it does not scale to a dynamic setting like the one we consider, where the number of GPUs required would be equal to the number of potential models available (in our case, 16), making this option prohibitively expensive. We instead propose a **batch inference mechanism** that groups samples into batches per model. We then run *all queries* assigned to a given model in a single batch, which is far faster than sequential processing. While this strategy accommodates up to 16 models *on a single GPU*, it can also be parallelized across multiple GPUs. This flexibility ensures both speedups with increased computing power, and accessibility for users with limited hardware resources.

We evaluate SYMBOLIC-MOE on 6 diverse benchmarks across math, science, and medical reasoning, using a diverse model pool. We show that our automated skill-based recruiting yields an average accuracy improvement of 8.15% over the best multi-agent baseline. Moreover, despite primarily using LLMs with 7-8 billion (B) parameters, SYMBOLIC-MOE achieves comparable performance with 70B models. Also, SYMBOLIC-MOE consistently surpasses all baselines, whereas the strongest baseline changes across tasks. Thus, our method eliminates the need to evaluate and compare a large number of baselines for each task. Notably, using a single GPU, SYMBOLIC-MOE has 44% less inference run-time than a mixture-of-agents baseline (Wang et al., 2024a); when four GPUs are available for both methods, we obtain an almost $2\times$ speedup. Finally, our analysis shows that SYMBOLIC-MOE generalizes well to unseen tasks, and selecting a task-specific aggregator achieves performance comparable to multi-round discussion while requiring substantially less compute.

2 RELATED WORK

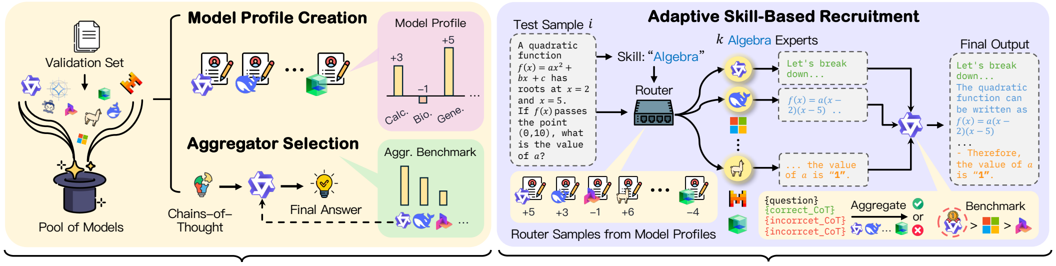
Mixture-of-Agents. Traditional Mixture-of-Experts (MoE) models (Jacobs et al., 1991; Jordan & Jacobs, 1994; Chen et al., 1999; Yuksel et al., 2012) distribute computation across multiple "expert" submodels, with growing interest in sparsity-driven approaches. The Sparse MoE (SMoE) approach (Shazeer et al., 2017a) improves efficiency by activating only the most relevant experts per input, enhancing scalability for high-dimensional data. This method has been widely applied in vision tasks (Riquelme et al., 2021; Wang et al., 2020; Yang et al., 2019; Abbas & Andreopoulos, 2020), language tasks (Lepikhin et al., 2021; Zhang et al., 2021; Zuo et al., 2022; Jiang et al., 2021) and multimodal learning (Kudugunta et al., 2021; Yun et al., 2024). Unlike SYMBOLIC-MOE, standard MoE approaches require experts to be trained jointly, with communication taking place in the parameter spaces of the submodels. Routing LLMs has also been an active research area. DER formulates routing as a Markov Decision Process and dynamically selects an optimal answering route (Hu et al., 2024). Router-R1 formulates routing and aggregation as a sequential decision process Zhang et al. (2025). On the test-time mixture side, LLM-Blender proposes ranking and fusing multiple models' output (Jiang et al., 2023). MoA (Wang et al., 2024a) combine LLM agents into ensembles that rely on a fixed set of agents. This approach requires multiple rounds of generation and aggregation before producing a final answer. Similarly, Self-MoA (Li et al., 2025) suggest that optimal performance can be achieved by invoking the task-best model multiple times alongside the task-best aggregator. Our work differs from MoA and Self-MoA by introducing *adaptive, instance-level, skill-based routing* while avoiding costly multi-model discussions in favor of streamlined aggregation. We also find that mixing different LLMs is advantageous when paired with effective routing and aggregation strategies.

Multi-Agent Reasoning. Multi-agent reasoning has emerged as a promising paradigm for enhancing complex problem-solving and decision-making in AI systems. Early approaches employed reinforcement learning-based coordination (Lowe et al., 2017; Foerster et al., 2018; Jaques et al., 2019), while recent efforts leverage LLM-based multi-agent frameworks. One line of research explores student-teacher paradigms (Magister et al., 2022; Fu et al., 2023; Ho et al., 2022; Du et al., 2023; Chen et al., 2024a), where reasoning capabilities are distilled from stronger to weaker agents. Another approach investigates multi-agent debate frameworks, where agents interact to refine arguments and enhance collective decision-making; this has been explored with multiple instances of a single model (Liang et al., 2023; Xiong et al., 2023; Chan et al., 2023) or debates between multiple LLM types (Chen et al., 2024b). In both cases, the set of models is predefined by the user. In contrast, our approach automatically selects models based on inferred skills. Additionally, our framework achieves superior efficiency by avoiding multi-round discussions.

3 METHODOLOGY

3.1 PROBLEM SETUP

Given a pool of n models $\mathcal{M} = \{M_i\}_{i=1}^{n}$, where each model represents a distinct LLM with potentially different pre-training datasets and architectures, our goal is to optimize performance through dynamic allocation – solving each problem with the most suitable subset of k models, allowing experts to combine information to enhance reasoning. To achieve this, we assume access to a small validation set to obtain (1) model profiles $P_i\ \forall i \leq n$, and (2) aggregator profiles that benchmark the ability of each model to act as an aggregator. We use these profiles to recruit experts (at the instance level) and to select the aggregator (at the task level).



(a) Preprocessing

(b) Inference

Figure 2: Overview of SYMBOLIC-MOE. (a) Preprocessing: Given a validation set and a pool of agents, we create model profiles and select an aggregator. (b) Inference-Time: For each test example, SYMBOLIC-MOE activates the most relevant models (experts) based on skill-based routing, using model profiles determined during preprocessing. These models generate CoT responses, which the aggregator (chosen based on its ability to select correct answers) synthesizes into a final answer.

3.2 PREPROCESSING

3.2.1 MODEL PROFILE CREATION

To recruit the k most suitable experts for a given question, we assess each model's specialized skills across various problem-solving domains, illustrated in Fig. 2 (a). This is done by evaluating their performance on the validation set for each task (see Table 12 for sizes), thereby constructing a model profile P_i for each model M_i. For each question in the validation set, we first prompt an LLM – referred to as the "Keyword LLM" – to identify the essential skills required to solve the problem. For consistency, we use Qwen2.5-7B-Instruct (Qwen Team, 2024) as the Keyword LLM. Later in Table 13, we show that the choice of the keyword LLM has little effect on the performance. To reduce noise, we generate keyword annotations for each question five times, and retain only those that appear more than once for each question. These extracted skills represent core knowledge areas necessary for solving the problem – for instance, a given college-level math problem may require skills such as algebra, calculus, and geometry. Once all questions are annotated with their required skills, each model M_i in the pool attempts to solve them using Chain-of-Thought reasoning (Wei et al., 2022). A correct answer increases the score of each associated skill by $+1$, while an incorrect answer results in a -1 penalty. At the end of this process, each model has a profile P_i represented as a dictionary, e.g., {'Algebra': 10, 'Biology': 3, 'Chemistry': -6, ...}.

3.2.2 AGGREGATOR SELECTION

An aggregator is a model that consolidates k outputs into a single high-quality response. Our pilot experiments, along with findings from Wang et al. (2024a) and Li et al. (2025), indicate that the aggregator model plays a crucial role in the final performance, and selecting the most effective model for aggregation is non-trivial. We find that *a strong reasoning model is not necessarily a strong aggregator and vice versa*; qualitatively, we show this later in Table 3. We also find that adaptively selecting an aggregator on the instance level based on model profiles is less effective, motivating us to choose the aggregator based on its ability to select correct answers. To identify the best aggregator per task, we construct a synthetic task using the same validation set. From the profile creation process, we obtain outputs from all models, some correct and some incorrect. For each question, we sample one correct reasoning chain and two incorrect ones, structuring the input as follows: `{question}`, `{correct_CoT}`, `{incorrect_CoT}`, `{incorrect_CoT}`. We shuffle the order of the CoTs and instruct each model to act as an aggregator (using the prompt shown in Appendix O), synthesizing a final output with a predicted answer. We then benchmark each model's aggregation ability and select the best-performing aggregator for each task.

3.3 Inference

3.3.1 Skill-based Recruiting

At inference time (see Fig. 2 (b)), we follow the same keyword annotation procedure as in Section 3.2.1 to generate relevant keywords for the test sample. To align a test sample's keywords with those in the model profiles, we employ Sentence-BERT (Reimers & Gurevych, 2020) to match keywords via the cosine similarity between their embeddings. This ensures that every keyword gets matched to the closest counterpart in the model profile. Next, expert recruitment is performed by selecting the top k models whose profiles best match the required skills of the test sample. This is determined by two factors: **(1) local suitability score** and **(2) global competency**. For each model M_i, its *local suitability score* for a test sample q, $\mathcal{S}(M_i, q)$ is computed as the sum of its skill scores over the set of keywords needed for q (denoted as K_q). It can be expressed as follows:

$$\mathcal{S}(M_i, q) = \sum_{k_j \in K_q} s_{k_j}^{(i)}$$

where $s_{k_j}^{(i)}$ represents the score of model M_i for the j-th skill in the test sample q. This results in an model ranking distribution $\mathcal{D}_q$ for each test sample q: $\mathcal{D}_q = \{\mathcal{S}(M_1, q), \mathcal{S}(M_2, q), ..., \mathcal{S}(M_n, q)\}$.

Intuitively, suppose M_1 has scores of $+3$, $+5$, and -2 for algebra, calculus, and geometry, respectively, which are needed for a given sample; its total score for this sample would be $3 + 5 - 2 = 6$. Calculating this score for all models yields a distribution of model strengths for the given sample, e.g., $\{M_1$: 6, M_2: 3, ..., M_n: -10$\}$, which is the ranking of *how suitable a model is for a sample*.

We also take into account each model's overall strength in a task, i.e., *global competency*. This is easily obtained by summing a model's score across all keywords in its profile, and normalizing it by the total sum of all models' scores. We denote this global strength as γ_i, representing a model's overall task performance *relative to others*. Finally, the expert selection is performed by sampling from the product of the local suitability score, $\mathcal{S}(M_i, q)$ (from a model rank distribution $\mathcal{D}_q$) and the global competency γ_i. That is, the relevance score of a model M_i for a test sample q is: $w_q^{(i)} = \gamma_i \mathcal{S}(M_i, q)$. We apply a softmax function with the temperature set to 0.5 to this distribution $\{w_q^{(i)}\}_{i=1}^n$, and then sample k experts with replacement for each problem, i.e.,

$$E_q^{(i)} \sim \text{Categorical}(w_q^{(1)}, w_q^{(2)}, ..., w_q^{(n)}), \ i = \{1, 2, ..., k\}$$

To enhance efficiency, we trim low-frequency experts, i.e., those who appear in fewer than 5% of test cases. For example, given a test set with 100 samples, where 3 experts are recruited per sample (totaling 300 selections), any expert appearing fewer than $300 \times 5\% = 6$ times is discarded and resampled from the remaining higher-frequency experts. To visualize this, we provide the expert distribution before and after trimming in Fig. 4.

3.3.2 Final Answer Generation

After expert recruitment, each sample will be passed to the experts, i.e., the input for each expert is the test problem, $x_0 = q$. These experts generate their reasoning paths to the problem in the form of Chain-of-Thought (Wei et al., 2022): $y_0^{(i)} = E^{(i)}(x_0) \ \forall \ i \in \{1, 2, ..., k\}$. Then, the task-specific aggregator A^* is introduced to synthesize the k outputs into a high-quality final answer (Wang et al., 2024a). That is, the final answer is produced by: $y = A^*(\|_{i=1}^k y_0^{(i)})$, where $\|$ denotes the concatenation operation. In Appendix P, we provide a detailed discussion on Symbolic-MoE in the context of sparse MoE frameworks and how it shares its design principles.

3.3.3 Scalable Batched Inference

In our experiments, we mostly consider 7B–8B parameter LLMs, which have a substantial memory footprint. Due to the adaptive nature of the recruitment process, the set of participating LLMs may change dynamically for different problems. For instance, one sample may require Qwen, Llama, and Mistral, while another may need Gemma, Exaone, and Phi. A naive implementation of this approach can lead to high latency, particularly when the required models change frequently. To mitigate these computational challenges, we introduce a novel batching strategy to maximize throughput.

Table 1: Comparison of SYMBOLIC-MOE with single-model and multi-model baselines. SYMBOLIC-MOE outperforms all multi-agent baselines and achieves performance comparable to strong proprietary models like GPT4o-mini, as well as 70B models, while primarily operating with 7-8B models. Notably, no single baseline consistently secures the second-best performance, even when the strongest models for each task are known. We **bold** the best results and underline the second-best (excluding methods using bigger or proprietary models, shown in gray).

Category	Method	Model	AIME	MMLU-Pro	MedMCQA	GPQA	Avg.
Close-Source Single Model	Zero-Shot CoT	GPT4o-mini	10.00	63.95	68.18	42.93	46.27
	Zero-Shot CoT	Gemini 1.5 Pro	36.67	76.38	72.68	61.62	61.84
	Zero-Shot CoT	DeepSeekV3	26.00	76.29	74.09	60.10	59.12
Open-Source 70B Model	Zero-Shot CoT	Qwen2.5 72B	25.55 ± 3.85	71.54 ± 0.88	69.02 ± 0.32	51.02 ± 0.27	54.28
	Zero-Shot CoT	Llama3.3 70B	32.22 ± 3.85	69.26 ± 0.47	59.78 ± 0.74	51.44 ± 0.62	53.18
Open-Source 7B Model	Zero-Shot CoT	QwenR1	55.93 ± 5.16	52.57 ± 0.45	38.72 ± 0.44	44.95 ± 1.49	48.04
	Zero-Shot CoT	Task-Best	55.93 ± 5.16	54.89 ± 0.53	55.44 ± 0.50	48.43 ± 3.10	53.62
Advanced Single Model	Self-Refine (SR)	Task-Best	53.33 ± 3.34	53.74 ± 0.20	49.57 ± 0.59	50.84 ± 3.65	51.87
	Self-Consistency (SC)	Task-Best x5	67.78 ± 1.57	56.71 ± 0.14	56.85 ± 0.11	53.54 ± 0.36	58.72
Single-Model Multi-Agent	Multi-Agent Debate	Task-Best x3	56.67 ± 6.67	56.21 ± 0.55	51.63 ± 0.80	50.51 ± 0.51	53.76
	Self-MoA	Task-Best x3	55.56 ± 5.09	55.43 ± 0.72	53.27 ± 0.60	52.86 ± 1.46	54.28
Multi-Model Multi-Agent	MoA	Top-3	41.11 ± 5.09	61.78 ± 0.25	59.29 ± 0.32	52.86 ± 3.37	53.76
	ReConcile	Top-3	50.00 ± 7.20	56.46 ± 0.10	**60.74 ± 0.43**	47.98 ± 2.32	53.80
	SYMBOLIC-MOE	Adaptive	**68.88 ± 5.08**	**63.71 ± 0.43**	59.35 ± 0.14	**57.78 ± 2.09**	**62.43**

Specifically, for a given set of instances, we precompute (using inferred skills) which LLMs will be called for each instance. We then group instances based on their required experts, as illustrated in Fig. 3 (III) and Algorithm 1 in Algorithm 1. In other words, each active expert receives all assigned instances at once, ensuring that each expert is loaded only once per batch. This enables efficient batched inference on *a single GPU* while supporting a pool of 16 LLMs. Moreover, this approach is flexible, as more GPUs can further accelerate inference through parallelization.

4 RESULTS AND ANALYSIS

4.1 EXPERIMENTAL SETUP

We evaluate 16 LLMs with sizes ranging from 3.5B to 12B parameters, with the majority falling in the 7–8B range. These include general-purpose instruction-tuned models, domain-specific fine-tuned variants on math and biology, and models distilled from DeepSeek R1's trajectories (DeepSeek-AI et al., 2025a). A complete list of models is provided in Table 8. We measure performance on a diverse range of datasets, chosen to require expertise in a number of domains. First, we consider MMLU-Pro (Wang et al., 2024c), which is a harder version of MMLU (Hendrycks et al., 2021), containing a variety of questions across 14 college-level subjects. Given its large test set of 12,000 samples and the computational cost of evaluating proprietary models, we employ stratified sampling to create a subset of 2,100 samples, ensuring each category contains 150 samples. We also evaluate on AIME 2024, which is a challenging mathematics competition dataset containing math Olympiad problems. For more science-specific reasoning, we test on GPQA Diamond (Rein et al., 2023), which contains questions across a variety of science fields written by experts, explicitly designed to be difficult to answer even by skilled humans with web access. Finally, we include MedMCQA (Pal et al., 2022), which covers questions sourced from medical entrance exams across 21 medical subjects. For each dataset, we sample around 350 samples as the validation set to create the model profiles.[2] Full dataset statistics are provided in Table 12, and the model pool we consider is shown in Appendix E. For baselines, we consider (i) the strongest task-specific LLM in the single-model setting and (ii) the top three models per task in the multi-model setting. *These task-best model selections for the baselines are also based on validation performance*, which is summarized in Table 9. Further details on baselines are provided in Appendix C and Appendix D.

4.2 MAIN RESULTS

SYMBOLIC-MOE consistently outperforms all baselines. We present the main results in Table 1. Across all domains, SYMBOLIC-MOE shows superior performance compared to all baselines, beat-

[2] For AIME, we sample validation questions from prior years' problems (2012-2023).

ing single-model baselines (e.g., SR, SC) using the best overall model, multi-agent debate with a single model (e.g., Debate, Self-MoA), as well as multi-model multi-agent baselines (e.g., MoA, ReConcile). SYMBOLIC-MOE outperforms the most competitive multi-agent baseline, Self-MoA, by 8.15% (absolute) on average, with consistent improvements across domains (e.g., 8.28% gain on MMLU-Pro, 13.45% on AIME, 4.92% on GPQA, 6.08% on MedMCQA). These gains are also seen when comparing to multi-model baselines like MoA and ReConcile that use the *top three strongest models* per domain. SYMBOLIC-MOE also substantially outperforms test-time scaling methods, such as SR (Madaan et al., 2023) and SC (Wang et al., 2023a). Surprisingly, with the task-best model, SC beats multi-agent debate baselines (e.g., Self-MoA, MoA), though it still underperforms SYMBOLIC-MOE by an average of 3.71%. This indicates that scaling test-time compute with the task-best model is a simple yet effective way to improve performance, and adaptively selecting suitable experts leads to further improvements.

SYMBOLIC-MOE generalizes well across tasks. No single baseline in Table 1 is universally effective across all tasks. For instance, while MoA performs well on MMLU-Pro, it struggles on AIME; ReConcile excels in MedMCQA but fails to generalize to GPQA. Therefore, knowing which method works best for a task is nontrivial. In contrast, SYMBOLIC-MOE consistently delivers strong performance across all domains. Moreover, while SC with the task-best model is the most competitive setting on AIME and GPQA, it falls short on MMLU-Pro and MedMCQA, where multi-agent baselines perform better. This discrepancy may stem from the broader subject diversity in MMLU-Pro and MedMCQA, whereas AIME is more math-focused, and the task-best model, QwenR1 (DeepSeek-AI et al., 2025a), delivers strong solo performance already. While QwenR1 demonstrates exceptional math and code reasoning capabilities (55.93% on AIME), leading to strong Self-Consistency performance (67.78%), it struggles to generalize to other domains such as MedMCQA, highlighting the need for a robust and flexible framework like SYMBOLIC-MOE to generalize across diverse tasks.

SYMBOLIC-MOE matches strong proprietary models and larger 70B models. In Table 1, we also find that SYMBOLIC-MOE achieves a similar average performance to models that have substantially more parameters. For example, SYMBOLIC-MOE outperforms Llama3.3 70B on AIME and GPQA and roughly matches it on MedMCQA, despite requiring only four 7-8B models (three for the experts and one for the aggregator). Similarly, SYMBOLIC-MOE outperforms or matches a number of strong proprietary models on average – for instance, it matches Gemini 1.5 Pro and outperforms GPT4o-mini, driven in part by sizable gains on AIME and GPQA.

4.3 ADDITIONAL ANALYSIS

Table 2: Accuracy on **OmniMATH**, with the model profiles from MMLU-Pro and AIME.

		Profile From	
	Model	MMLU-Pro	AIME
Omni Acc.	Debate	34.51	42.93
	MoA	31.55	47.36
	Self-MoA	14.63	48.75
	ReConcile	22.01	42.62
	Symbolic-MoE	**49.32**	**52.03**

SYMBOLIC-MOE generalizes to unseen tasks. Given the constant introduction of new, unseen data in LLM deployments, we evaluate whether SYMBOLIC-MOE can generalize using existing model profiles on an additional benchmark: OmniMATH (Gao et al., 2025), an Olympiad-level math dataset with 4.4k test samples. In Table 2, we reuse model profiles constructed from MMLU-Pro and AIME's validation set, and directly test on OmniMATH. All multi-agent baselines also use top-3 models selected from the same validation sets. When using the MMLU-Pro profile, multi-agent baselines struggle on OmniMATH because MMLU-Pro is broad and diverse, whereas OmniMATH is highly math-focused. This domain discrepancy, combined with the fact that the baselines rely on a fixed set of task-best models, leads to a -14.81% accuracy gap between the best baseline (Debate) and SYMBOLIC-MOE. In contrast, SYMBOLIC-MOE mitigates the profile-domain gap by leveraging fine-grained skill representations: rather than committing to a fixed model set, it dynamically recruits experts at the instance level based on the specific skills required. As skills transfer more readily across domains than task-specific model selections, SYMBOLIC-MOE is more robust to domain shift. Switching to the AIME profile allows the baselines to leverage math-specialized models, substantially improving their performance. Nonetheless, SYMBOLIC-MOE's instance-level recruitment still surpasses the strongest baseline (Self-MoA) by 3.28%.

Role and selection of the aggregator. Unlike most of our discussion-based multi-agent baselines, SYMBOLIC-MOE collects a single CoT and answer from each expert and combines them via an aggregator. This provides efficiency gains, as shown in Table 6; here, we investigate the role of the aggregator in our framework. While experts are selected per instance, the aggregator is chosen per task, as we find that reasoning ability does not necessarily translate to effective aggregation. Table 3 compares three strategies: (1) a randomly chosen aggregator from the model pool (**Random**), (2) an instance-level aggregator selection based on model profiling (**Adaptive**), and (3) a task-specific aggregator determined by task-level performance (**Task-specific**). Evaluated on MMLU-Pro and GPQA, the results indicate that a random aggregator substantially degrades performance, showing that the aggregator plays a crucial role. While the instance-level aggregator improves outcomes on both datasets, the task-specific aggregator outperforms it on MMLU-Pro and performs comparably on GPQA. We further find that the similar performance of instance-specific and task-specific aggregation on GPQA is due to a high degree of overlap in selected aggregators. Overall, this suggests that good reasoners will not always be good aggregators, supporting our task-based selection.

Table 3: Ablations on different aggregators in our full setting.

Aggregator	MMLU-Pro	GPQA
Random	52.29	48.92
Adaptive	57.12	**58.01**
Task-specific	**63.71**	57.78

Synergy between expert and aggregator selection. As demonstated that the selection of an aggregator plays an important role, we further investigate the synergy between good aggregator selection and good expert selection. In Table 4, we experiment with two expert selection strategies: (1) randomly recruiting k experts without using model profiles, and (2) using our model profiles to recruit experts. We also employ three settings for aggregator selection: (1) based on the task performance as SYMBOLIC-MOE uses, (2) randomly selecting an aggregator, and (3) using majority vote to select the final answer without using any aggregator. Our findings indicate that the combination of strong models with a task-specific aggregator yields the highest performance. When the aggregator is suboptimal, majority voting can serve as a robust alternative. However, when the expert models themselves are weak (chosen randomly), even a strong aggregator cannot compensate for the performance drop.

Table 4: Comparison of different expert selection and aggregation strategies.

Expert	Aggregator	GPQA
Random	Task-Specific	31.82
Recruited	Random	51.52
Recruited	Majority Vote	53.54
Recruited	Task-Specific	**57.78**

Utility of model profile. SYMBOLIC-MOE profiles models based on their skills and leverages this information to recruit experts effectively. To underscore the importance of this step, we compare several alternative selection strategies in Table 5, evaluating accuracy on GPQA. In the top-k approach, experts are fixed as the best-performing models for the task, whereas in the random-k strategy, the selected experts vary across instances. Our results demonstrate that skill-based selection is essential. Notably, although selecting the top k experts for a task may seem intuitive, it consistently underperforms compared to SYMBOLIC-MOE's adaptive instance-level expert selection. Interestingly, top-5 selection performs worse than top-3 selection, suggesting that a broader selection increases the likelihood of including weaker models, leading to performance degradation. Additionally, the random selection strategy consistently harms performance, showing a 12.86% to 15.61% drop compared to SYMBOLIC-MOE, likely also due to the inclusion of weaker experts.

Table 5: Comparison of different recruiting strategies on GPQA.

Recruiting Strategy	Acc.
Top-3 Experts	52.86
Top-5 Experts	47.68
3 Random Experts	42.61
5 Random Experts	44.92
Model Profile (Ours)	**57.78**

4.4 EFFICIENCY ANAYLSIS

Run time efficiency. Enabling our batch inference strategy (Fig. 3) substantially reduces inference time. We evaluate average run-time on GPQA and compare SYMBOLIC-MOE to a naive sequential implementation and to MoA (Table 6). As expected, sequential inference yields the highest latency, since the model must be repeatedly loaded and offloaded for each instance. In contrast, SYMBOLIC-MOE achieves 44% lower latency than MoA on a single GPU—while also delivering higher accuracy.

Notably, Symbolic-MoE on a single GPU matches the run-time of MoA on 4 GPUs, and when scaled to 4 GPUs, Symbolic-MoE attains nearly a 2× speedup over MoA. We also report token usage in Appendix I, showing that Symbolic-MoE achieves the best trade-off between efficiency and performance among multi-agent baselines. While our batch inference strategy does assume that test samples arrive in batches (so expert assignments can be pre-determined), this assumption is not restrictive in practice: many widely-used inference acceleration techniques make the same assumption (e.g, vLLM (Kwon et al., 2023)), and popular services like ChatGPT (OpenAI, 2025) and Gemini (Gemini, 2025) also support batched inference for efficiency.

Table 6: Efficiency comparison of MoA and Symbolic-MoE on the GPQA test set. Run time is averaged per sample.

Method	# GPUs	Run Time (s)
Sequential	1	196.92
MoA	1	45.98
MoA	4	21.66
Symbolic-MoE	1	25.76
Symbolic-MoE	4	**10.85**

Methodological efficiency. Like multi-agent discussion baselines, Symbolic-MoE can also operate in a discussion-based manner. Instead of immediately aggregating initial responses, models first engage in three rounds of discussion—observing each other's outputs and generating refined responses—before submitting them to the aggregator. Table 7 reports results for this setting, comparing the adaptive aggregator (suboptimal), the task-best aggregator (suboptimal), and the task-specific aggregator (optimal) on MMLU-Pro and GPQA. With the task-specific aggregator, discussion yields only marginal gains on MMLU-Pro (63.83 vs. 63.71) and a slight drop on GPQA (57.72 vs. 57.78). While round-wise discussion can improve performance incrementally, the final outcome is primarily determined by the strength of the aggregator. Consequently, Symbolic-MoE with a strong task-specific aggregator can skip the costly multi-round discussion, reducing runtime and improving methodological efficiency, while surpassing the performance of discussion-based baselines, as shown in Table 6.

Table 7: Comparison of with and without discussion across varying aggregators. We show that using a task-specific aggregator leads to the best performance, and while multi-round discussion stabilizes performance with suboptimal aggregators, it *has little effect* with an optimal aggregator.

Discuss	Aggr.	MMLU-Pro	GPQA
✓	Adaptive	59.07	57.01
✗	Adaptive	57.12	**58.01**
✓	Task-best	57.81	57.78
✗	Task-best	56.67	57.01
✓	Task-specific	**63.83**	57.72
✗	Task-specific	63.71	57.78

5 Discussion and Conclusion

A key feature highlighted in Table 1 is the *consistency* of Symbolic-MoE's performance. While baseline methods occasionally do well in isolated settings (e.g., MoA on MMLU-Pro, ReConcile on MedMCQA), it is important to highlight that no baseline does well consistently across settings. This means that – without Symbolic-MoE – getting a strong overall result would require evaluating all the baseline methods and choosing the best settings manually. In contrast, Symbolic-MoE achieves high performance by automatically recruiting the experts based on skills needed for each instance, serving as a robust recipe that generalizes across domains.

Modularity. Another key advantage of Symbolic-MoE is its modularity. Unlike typical Mixture-of-Experts (MoE) frameworks, which need to be trained end-to-end from scratch in a centralized manner, Symbolic-MoE uses the symbolic output channel of existing models to combine experts. This gradient-free approach enables seamless integration of pre-trained models without updates, allowing them to be trained independently and distributedly. Moreover, while standard MoEs have a fixed size determined before training, Symbolic-MoE can *adaptively grow and evolve* as models are updated. Given the rapid advancements in LLMs, state-of-the-art models are often replaced within months. Symbolic-MoE's modular and gradient-free design simplifies the incorporation of these updates, requiring only a few calls to obtain a new model's profile, which can then be easily plugged into this framework. It is also straightforward to increase the number of experts recruited at test time, which is usually fixed in the typical MoE setting.

Conclusion. We introduced Symbolic-MoE, a scalable MoE framework that coordinates models through symbolic outputs (i.e., natural language discussion). Symbolic-MoE identifies the skills required for a given problem, recruits the corresponding experts, and aggregates their responses into a single high-quality answer. Across four diverse reasoning datasets, SYMBOLIC-MOE consistently outperforms inference-time scaling methods, debate frameworks, and recent mixture-of-agents approaches, delivering strong and stable performance even as the best baseline varies across domains. Its average performance on heterogeneous tasks also matches advanced proprietary systems such as GPT-4o-mini and larger 70B-parameter models. Detailed analysis shows that both expert selection and the choice of aggregator are crucial for downstream performance, and with a strong aggregator, costly multi-round discussions can often be skipped without sacrificing quality. Finally, SYMBOLIC-MOE introduces a novel batching strategy that runs efficiently on a *single GPU*, while retaining the flexibility to scale further with additional GPUs—achieving both high performance and efficiency.

ETHICS STATEMENT

In this work, we propose an inference-time method, SYMBOLIC-MOE, which operates without the need for additional training or fine-tuning. Consequently, the LLMs utilized by SYMBOLIC-MOE may still exhibit stereotypes, biases, and other negative traits inherent in their pre-training data (Weidinger et al., 2021), over which we have no control. Therefore, the outputs produced by SYMBOLIC-MOE carry the same potential for misuse as those from other test-time methods. Further research is necessary to assess and mitigate these biases in LLMs.

REPRODUCIBILITY STATEMENT

We are making our code available in the supplementary materials to enable replication of our findings. We also provide implementation details of SYMBOLIC-MOE in Appendix D and prompts in Appendix O. The datasets we use are all publicly available.

LIMITATIONS

In cases where expert-grouped batches do not form, e.g., when traffic is very sparse or mixed, our proposed batch inference strategy simply reverts to the standard inference setup. Concretely: (1) With k GPUs, we load one expert model per GPU and run them independently. (2) With a single GPU, we load experts sequentially and collect their outputs. We acknowledge that this strategy is designed to improve efficiency under typical online traffic or benchmark evaluations. As with most batching methods, its efficiency benefits are less pronounced under extremely small or irregular batches, but it wouldn't affect the correctness or operation of the system.

REFERENCES

Alhabib Abbas and Yiannis Andreopoulos. Biased mixtures of experts: Enabling computer vision inference under data transfer limitations. *IEEE Transactions on Image Processing*, 29:7656–7667, 2020.

Meta AI. Meta-llama-3.1-8b-instruct. Hugging Face Model Hub, 2024. URL https://huggingface.co/meta-llama/Meta-Llama-3.1-8B-Instruct. Accessed: 2024-01-31.

AI@Meta. Llama 3.3 model card. 2024. URL https://github.com/meta-llama/llama-models/blob/main/models/llama3_3/MODEL_CARD.md.

Chi-Min Chan, Weize Chen, Yusheng Su, Jianxuan Yu, Wei Xue, Shanghang Zhang, Jie Fu, and Zhiyuan Liu. Chateval: Towards better llm-based evaluators through multi-agent debate. *arXiv preprint arXiv:2308.07201*, 2023.

Justin Chen, Swarnadeep Saha, Elias Stengel-Eskin, and Mohit Bansal. Magdi: Structured distillation of multi-agent interaction graphs improves reasoning in smaller language models. In *Forty-first International Conference on Machine Learning*, 2024a.

Justin Chih-Yao Chen, Swarnadeep Saha, and Mohit Bansal. Reconcile: Round-table conference improves reasoning via consensus among diverse llms, 2024b. URL https://arxiv.org/abs/2309.13007.

Ke Chen, Lei Xu, and Huisheng Chi. Improved learning algorithms for mixture of experts in multiclass classification. *Neural networks*, 12(9):1229–1252, 1999.

Tianzhe Chu, Yuexiang Zhai, Jihan Yang, Shengbang Tong, Saining Xie, Dale Schuurmans, Quoc V. Le, Sergey Levine, and Yi Ma. Sft memorizes, rl generalizes: A comparative study of foundation model post-training. *arXiv preprint arXiv:2501.17161*, 2025. URL https://arxiv.org/abs/2501.17161.

Herbert H Clark. *Using language*. Cambridge university press, 1996.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025a. URL https://arxiv.org/abs/2501.12948.

DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, et al. Deepseek-v3 technical report. 2025b. URL https://arxiv.org/abs/2412.19437.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B Tenenbaum, and Igor Mordatch. Improving factuality and reasoning in language models through multiagent debate. *arXiv preprint arXiv:2305.14325*, 2023.

David Eigen, Marc'Aurelio Ranzato, and Ilya Sutskever. Learning factored representations in a deep mixture of experts. *arXiv preprint arXiv:1312.4314*, 2013.

William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39, 2022. URL http://jmlr.org/papers/v23/21-0998.html.

Jakob Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. Counterfactual multi-agent policy gradients. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

Yao Fu, Hao Peng, Litu Ou, Ashish Sabharwal, and Tushar Khot. Specializing smaller language models towards multi-step reasoning. In *International Conference on Machine Learning*, pp. 10421–10430. PMLR, 2023.

Bofei Gao, Feifan Song, Zhe Yang, Zefan Cai, Yibo Miao, Qingxiu Dong, Lei Li, Chenghao Ma, Liang Chen, Runxin Xu, Zhengyang Tang, Benyou Wang, Daoguang Zan, Shanghaoran Quan, Ge Zhang, Lei Sha, Yichang Zhang, Xuancheng Ren, Tianyu Liu, and Baobao Chang. Omni-MATH: A universal olympiad level mathematic benchmark for large language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=yaqPf0KAlN.

Gemini. Batch prediction with gemini, 2025. URL https://cloud.google.com/vertex-ai/generative-ai/docs/multimodal/batch-prediction-gemini.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

Namgyu Ho, Laura Schmid, and Se-Young Yun. Large language models are reasoning teachers. *arXiv preprint arXiv:2212.10071*, 2022.

Jinwu Hu, Yufeng Wang, Shuhai Zhang, Kai Zhou, Guohao Chen, Yu Hu, Bin Xiao, and Mingkui Tan. Dynamic ensemble reasoning for llm experts. *arXiv preprint arXiv:2412.07448*, 2024.

Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87, 1991.

Natasha Jaques, Angeliki Lazaridou, Edward Hughes, Caglar Gulcehre, Pedro Ortega, DJ Strouse, Joel Z Leibo, and Nando De Freitas. Social influence as intrinsic motivation for multi-agent deep reinforcement learning. In *International conference on machine learning*, pp. 3040–3049. PMLR, 2019.

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

Hao Jiang, Ke Zhan, Jianwei Qu, Yongkang Wu, Zhaoye Fei, Xinyu Zhang, Lei Chen, Zhicheng Dou, Xipeng Qiu, Zikai Guo, et al. Towards more effective and economic sparsely-activated model. *arXiv preprint arXiv:2110.07431*, 2021.

Michael I Jordan and Robert A Jacobs. Hierarchical mixtures of experts and the em algorithm. *Neural computation*, 6(2):181–214, 1994.

Sneha Kudugunta, Yanping Huang, Ankur Bapna, Maxim Krikun, Dmitry Lepikhin, Minh-Thang Luong, and Orhan Firat. Beyond distillation: Task-level mixture-of-experts for efficient inference. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 16-20 November, 2021*, pp. 3577–3599. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.findings-emnlp.304. URL https://doi.org/10.18653/v1/2021.findings-emnlp.304.

Ananya Kumar, Aditi Raghunathan, Robbie Matthew Jones, Tengyu Ma, and Percy Liang. Fine-tuning can distort pretrained features and underperform out-of-distribution. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=UYneFzXSJWh.

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

Gregory Kang Ruey Lau, Wenyang Hu, Diwen Liu, Jizhuo Chen, See-Kiong Ng, and Bryan Kian Hsiang Low. Dipper: Diversity in prompts for producing large language model ensembles in reasoning tasks, 2025. URL https://arxiv.org/abs/2412.15238.

Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. Gshard: Scaling giant models with conditional computation and automatic sharding. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=qrwe7XHTmYb.

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

Tian Liang, Zhiwei He, Wenxiang Jiao, Xing Wang, Yan Wang, Rui Wang, Yujiu Yang, Shuming Shi, and Zhaopeng Tu. Encouraging divergent thinking in large language models through multi-agent debate. *arXiv preprint arXiv:2305.19118*, 2023.

Elita Lobo, Chirag Agarwal, and Himabindu Lakkaraju. On the impact of fine-tuning on chain-of-thought reasoning. *arXiv preprint arXiv:2411.15382*, 2024.

Ryan Lowe, Yi I Wu, Aviv Tamar, Jean Harb, OpenAI Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. *Advances in neural information processing systems*, 30, 2017.

MAA. American invitational mathematics examination - aime. in american invitational mathematics examination, 2 2024. URL https://maa.org/math-competitions/american-invitational-mathematics-examination-aime.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Sean Welleck, Bodhisattwa Prasad Majumder, Shashank Gupta, Amir Yazdanbakhsh, and Peter Clark. Self-refine: Iterative refinement with self-feedback, 2023.

Lucie Charlotte Magister, Jonathan Mallinson, Jakub Adamek, Eric Malmi, and Aliaksei Severyn. Teaching small language models to reason. *arXiv preprint arXiv:2212.08410*, 2022.

Brando Miranda, Alycia Lee, Sudharsan Sundar, and Sanmi Koyejo. Beyond scale: the diversity coefficient as a data quality metric demonstrates LLMs are pre-trained on formally diverse data, 2024. URL https://openreview.net/forum?id=506Sxc0Adp.

Ranjita Naik, Varun Chandrasekaran, Mert Yuksekgonul, Hamid Palangi, and Besmira Nushi. Diversity of thought improves reasoning abilities of llms, 2024. URL https://arxiv.org/abs/2310.07088.

OpenAI. Gpt-4o mini: advancing cost-efficient intelligence, 2024. URL https://openai.com/index/gpt-4o-mini-advancing-cost-efficient-intelligence/.

OpenAI. Batch api, 2025. URL https://platform.openai.com/docs/guides/batch.

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.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, et al. Qwen2.5 technical report. 2025. URL https://arxiv.org/abs/2412.15115.

Qwen Team. Qwen2.5: A party of foundation models, September 2024. URL https://qwenlm.github.io/blog/qwen2.5/.

Nils Reimers and Iryna Gurevych. Making monolingual sentence embeddings multilingual using knowledge distillation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 11 2020. URL https://arxiv.org/abs/2004.09813.

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, 2023. URL https://arxiv.org/abs/2311.12022.

Carlos Riquelme, Joan Puigcerver, Basil Mustafa, Maxim Neumann, Rodolphe Jenatton, André Susano Pinto, Daniel Keysers, and Neil Houlsby. Scaling vision with sparse mixture of experts. *Advances in Neural Information Processing Systems*, 34:8583–8595, 2021.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, 2017a.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc V. Le, Geoffrey E. Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017b. URL https://openreview.net/forum?id=B1ckMDqlg.

Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, Anmol Gulati, Garrett Tanzer, et al. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context, 2024a. URL https://arxiv.org/abs/2403.05530.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, et al. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*, 2024b.

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

Qineng Wang, Zihao Wang, Ying Su, Hanghang Tong, and Yangqiu Song. Rethinking the bounds of LLM reasoning: Are multi-agent discussions the key? In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 6106–6131, Bangkok, Thailand, August 2024b. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.331. URL https://aclanthology.org/2024.acl-long.331/.

Xin Wang, Fisher Yu, Lisa Dunlap, Yi-An Ma, Ruth Wang, Azalia Mirhoseini, Trevor Darrell, and Joseph E Gonzalez. Deep mixture of experts via shallow embedding. In *Uncertainty in artificial intelligence*, pp. 552–562. PMLR, 2020.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *ICLR*. OpenReview.net, 2023a. URL https://openreview.net/pdf?id=1PL1NIMMrw.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*, 2023b. URL https://openreview.net/forum?id=1PL1NIMMrw.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. MMLU-Pro: A More Robust and Challenging Multi-Task Language Understanding Benchmark. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, November 2024c.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Laura Weidinger, John Mellor, Maribeth Rauh, Conor Griffin, Jonathan Uesato, Po-Sen Huang, Myra Cheng, Mia Glaese, Borja Balle, Atoosa Kasirzadeh, et al. Ethical and social risks of harm from language models. *arXiv preprint arXiv:2112.04359*, 2021. URL https://arxiv.org/abs/2112.04359.

Kai Xiong, Xiao Ding, Yixin Cao, Ting Liu, and Bing Qin. Examining inter-consistency of large language models collaboration: An in-depth analysis via debate. *arXiv preprint arXiv:2305.11595*, 2023.

Enwei Xu, Wei Wang, and Qingxia Wang. The effectiveness of collaborative problem solving in promoting students' critical thinking: A meta-analysis based on empirical literature. *Humanities and Social Sciences Communications*, 10(1):16, 2023. ISSN 2662-9992. doi: 10.1057/s41599-023-01508-1. URL https://doi.org/10.1057/s41599-023-01508-1.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, Keming Lu, Mingfeng Xue, Runji Lin, Tianyu Liu, Xingzhang Ren, and Zhenru Zhang. Qwen2.5-math technical report: Toward mathematical expert model via self-improvement. *arXiv preprint arXiv:2409.12122*, 2024. URL https://arxiv.org/abs/2409.12122.

Brandon Yang, Gabriel Bender, Quoc V Le, and Jiquan Ngiam. Condconv: Conditionally parameterized convolutions for efficient inference. *Advances in Neural Information Processing Systems*, 32, 2019.

W. Quin Yow and Tony Zhao Ming Lim. Sharing the same languages helps us work better together. *Palgrave Communications*, 5(1):154, 2019. ISSN 2055-1045. doi: 10.1057/s41599-019-0365-z. URL https://doi.org/10.1057/s41599-019-0365-z.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. Metamath: Bootstrap your own mathematical questions for large language models. *arXiv preprint arXiv:2309.12284*, 2023.

Seniha Esen Yuksel, Joseph N. Wilson, and Paul D. Gader. Twenty years of mixture of experts. *IEEE Transactions on Neural Networks and Learning Systems*, 23(8):1177–1193, 2012. doi: 10.1109/TNNLS.2012.2200299.

Sukwon Yun, Inyoung Choi, Jie Peng, Yangfan Wu, Jingxuan Bao, Qiyiwen Zhang, Jiayi Xin, Qi Long, and Tianlong Chen. Flex-moe: Modeling arbitrary modality combination via the flexible mixture-of-experts. *arXiv preprint arXiv:2410.08245*, 2024.

Chenrui Zhang, Lin Liu, Jinpeng Wang, Chuyuan Wang, Xiao Sun, Hongyu Wang, and Mingchen Cai. Prefer: Prompt ensemble learning via feedback-reflect-refine, 2023. URL https://arxiv.org/abs/2308.12033.

Haozhen Zhang, Tao Feng, and Jiaxuan You. Router-r1: Teaching llms multi-round routing and aggregation via reinforcement learning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Zhengyan Zhang, Yankai Lin, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. Moefication: Conditional computation of transformer models for efficient inference. *arXiv preprint arXiv:2110.01786*, 2021.

Simiao Zuo, Xiaodong Liu, Jian Jiao, Young Jin Kim, Hany Hassan, Ruofei Zhang, Jianfeng Gao, and Tuo Zhao. Taming sparsely activated transformer with stochastic experts. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=B72HXs80q4.

Appendix

A The Use of Large Language Models (LLMs)

We use ChatGPT[3] for grammar correction and refinement. The model was only used to polish text written by the authors and was not used to contribute to research ideation or the generation of original content.

B Illustrations of Variations in Batch Inference

As discussed in Section 3.3, there are mainly three different ways to serve multiple LLMs to solve every instance adaptively. We illustrate these variations in Fig. 3. Our batch inference method allows for the speedups featured in Table 6.

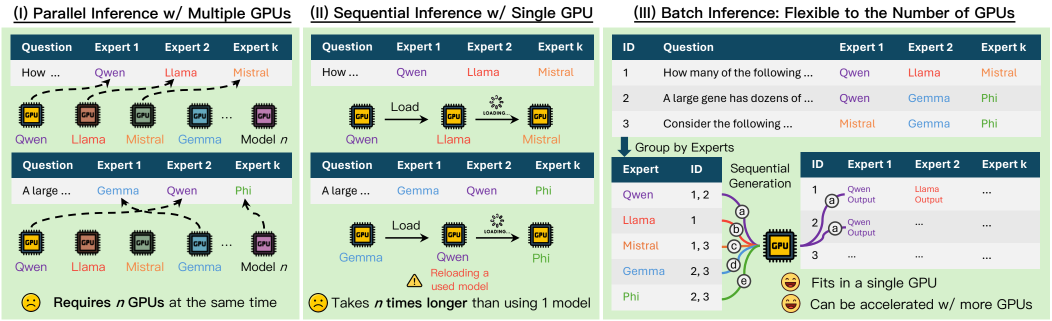
C Baselines Details

We compare against four categories of baselines.

- **Zero-shot single-model methods**: This category includes proprietary models such as GPT-4o-mini (OpenAI, 2024), Gemini 1.5 Pro (Team et al., 2024a), and DeepSeek-V3 (DeepSeek-AI et al., 2025b); high-capacity open-source models like Qwen2.5 72B (Qwen et al., 2025) and Llama 3.3 70B (AI@Meta, 2024); and strong distilled 7B models such as QwenR1 (DeepSeek-AI et al., 2025a). For reference, we also report the best task-specific model from our pool for each task, denoted as Task-Best.
- **Advanced single-model baselines with inference-time compute**: We evaluate methods that enhance inference-time reasoning, specifically Self-Refine (SR) (Madaan et al., 2023) and Self-Consistency (SC) (Wang et al., 2023b). To ensure a fair comparison, we set SC's sample size to 5, aligning with the number of large language model (LLM) calls in Symbolic-MoE, which engages three experts and one aggregator model.[4] Additionally, for these baselines, we use the

[3] https://chatgpt.com/

[4] We use an odd number of SC calls to avoid ties.


(I) Parallel Inference w/ Multiple GPUs
GPU
Qwen
Llama
Mistral
Gemma
...
Model n
Requires n GPUs at the same time
(II) Sequential Inference w/ Single GPU
Load
LOADING...
Qwen
Llama
Mistral
Gemma
Qwen
Phi
Reloading a used model
Takes n times longer than using 1 model
(III) Batch Inference: Flexible to the Number of GPUs
Group by Experts
Sequential Generation
a
b
c
d
e
GPU
Fits in a single GPU
Can be accelerated w/ more GPUs

Question	Expert 1	Expert 2	Expert k
How ...	Qwen	Llama	Mistral

Question	Expert 1	Expert 2	Expert k
A large ...	Gemma	Qwen	Phi

Question	Expert 1	Expert 2	Expert k
How ...	Qwen	Llama	Mistral

Question	Expert 1	Expert 2	Expert k
A large ...	Gemma	Qwen	Phi

ID	Question	Expert 1	Expert 2	Expert k
1	How many of the following ...	Qwen	Llama	Mistral
2	A large gene has dozens of ...	Qwen	Gemma	Phi
3	Consider the following ...	Mistral	Gemma	Phi

Expert	ID
Qwen	1, 2
Llama	1
Mistral	1, 3
Gemma	2, 3
Phi	2, 3

ID	Expert 1	Expert 2	Expert k
1	Qwen Output	Llama Output	...
2	Qwen Output	...	...
3	...	...	...

Figure 3: Different approaches to achieving adaptiveness in SYMBOLIC-MOE, which uses different models for each instance. In a naive setup (I), k GPUs must be hosted simultaneously, allowing immediate access to outputs from each model. Another naive setup (II) requires only a single GPU but involves constant loading and offloading of models to obtain outputs from the corresponding model. Our scalable batch inference process (III) strikes a balance between (I) and (II). When models are assigned to problems, we group samples by model and sequentially load the corresponding LLM onto a single GPU to generate outputs efficiently. Moreover, this approach still allows us to parallelize across GPUs if they are available.

best-performing LLM for each task, inferred on the same dev set used for our agent profile creation.

- **Single-model multi-agent baselines**: To isolate the impact of SYMBOLIC-MOE's recruitment strategy, we compare against methods where multiple instances of the same model collaborate. Specifically, we consider Multi-Agent Debate (Debate) (Du et al., 2023) and Self-Mixture-of-Agents (Self-MoA) (Li et al., 2025), both of which rely on iterative, multi-round discussions using a single model. These baselines employ three agents, each using the same task-best model, and conduct two rounds of discussion, resulting in a total of 6 LLM calls per sample.
- **Multi-model multi-agent baselines**: We also evaluate approaches leveraging diverse models in a multi-agent setup. This includes Mixture-of-Agents (MoA) (Wang et al., 2024a) and ReConcile (Chen et al., 2024b), both of which incorporate a fixed set of models in multi-round interactions. To ensure a fair comparison with our approach, particularly in the use of the validation set, we select the top three performing models from the validation set and conduct multi-round interactions. In MoA, agents participate in two rounds of discussion, while agents in ReConcile engage in three rounds, leading to 6 and 9 LLM calls per sample, respectively.

D IMPLEMENTATION DETAILS

We conduct our experiments for SYMBOLIC-MOE and other single-model baselines on a single A6000 GPU with 48 GB of memory, while MoA and ReConcile are executed on 8 A6000 GPUs for parallelization. For the 70B models, we use the original version without quantization and perform inference on 4 A6000 GPUs. All open-source models utilize vLLM (Kwon et al., 2023) for inference. The temperature is set to 0.7 for all methods. The maximum output token length is fixed at 4096 for all models, except for QwenR1 and LlamaR1, which have a limit of 32768 since they are trained with longer trajectories and tend to generate longer outputs. All results, except those from proprietary models (due to budget constraints), are averaged over three random seeds. Further details on the model pool, distribution of the expert recruited, and all the prompts we use can be found in Table 8 and Appendix O.

E MODEL POOL

We provide the full list of our model pool in Table 8, including their names, sizes, and publicly available checkpoints on Huggingface. Most of the model sizes are 7 to 8 billion.

Table 8: The models constituting the model pool.

Model Name	**Size**	**Huggingface Link**
BioLlama	8B	ContactDoctor/Bio-Medical-Llama-3-8B
DeepSeekMath	7B	deepseek-ai/deepseek-math-7b-instruct
Exaone	7.8B	LGAI-EXAONE/EXAONE-3.5-7.8B-Instruct
Gemma2	9B	google/gemma-2-9b-it
GLM4	9B	zai-org/glm-4-9b-chat
Granite	8B	ibm-granite/granite-3.1-8b-instruct
InternLM3	8B	internlm/internlm3-8b-instruct
Llama3.1	8B	meta-llama/Llama-3.1-8B-Instruct
LlamaR1	8B	deepseek-ai/DeepSeek-R1-Distill-Llama-8B
Mathstral	7B	mistralai/Mathstral-7B-v0.1
Mistral	12B	mistralai/Mistral-Nemo-Instruct-2407
Phi3.5-mini	3.5B	microsoft/Phi-3.5-mini-instruct
Qwen2.5	7B	Qwen/Qwen2.5-7B-Instruct
Qwen2.5-Coder	7B	Qwen/Qwen2.5-Coder-7B-Instruct
Qwen2.5-Math	7B	Qwen/Qwen2.5-Math-7B-Instruct
QwenR1	7B	deepseek-ai/DeepSeek-R1-Distill-Qwen-7B

F PERFORMANCE ON THE VALIDATION SET

Table 9 shows the performance of each model on the validation set. We highlight the top-1 and top-3 models in bold font and yellow background, respectively. This information is also used for the baselines we compare against in Table 1.

Table 9: Comparison of model performance on the validation set. The best model on each task is **bolded**, and the top 3 models on each task are highlighted in yellow.

Model	**MMLU-Pro**	**AIME**	**GPQA**	**MedMCQA**
BioLlama	37.71	0.85	27.31	42.86
DeepSeekMath	32.57	3.32	28.11	35.71
Exaone	52.29	25.99	32.13	56.35
Gemma	53.71	7.73	36.95	64.29
GLM	50.29	7.37	30.92	58.33
Granite	43.43	5.92	34.14	56.15
InternLM	43.14	7.91	36.14	55.56
Llama	46.00	6.78	33.73	66.87
LlamaR1	**54.29**	51.98	**56.22**	53.37
Mathstral	34.57	3.11	36.55	52.38
Mistral	45.14	1.41	33.73	46.43
Phi	46.57	1.41	47.79	65.87
Qwen	54.00	13.56	37.35	**67.06**
QwenCode	46.29	9.89	30.52	50.79
QwenMath	31.71	11.13	28.51	36.90
QwenR1	53.43	**57.06**	51.41	37.90

G PERFORMANCE OF EACH MODEL AS AN AGGREGATOR

Table 10 shows the performance of each model when acting as an aggregator. Note that the best-performing model in Table 9 can be different from the best aggregator model in Table 10, motivating us to choose the aggregator based on this synthetic task described in Section 3.2.2.

Table 10: Performance of each model when used as an aggregator, on the validation set. The best model on each task is **bolded**, and is selected as the task-specific aggregator.

Model	**MMLU-Pro**	**AIME**	**GPQA**	**MedMCQA**
BioLlama	37.31	21.47	30.12	42.46
DeepSeekMath	32.57	5.37	21.69	35.71
Exaone	57.43	47.92	35.34	52.58
Gemma	49.71	3.11	31.73	53.37
GLM	52.57	26.27	35.34	51.39
Granite	48.86	36.44	38.96	48.02
InternLM	55.14	16.95	42.57	51.59
Llama	51.14	11.86	40.56	50.60
LlamaR1	**59.71**	53.67	46.18	49.01
Mathstral	41.71	26.27	35.74	46.43
Mistral	48.00	18.93	33.33	46.43
Phi	27.71	9.04	26.10	25.40
Qwen	56.86	38.14	39.36	**53.37**
QwenCode	51.14	29.66	38.96	50.79
QwenMath	31.71	5.93	16.06	36.90
QwenR1	58.00	**57.63**	**48.59**	45.44

H Distribution of Experts

We present the distribution of recruited experts across different datasets in Fig. 4. As noted in Section 3.2.1, we trim experts with occurrences below 5% to reduce model loading time. In Fig. 4, the top row shows the distribution before trimming, and the bottom row shows the distribution after trimming. The distribution varies significantly across datasets – on more diverse datasets such as MMLU-Pro, the recruited experts are also more varied. In contrast, for AIME and GPQA, which focus more on math and science, the recruited experts are dominated by a few models.

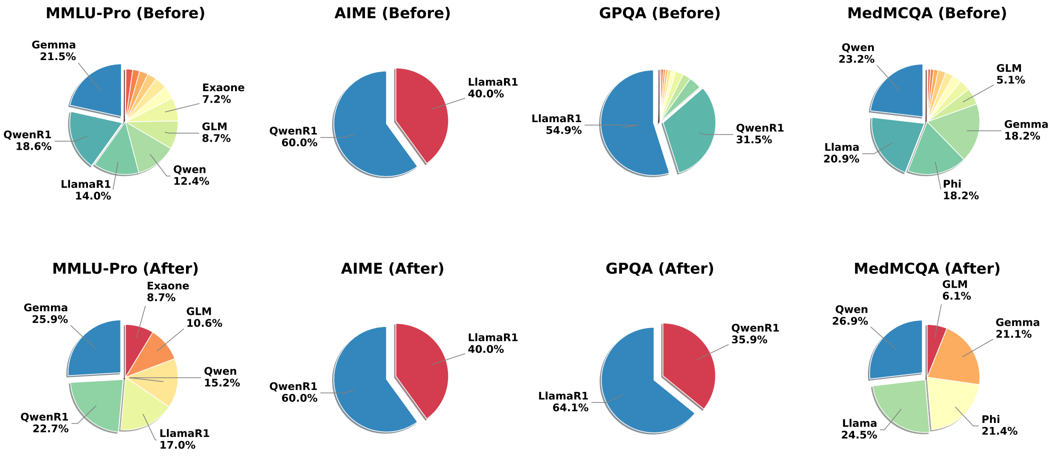


Figure 4: Distribution of the recruited experts across datasets. Top row: the distribution before trimming. Bottom row: the distribution after trimming and resampling.

I Test-time Token Count Analysis

In addition to measuring GPU run time in Table 6, we compare the test-time token count with multi-agent baselines. As shown in Fig. 5, Symbolic-MoE uses fewer tokens than Self-MoA while achieving a significant performance gain. However, compared to MoA and ReConcile, SYMBOLIC-MOE generates more tokens, particularly on GPQA. The primary reason, as illustrated in Fig. 4, is that SYMBOLIC-MOE predominantly recruits LlamaR1 and QwenR1, both of which are trained with long reasoning trajectories, resulting in substantially longer outputs compared to other models. This explains why SYMBOLIC-MOE requires less GPU run time despite producing more tokens: by skipping the expensive multi-round discussions, we eliminate the time spent loading and offloading models. However, the inherent verbosity of the R1 models contributes to the higher token count.

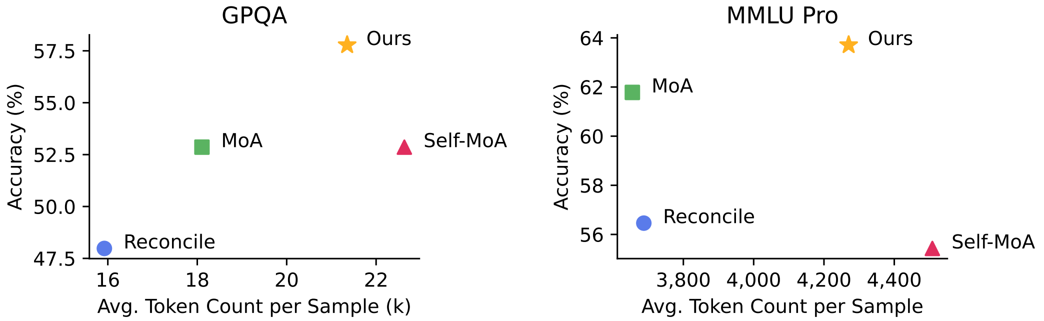


Figure 5: Comparison of the test-time token count used in different methods.

J THE EFFECT OF TRIMMING AND RE-SAMPLING

Table 11: Trimming the low-frequent experts improves both accuracy and efficiency.

	MMLU-Pro		GPQA	
	Acc ↑	Time ↓	Acc ↑	Time ↓
w/o Trimming	**63.94**	18.83	55.26	21.78
w/ Trimming	63.71	**12.27**	**57.78**	**10.85**

As described in Section 3.3.1, we trim the recruited experts if their occurrence falls below 5% of the total selections. Here, we analyze the impact of this trimming process. Without trimming, the diversity of selected experts increases, but the model loading time also increases. Table 11 presents a quantitative comparison of accuracy and GPU run time using 4 GPUs. As expected, trimming reduces GPU run time across both datasets by minimizing the need to load infrequently used models. Interestingly, we also observe that trimming improves accuracy on GPQA. This improvement may be due to the fact that after trimming, only LlamaR1 and QwenR1 remain as the recruited experts, which are particularly effective on this task.

K KEYWORD DISTRIBUTION IN VALIDATION DATA

We provide the annotated keyword distribution in the validation set in Fig. 6.

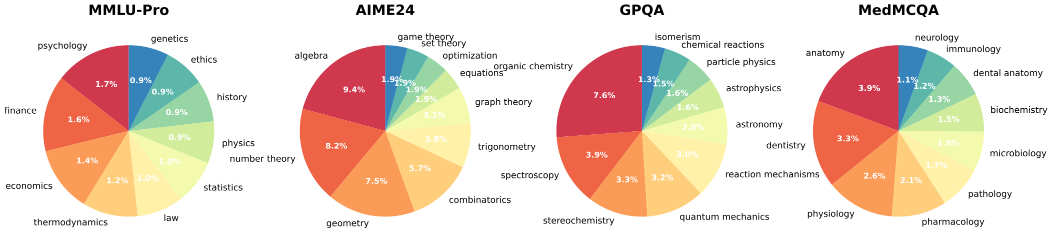


Figure 6: Keyword distribution in the validation set. For brevity, we show only the top 10 keywords.

L Dataset Statistics and Licenses

We provide the sample sizes and licenses of the datasets used in this work in Table 12. All the datasets are in English, and all datasets are used in a fashion consistent with their intended use.

Table 12: The statistics and licenses of the datasets we use in this work.

	Validation Size	Test Size	License
MMLU-Pro (Wang et al., 2024c)	350	2,100	Apache License
AIME (MAA, 2024)	354	30	CC0
GPQA (Rein et al., 2023)	249	198	MIT License
MedMCQA (Pal et al., 2022)	504	4,183	MIT License

M Sensitivity to the keyword LLM

We choose Qwen 2.5 7B (Qwen Team, 2024) as the "Keyword LLM" to generate the required skills for each instance during both preprocessing and inference. Here, we investigate the sensitivity of the results to the choice of the Keyword LLM, testing three different models: Qwen 2.5 7B (Qwen Team, 2024), Llama 3.1 8B (AI, 2024), and Gemma 2 9B (Team et al., 2024b). As shown in Table 13, the final performance remains consistent regardless of the chosen model, indicating that the selection of the Keyword LLM has minimal influence on performance.

Table 13: Keyword LLM has little effect on the final performance.

Keyword LLM	**MMLU-Pro**	**GPQA**
Llama 3.1 8B	64.19	56.62
Gemma 2 9B	64.02	57.01
Qwen 2.5 7B	63.71	57.78

N Algorithm

We provide the algorithm for our batched inference strategy in Algorithm 1.

Algorithm 1 BatchedInference

```
Require: Test samples $\mathcal{Q}$, Model pool $\mathcal{M}$
Ensure: Inference results for all samples
expert_sample_map ← $\emptyset$                                   ▷ Expert-to-samples mapping
for $q \in \mathcal{Q}$ do
    $E_q^{(1)}, E_q^{(2)}, ..., E_q^{(k)} \leftarrow$ RecruitExperts$(q, \mathcal{M})$   ▷ Select $k$ experts per sample (§3.3.1)
    for $e \in E_q$ do
        expert_sample_map$[e] \leftarrow$ expert_sample_map$[e] \cup \{q\}$
    end for
end for

results ← $\emptyset$                                             ▷ Results collection
for $(e, q_e) \in$ expert_sample_map do
    results ← results $\cup$ $e$.Generate$(q_e)$                 ▷ Batch inference per expert
end for
return results
```

O Prompts

Prompt for the Keyword LLM to Generate Keywords

Question: {question}

What are the core knowledge, subjects or skills needed to solve this problem? List 2-5 keywords separated in comma. Example keywords: psychology, virology, behavioral theory, microbiology, diplomacy, political science, property law, finance, business. Give ONLY the keywords, no other words or explanation.

Follow this format: Keywords: <keyword1>, <keyword2>...

Prompt for Zero-shot Chain-of-Thought Generation (Multiple Choice)

Question: {question}

Provide your step-by-step reasoning first, and then print "The answer is (X)" where X is the answer choice (one capital letter), at the end of your response.

Prompt for Zero-shot Chain-of-Thought Generation (Math)

Question: {question}

Provide your step-by-step reasoning first, and then print "The answer is \\boxed{X}", where X is the final answer, at the end of your response.

Prompt for the Aggregator (Wang et al., 2024a)

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

Responses from models:

{model_1_response}

{model_2_response}

{model_3_response}

Question: {question}

Provide your step-by-step reasoning first, and then print "The answer is (X)" where X is the answer choice (one capital letter), at the end of your response.

P Symbolic-MoE as a Sparse Mixture-of-Expert

In the Sparse Mixture-of-Experts (SMoE) framework (Shazeer et al., 2017a), a trainable router dynamically selects a subset of experts for each input. Formally, given an input x, the output of an SMoE layer, y is computed as:

$$\begin{aligned} y &= \sum_{i=1}^{k} \mathcal{R}(x)_i \cdot f_i(x), \\ \mathcal{R}(x) &= \text{softmax}(\texttt{Top-K}(g(x)), k) \end{aligned} \tag{1}$$

where $f_i(x)$ represents the response of the i-th expert, and $\mathcal{R}(x)$ is a trainable router that assigns selection probabilities to each expert based on $g(x)$, typically a small feedforward network (Shazeer et al., 2017b; Riquelme et al., 2021). The `Top-K` operation retains only the top k experts, setting the probabilities of others to zero after the softmax operation.

However, directly applying SMoE in our framework presents key challenges. Unlike SMoE, our method operates in a symbolic, text-based space and is designed for test-time inference, meaning that we do not rely on a trainable router to learn expert selection, nor do the experts in our method refer to model parameters. Instead, we introduce a skill-based routing mechanism to select relevant experts based on predefined competencies rather than learned gating functions. Formally, our aggregation process can be expressed as:

$$\begin{aligned} y &= A^*(\|_{i=1}^{k} y^{(i)}) \\ y^{(i)} &= E^{(i)}(x) \; \forall \; i \in \{1, 2, ..., k\} \\ E^{(i)} &\sim \text{Categorical}(w^{(1)}, w^{(2)}, ..., w^{(n)}) \; \forall i \leq k \end{aligned} \tag{2}$$

where A^* is the aggregator model determined via validation set, and $\|$ denotes the concatenation of experts' responses, i.e., $y^{(\cdot)}$. Here, $y^{(j)}$ represents the output of expert j's forward response given an input x, defined as $E^{(j)}(x)$. Each expert $E^{(i)}$, $\forall i \leq k$ is selected from our proposed skill-based routing strategy (Section 3.3.1). In short, we construct model profiles using a validation set to evaluate each model's specialization across different skills. This allows us to estimate a probability distribution $w^{(j)}$ over models based on both their suitability for the required skills and their global competence relative to other experts.

This skill-based routing framework retains the core benefits of SMoE while removing the reliance on a trainable gating mechanism. Specifically, the aggregator model A^* in SYMBOLIC-MOE plays a role analogous to the weighted sum ($\sum$) operation in SMoE, synthesizing outputs from selected experts. Likewise, the recruited agent $E^{(i)}$ corresponds to the `Top-k` operation in SMoE, ensuring that only the most relevant and specialized experts contribute to the final output. We inherit the key conceptual benefits of SMoE – dynamic expert selection and response aggregation – while also introducing additional advantages. SYMBOLIC-MOE is gradient-free, eliminating the need for retraining, and is entirely automatic, leveraging a large pool of pre-trained models to deliver a better performance.

Q DISCUSSION WITH DIVERSITY-BASED PROMPT ENSEMBLES

Recent work has explored improving LLM performance through prompt diversity using a single model. PREFER (Zhang et al., 2023) employs an AdaBoost-inspired framework in which prompts serve as weak learners, iteratively refined through a feedback-reflect-refine loop that converts training errors into natural language feedback for prompt optimization. The method then ensembles these learned prompts via weighted voting, with weights determined by each prompt's performance on reweighted training instances. DIPPER (Lau et al., 2025) takes a training-free approach, generating diverse candidate prompts and selecting a subset that maximizes a Fidelity-Adjusted Semantic Volume (FASV) metric, which balances semantic diversity (computed from response embeddings) with task fidelity. Diversity of Thought (Naik et al., 2024) similarly leverages prompt diversity by using an LLM to extract different reasoning approaches and personas, and then augmenting few-shot examples with these extracted strategies. Collectively, these methods operate on the hypothesis that varied prompting strategies can elicit more comprehensive reasoning from a single model, aggregating responses that originate from the same LLM prompted in different ways.

Symbolic-MoE addresses an orthogonal dimension, focusing on model diversity rather than prompt diversity. While the above methods ask "can we prompt one model in multiple ways?", we ask "can we leverage complementary expertise across heterogeneous models?" This distinction yields several unique advantages. First, Symbolic-MoE breaks the single-model capability ceiling. Prompt diversity is fundamentally limited by the base model's knowledge and reasoning abilities; no amount of rephrasing can make a general-purpose model match a domain specialist on its specialty (for example, a medical model on clinical reasoning or a math-specialized model on olympiad problems). Symbolic-MoE sidesteps this by recruiting models with genuinely different training data, architectures, and areas of specialization. Second, it enables adaptive instance-level routing. Rather than applying a fixed set of prompts to all problems, the system dynamically selects which expert models to activate based on inferred skills for each instance. Third, it provides inference efficiency. Prompt-diversity methods such as PREFER require iterative refinement phases, and methods like DIPPER or Diversity of Thought still generate and evaluate many candidate prompts per instance. Symbolic-MoE achieves superior performance (8.15% average improvement over the best baseline in Table 1) with a single forward pass per recruited expert, and completes inference in 44% less time than multi-round discussion baselines such as MoA (Table 6).

These two directions are ultimately complementary, since one could combine prompt diversity with model diversity. However, our results show that model heterogeneity alone already provides substantial gains, and that skill-based routing effectively identifies which models to activate without requiring prompt-level optimization.