LLMRouterBench: A Massive Benchmark and Unified Framework for LLM Routing

Hao Li[1,2,*,‡] **Yiqun Zhang**[2,*‡] **Zhaoyan Guo**[1,2,*] **Chenxu Wang**[2,*‡]
Shengji Tang[2] **Qiaosheng Zhang**[2] **Yang Chen**[2] **Biqing Qi**[2]
Peng Ye[2] **Lei Bai**[2] **Zhen Wang**[1,2,†] **Shuyue Hu**[2,†]
[1] Northwestern Polytechnical University [2] Shanghai Artificial Intelligence Laboratory
{li.hao,guozhaoyan}@mail.nwpu.edu.cn w-zhen@nwpu.edu.cn
{zhangyiqun,wangchenxu,tangshengji,zhangqiaosheng}@pjlab.org.cn
{chenyang4,qibiqing,yepeng,bailei,hushuyue}@pjlab.org.cn

Abstract

Large language model (LLM) routing assigns each query to the most suitable model from an ensemble. We introduce LLMRouterBench, a large-scale benchmark and unified framework for LLM routing. It comprises over 400K instances from 21 datasets and 33 models. Moreover, it provides comprehensive metrics for both performance-oriented and performance-cost trade-off routing, and integrates 10 representative routing baselines. Using LLMRouterBench, we systematically re-evaluate the field. While confirming strong model complementarity—the central premise of LLM routing—we find that many routing methods exhibit similar performance under unified evaluation, and several recent approaches, including commercial routers, fail to reliably outperform a simple baseline. Meanwhile, a substantial gap remains to the Oracle, driven primarily by persistent model-recall failures. We further show that backbone embedding models have limited impact, that larger ensembles exhibit diminishing returns compared to careful model curation, and that the benchmark also enables latency-aware analysis. All code and data are available at https://github.com/ynulihao/LLMRouterBench.

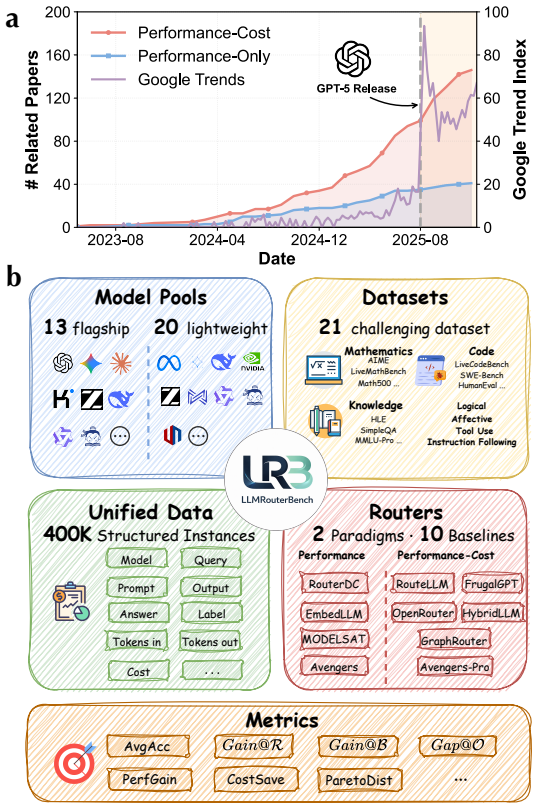

Figure 1: (a) Interest in LLM routing over time, as measured by the cumulative number of related papers and Google Trends. (b) An Overview of LLMRouterBench.

1 Introduction

The rapid evolution of large language models (LLMs) has led to a proliferation of publicly available models. In this landscape, LLM routing has emerged as an important direction: rather than relying on a single model, routing methods operate over an ensemble of LLMs and dynamically assign each query to the model best suited to handle it. Since early studies in 2023 that focused primarily on improving the collective performance of LLM ensembles (Jiang et al., 2023; Lu et al., 2024; Huang et al., 2025a), the field has broadened to address performance-cost trade-offs (Ding et al., 2024; Wang et al., 2025b; Ding et al., 2025; Jitkrittum et al., 2025); more recently, the integration of real-time routing into production systems such as GPT-5 (OpenAI, 2025a) and HuggingChat-Omni (Hugging Face, 2025) has further boosted academic, industrial, and even public interest (Fig. 1 (a)).

Despite its academic origins, progress in this field risks becoming increasingly marginalized within the research community for two primary reasons. First, developing a routing method typ-

[*] Equal contribution.
[†] Corresponding author.
[‡] This work was done during their internship at Shanghai Artificial Intelligence Laboratory.

Findings of the Association for Computational Linguistics: ACL 2026, pages 37733–37754

ically requires evaluating an ensemble of LLMs across a wide range of datasets; this incurs *substantial computational costs* from deploying multiple large models on dedicated hardware or *significant financial costs* from relying on online service providers (e.g., OpenRouter). Second, the *lack of open-source infrastructure* has led to a highly fragmented research landscape: individual studies implement bespoke routing pipelines using different model ensembles, datasets, and evaluation protocols, hindering fair comparison, reproducibility, and cumulative progress in the field.

To reduce these barriers for the community, we introduce *LLMRouterBench*, a benchmark and framework for LLM routing (Fig. 1 (b)). As summarized in Table 1, unlike existing benchmarks that rely largely on earlier-generation open-source models or evaluate only a narrow class of routing methods, LLMRouterBench provides *timely*, *large-scale*, *high-quality* data curated from 21 datasets and 33 models. These include 13 recently released flagship models and 5 widely used proprietary models, totaling over 400K instances, $2.7K in API costs, and approximately 1K GPU hours. Beyond scale, LLMRouterBench natively supports both dominant research paradigms in this field (i.e. performance-oriented routing and performance-cost tradeoff routing), defines a comprehensive suite of metrics for each setting, and offers adapters to interface with publicly available routing implementations with minimal modification; crucially, it enables *reproducible*, *unified* evaluation across 10 representative routing baselines.

With LLMRouterBench, we systematically re-examine the landscape of LLM routing and uncover several key findings. First, we reaffirm the field's central premise: models exhibit clear *complementarity* both in performance and cost-efficiency, with *no* single model achieving universal dominance. Second, by evaluating routing baselines on a unified model ensemble and dataset suite, we identify a surprising pattern: despite ongoing methodological innovation, contemporary routing approaches deliver nearly *indistinguishable* results in performance across various metrics. This pattern also extends to performance-cost tradeoff settings: several recent routing methods, even including the commercial router OpenRouter, do *not* outperform a simple baseline (the Best Single model), *nor* do they reliably reduce cost without sacrificing performance relative to Best Single. These findings suggest that under unified, large-scale evaluation, the practical gains of current routing methods may be *less* significant than previously assumed.

Third, although the above results may initially appear discouraging, we find that LLM routing remains far from its capability ceiling: a substantial performance gap persists relative to an Oracle baseline that always selects the best-performing model per query in hindsight. This gap is primarily driven by persistent *model-recall failures*: for many queries, only a single candidate model produces a correct response, yet current routers frequently fail to identify it, highlighting a clear opportunity for future improvement. Fourth, while several routing methods rely on embedding models, our ablation study shows that embeddings have *little* impact on routing performance. Fifth, although expanding the model ensemble is often assumed to improve collective performance (Huang et al., 2025b), we observe clear *diminishing* returns from adding more models; in contrast, a carefully selected subset can yield substantially better outcomes. This suggests that model curation should be studied jointly with routing, as careful curation can outweigh simply scaling the ensemble. Finally, we show that LLMRouterBench also enables latency-aware analysis, opening a path toward performance-cost-latency optimization in future routing research.

Our key contributions are summarized as follows:

- **Timely, high-quality, large-scale data:** 400K+ instances curated from 21 challenging datasets and 33 recently released models, lowering the cost barrier and improving the accessibility of LLM routing research;
- **An open-source, unified routing framework:** integration of 10 representative routing baselines and comprehensive quantitative metrics in both performance-oriented and performance-cost tradeoff settings, enabling fair, reproducible evaluation;
- **A systematic re-examination of LLM routing:** an extensive comparative study of leading routing methods that systematically analyzes their strengths and limitations, characterizes the capability ceiling, rigorously tests common practices and prevailing hypotheses, and outlines promising directions for future research.

2 Related Work

Routing for Performance. This paradigm aims to enhance collective performance of an ensemble

Benchmark	Performance-oriented	Performance–Cost	Datasets	Proprietary Models	Unified Model Pool	Routing Baselines	Data Release
RouterBench	✗	✓	8	GPT-4&3.5, Claude v1&2	✓	3	✓
EmbedLLM	✓	✗	10	✗	✓	1	✓
RouterEval	✓	✗	12	✗	✓	4	✓
FusionFactory	✗	✓	14	✗	✓	5	✓
RouterArena	✗	✓	23†	✓‡	✗	✗	✗
LLMRouterBench	✓	✓	21	GPT-5, GPT-5-Chat, Claude 4 Gemini 2.5 Pro...	✓	10	✓

Table 1: Comparison with existing routing benchmarks. †RouterArena treats the routing system as a black box, reporting only overall performance without per-prompt, per-model details, and thus cannot directly support developing new routing algorithms. ‡Each routing system uses different models.

Benchmark	Routing Baseline
RouterBench	KnnRouter, MLPRouter, SvmRouter
EmbedLLM	EmbedLLM (Zhuang et al., 2024)
RouterEval	KnnRouter, MLPRouter, Multi-class classification Router, RoBERTa-Kmeans Router
FusionFactory	KnnRouter, BertRouter, MLPRouter, SvmRouter, GraphRouter (Feng et al., 2025a)
RouterArena	✗
LLMRouterBench	RouterDC (Chen et al., 2024b), EmbedLLM (Zhuang et al., 2024), MODEL-SAT (Zhang et al., 2025b) Avengers (Zhang et al., 2025d), HybridLLM (Ding et al., 2024), FrugalGPT (Chen et al., 2024a) RouteLLM (Ong et al., 2024), GraphRouter (Feng et al., 2025a), Avengers-Pro (Zhang et al., 2025c) OpenRouter (OpenRouter, Inc., 2025b)

Table 2: Benchmarks and the corresponding routing baselines provided in their official codebases.

of LLMs by routing each query to the model that is most likely to produce a correct answer (Yue et al., 2025; Zhang et al., 2025a; Fein-Ashley et al., 2025; Wang et al., 2025a). LLM-Blender (Jiang et al., 2023) selects top candidate models through pairwise ranking and fuses their outputs. RouterDC (Chen et al., 2024b) applies dual contrastive learning to boost routing accuracy. EmbedLLM (Zhuang et al., 2024) leverages compact model and query embeddings. MODEL-SAT (Zhang et al., 2025b) constructs capability-aligned embeddings to train a lightweight LLM as a router. Avengers (Zhang et al., 2025d) shows that a clustering-based routing, combined with aggregation methods, can empower a set of small LLMs to surpass proprietary models.

Routing for Performance-Cost Tradeoff. This paradigm routes queries to models to tradeoff between performance and cost (Song et al., 2025; Jin et al., 2025; Patel et al., 2025; Fernandez et al., 2025; Guo et al., 2025b), as more capable models are often associated with higher financial and computational costs. Earlier studies primarily focused on routing between only two models (typically a small one and a larger one). HybridLLM (Ding et al., 2024) targets the tradeoff based on predicted query difficulty. FrugalGPT (Chen et al., 2024a) adopts cascaded inference from small to large models. RouteLLM (Ong et al., 2024) trains the router from preference data. More recently, research has extended to larger model ensembles. Avengers-Pro (Zhang et al., 2025c) are both build upon clustering-based routing. GraphRouter (Feng et al., 2025a) employs a heterogeneous graph to represent task-query-LLM interactions, formulating model routing as an edge prediction problem.

Routing Benchmark. Recent work has proposed several benchmarks for evaluating LLM routing. RouterBench (Hu et al., 2024) targets multi-LLM routing but is restricted to early-generation models and eight relatively simple datasets. EmbedLLM (Zhuang et al., 2024), RouterEval (Huang et al., 2025b), and FusionFactory (Feng et al., 2025b) benchmark routing over open-source models, with EmbedLLM and RouterEval providing no inference cost information. RouterArena (Lu et al., 2025) is a concurrent but distinct effort that treats routing systems as black boxes and constructs a dataset for comparing routing systems. Specifically, it focuses on black-box router evaluation and uses different model pools across routers, whereas LLMRouterBench releases per-prompt, per-model evaluated outcomes under fixed model pools, directly supporting router training, controlled cross-method comparison, and fine-grained diagnosis. As summarized in Tables 1 and 2, existing benchmarks

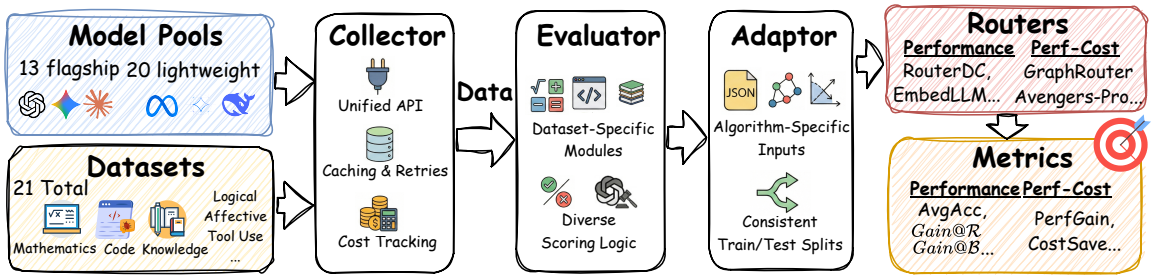

Figure 2: LLMRouterBench framework integrating Collector, Evaluator, and Adaptor components for standardized evaluation of LLM routing methods.

narrowly focus on a small number of relatively easy tasks, lack coverage of flagship models with realistic inference costs, and do not provide a unified interface for fair comparisons across routing methods. These gaps motivate LLMRouterBench, a massive routing benchmark that spans diverse and challenging tasks, jointly evaluates lightweight and flagship models under realistic costs, and offers a unified interface for plug-and-play comparison of routing methods.

3 LLMRouterBench

We construct the LLMRouterBench to systematically support two dominant paradigms in LLM routing research: routing for *performance* and routing for *performance-cost tradeoff*.

3.1 Model Pools

In the performance-oriented setting, we benchmark routing methods' ability to exploit complementary strengths among comparable-sized LLMs.* In contrast, the performance-cost setting intentionally includes models with substantial variations in size, capability, and cost to reflect realistic performance-cost tradeoffs. This yields two pools totaling 33 models:

- For the performance-oriented setting, we construct a pool of 20 state-of-the-art ∼7B **lightweight LLMs**, such as DS-Qwen3 and Qwen3-8B (see Appendix Table 10 for details)
- For the performance-cost setting, we build another pool of 13 **flagship LLMs** from 8 providers, varying substantially in capability and cost, such as GPT-5 and Gemini-Flash, with cost information collected from OpenRouter (OpenRouter, Inc., 2025b) and official APIs (see Appendix Table 11 for details).

*It is trivial to consider models with significant variations in sizes, as larger models typically yield better performance than smaller models.

3.2 Datasets

We curate 21 datasets spanning multiple domains (full list in Appendix Table 12), including **Mathematics** (e.g., AIME, LiveMathBench (Liu et al., 2024c)), **Code** (e.g., HumanEval (Chen et al., 2021a), SWE-Bench (Jimenez et al., 2023)), **Logic** (e.g., BBH (Suzgun et al., 2022), KORBench (Ma et al., 2024)), **Knowledge** (e.g., HLE (Phan et al., 2025), SimpleQA (Wei et al., 2024)), **Affective** (e.g., MELD (Poria et al., 2019)), **Instruction Following** (e.g., ArenaHard (Li et al., 2024)), and **Tool Use** (e.g., τ^2-Bench (Barres et al., 2025)).

For flagship models under the performance-cost setting, we select 10 datasets, excluding saturated datasets to effectively evaluate frontier capabilities. For lightweight models under the performance-oriented setting, we select 15 datasets, excluding excessively challenging datasets on which all models perform uniformly poorly to ensure meaningful comparisons (see Appendix Table 12 for details).

3.3 Modular Design

To support flexible integration of diverse models, datasets, and routing algorithms, LLMRouterBench (see Fig. 2) adopts a modular design built around three key modules: *Collector*, *Evaluator*, and *Adaptor*. The Collector exposes a unified API to candidate LLMs, handling caching, retries, and cost tracking while consolidating all model outputs into the standardized format described above. The Evaluator implements rigorous, dataset-specific evaluation metrics (see Appendix Table 12 for details) to ensure fair comparisons across models. The Adaptor converts the standardized format into algorithm-specific inputs for each routing method, maintaining consistent train-test splits given the same random seed. This modular design allows LLMRouterBench to interface with publicly available routing implementations with minimal modification, facilitating unified evaluation and faithful

alignment with their original implementations.

3.4 Evaluation Metrics

Performance metrics Let $\mathcal{D}$ be the set of evaluation datasets, with $d \in \mathcal{D}$ a generic dataset. For each routing method a, let $\mathrm{Acc}(a, d)$ denote its accuracy on d, computed based on the correctness of the answers produced by the model selected by a for each query in d. The Average Accuracy (AvgAcc) for each routing method a is then given by $\mathrm{AvgAcc} = \frac{1}{|\mathcal{D}|} \sum_{d \in \mathcal{D}} \mathrm{Acc}(a, d)$.

We propose to compare each routing method to three baselines: (1) Random Router ($\mathcal{R}$) randomly selects a model from the candidate pool per instance; (2) Best Single ($\mathcal{B}$) selects a single model with the highest average accuracy across all datasets in hindsight; (3) Oracle ($\mathcal{O}$) selects, for each instance, a model that yields a correct prediction if such a model exists in hindsight, choosing the one with minimum cost when multiple exist. Random Router serves as a lower-bound baseline, while Best Single and Oracle provide meaningful reference points for achievable performance. We propose three metrics based on these baselines. We define $\mathrm{Gain@b} = \frac{1}{|\mathcal{D}|} \sum_{d \in \mathcal{D}} \left(\frac{\mathrm{Acc}(a,d)}{\mathrm{Acc(b},d)} - 1 \right)$ and $\mathrm{Gap@}\mathcal{O} = \frac{1}{|\mathcal{D}|} \sum_{d \in \mathcal{D}} \left(1 - \frac{\mathrm{Acc}(a,d)}{\mathrm{Acc}(\mathcal{O},d)} \right)$, where $\mathrm{b} \in \{\mathcal{R}, \mathcal{B}\}$. While $\mathrm{Gain@}\mathcal{R}$ and $\mathrm{Gain@}\mathcal{B}$ measure relative performance gains over Random Router and Best Single, respectively, $\mathrm{Gap@}\mathcal{O}$ measures the gap to the Oracle.

Performance-cost metrics In the performance-cost setting, each routing method a typically has a tunable parameter inducing configurations Θ with different performance-cost tradeoffs. For a configuration $\theta \in \Theta$, we denote its total inference cost by $\mathrm{Cost}(\theta)$. We compare each routing method to the Best Single $\mathcal{B}$, focusing on two configurations: (i) $\theta^* = \arg\max_{\theta \in \Theta} \mathrm{AvgAcc}(\theta)$, the configuration with the highest average accuracy (ignoring cost); and (ii) $\theta^\dagger = \arg\min_{\theta \in \Theta} \mathrm{Cost}(\theta)$ with $\mathrm{AvgAcc}(\theta) \geq \mathrm{AvgAcc}(\mathcal{B})$, the least-cost configuration with accuracy no worse than $\mathrm{AvgAcc}(\mathcal{B})$. Based on these, we define two metrics: $\mathrm{PerfGain} = \frac{\mathrm{AvgAcc}(\theta^*)}{\mathrm{AvgAcc}(\mathcal{B})} - 1$, measuring the best achievable performance improvement, and $\mathrm{CostSave} = 1 - \frac{\mathrm{Cost}(\theta^\dagger)}{\mathrm{Cost}(\mathcal{B})}$, measuring the maximal cost reduction without sacrificing performance relative to the Best Single.

In addition, we perform a *Pareto analysis*, a technique for evaluating tradeoffs in multi-objective optimization. We define S to include (i) all routing method configurations and (ii) all single models. We say that $x \in S$ *Pareto-dominates* $y \in S$ if $\mathrm{AvgAcc}(x) \geq \mathrm{AvgAcc}(y)$ and $\mathrm{Cost}(x) \leq \mathrm{Cost}(y)$, and at least one inequality is strict. The *Pareto frontier* $\mathcal{P}$ is the set of configurations in S that are not Pareto-dominated by any other configuration in S. For each routing method, we measure its distance to the Pareto frontier. Let $\tilde{\theta}$ and $\tilde{y}^*$ denote the normalized coordinates and frontier configurations, respectively. For a routing method a with configurations Θ, we define the distance as $\mathrm{ParetoDist} = \frac{1}{|\Theta|} \sum_{\theta \in \Theta} \min_{y^* \in \mathcal{P}} \|\tilde{\theta} - \tilde{y}^*\|_1$. Smaller values of $\mathrm{ParetoDist}$ indicate that the configurations of the method are, on average, closer to the Pareto frontier.

3.5 Overall Benchmark Statistics

Table 3 summarizes key statistics of LLMRouterBench across two routing settings. In the performance-oriented setting, we evaluate 20 lightweight (∼7B) models on 15 challenging datasets, totaling 745.0M tokens. The performance-cost setting assesses 13 flagship models across 10 datasets, totaling 1,030.3M tokens. Taken together, LLMRouterBench comprises 23,945 prompts, 391,645 instances, and approximately 1.8B tokens, requiring substantial data collection efforts (1K GPU hours for lightweight inference and $2,771.84 in API costs). Notably, compared to recent studies, LLMRouterBench contains approximately **17**× more total tokens (1.8B vs. 104M) than FusionFactory (Feng et al., 2025b), and about **3**× more prompts (23,945 vs. 8,400) than RouterArena (Lu et al., 2025).

Metric	Performance-Oriented	Performance-Cost
Datasets	15	10
Models	20	13
Prompts	11,480	12,446
Instances	229,600	161,520
Tokens (M)	745.0	1,030.3
Total cost	A800 1000 GPU hours	$2,771.84†

Table 3: Overall statistics of LLMRouterBench. †About $500 comes from LLM-based judging.

4 Experiments

4.1 Experimental Setup

We evaluate 9 representative routing methods from recently published studies with available open-source implementations, and additionally include

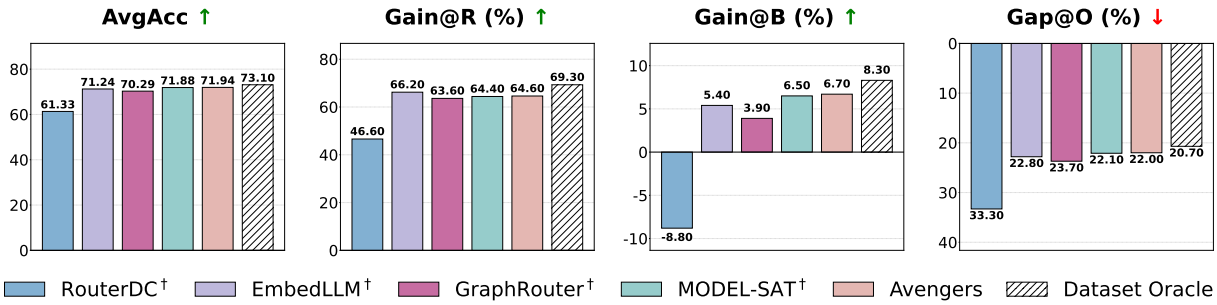

Figure 3: Performance metrics on LLMRouterBench. †Peak score on the test set.

OpenRouter as a commercial router:

- Performance-oriented setting: RouterDC (Chen et al., 2024b), EmbedLLM (Zhuang et al., 2024), MODEL-SAT (Zhang et al., 2025b), GraphRouter (Feng et al., 2025a), and Avengers (Zhang et al., 2025d).
- Performance-cost setting: HybridLLM (Ding et al., 2024), FrugalGPT (Chen et al., 2024a), RouteLLM (Ong et al., 2024), GraphRouter (Feng et al., 2025a), Avengers-Pro (Zhang et al., 2025c), and OpenRouter.

Appendix A.2 and A.3 provide detailed descriptions and implementation details.

Domain	Model	Performance
Math (AIME)	**Intern-S1-mini**	**76.67**
	Qwen3-8B	76.67
	NVIDIA-Nemo	70.00
Code (HumanEval)	**Fin-R1**	**77.44**
	Qwen-Coder	77.44
	Glm-4-chat	74.39
Logical (BBH)	**DS-Qwen3**	**89.44**
	MiniCPM	88.06
	NVIDIA-Nemo	86.48
Affective (EmoryNLP)	**Gemma-2-it**	**39.74**
	NVIDIA-Nemo	39.02
	Glm-4-chat	39.02

Table 4: Performance of the top-3 models across domains. No single model dominates all domains.

4.2 Results and Analyses

Due to the lack of space, we present detailed benchmark results across all evaluated models and datasets in Appendix Tables 7, 8, and 9. The reported results are averaged across runs; Appendix Tables 13, 14, and 15 additionally report per-dataset results averaged over five runs with different random seeds. In the following, we highlight and discuss the key findings.

4.2.1 Performance-Oriented Setting

No single model rules every domain; models exhibit complementary strengths. A central premise of performance-oriented routing is that different models excel in various domains. As shown in Table 4, we find clear evidence of this complementarity: mathematics benchmarks are often led by models such as Intern-S1-mini or Qwen3-8B, code benchmarks by Qwen-Coder or Fin-R1, and logical and affective benchmarks by other models. This confirms the central premise of this field.

Top routing methods are comparable, but can be free of neural network training. LLM routing is not new, resulting in a substantial body of literature. Nonetheless, our results indicate that, despite continued methodological innovation, many routing approaches achieve broadly comparable performance in practice. As shown in Fig. 3, across multiple metrics (Gain@$\mathcal{R}$, AvgAcc, Gain@$\mathcal{B}$, and Gap@$\mathcal{O}$), leading routing methods (EmbedLLM, GraphRouter, MODEL-SAT, and Avengers) yield similar outcomes. Note that the Avengers achieve such performance primarily through clustering and do not require neural-network training.

This observation has two implications. First, it is favorable from a deployment standpoint: lightweight routers, which are inexpensive to develop and straightforward to update, may be sufficient in practice. On the other hand, the lack of differentiation among leading methods suggests that a large fraction of routing gains may be attributable to capturing coarse-grained domain structure (e.g., distinguishing mathematics and code) rather than learning highly nuanced decision boundaries. This is supported by the proximity of these methods to the Dataset Oracle (the hatched bar in Fig. 3), which assigns each dataset to the single model with the highest accuracy on that dataset.

Routing remains far from its capability ceiling: current methods often miss the lone correct model. Although top routing methods perform similarly, which may raise the question of whether further gains are available, the gap to the Oracle, shown in Fig. 3 (d), reveals a significant gap to the routing upper bound given by the Oracle baseline. A key contributor is model-recall failure: when only one or a few candidate models produce the correct answer, current routers often fail to select them. As analyzed in Fig. 4, this issue accounts for a non-trivial portion of the remaining error. For example, on queries where at most three experts answer correctly (410 queries, 11.9% of the test set), Avengers and EmbedLLM achieve low accuracy (24.6% and 23.2%, respectively).

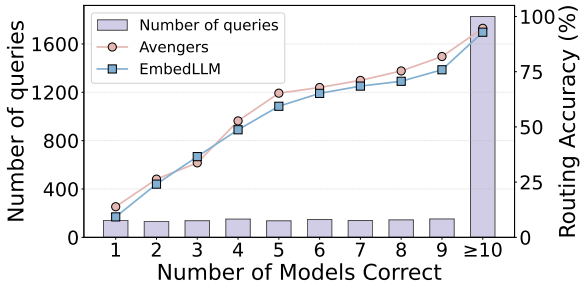

Figure 4: Query hardness distribution (by number of correct models) with router accuracy.

One might ask whether the correct model is narrowly missed—ranked just below the top choice—or missed entirely. To test this, we measure Recall@k, i.e., whether any correct model appears in the router's top-k candidates. As shown in Table 5, even with k=3, Recall remains limited for both Avengers (50.6%) and EmbedLLM (46.1%), meaning the correct model is not ranked among the top candidates.

Method	Recall@1	Recall@3	Recall@5
EmbedLLM	23.2	46.1	60.8
Avengers	24.6	50.6	64.81

Table 5: Recall@k on queries where at most three candidate models answer correctly.

Closing this gap will likely require routers that more reliably detect and prioritize these cases, for example via improved uncertainty or difficulty estimation, or explicit mechanisms to boost recall of rare-but-critical experts, which constitutes a promising direction for future work.

Embedding models have little influence on routing performance. Because many routing methods rely on an embedding model, we examine how this choice affects performance. Surprisingly, the embedding models have consistently little differences across routing methods. Table 6 shows that by replacing *gte-qwen2-7B-instruct* with *nli-bert-base* (Reimers and Gurevych, 2019) and *all-MiniLM-L6-v2* (Wang et al., 2020), we do not observe a significant difference in performance among GraphRouter, EmbedLLM, and Avengers, all of which depend on embeddings. Note that both alternatives are weaker backbones: *all-MiniLM-L6-v2* involves only 22.7M parameters, and *nli-bert-base* is deprecated in Sentence-Transformers for poor sentence-embedding quality.

Embedding Model	GraphRouter	EmbedLLM	Avengers
nli-bert-base	69.60	70.55	70.43
all-MiniLM-L6-v2	68.05	70.95	71.03
gte-Qwen2-7B-instruct	70.29	71.24	71.94

Table 6: Performance comparison of GraphRouter, EmbedLLM, and Avengers on different embedding models.

This suggests that embedding quality may not be the primary bottleneck for current embedding-based routers. Future gains may come less from improving semantic representations per se, but more from developing routing mechanisms that better translate representations into reliable model selection, particularly under distribution shift and in rare cases where correct performance hinges on selecting a specific specialist model.

Adding more models yields diminishing returns, but a well-chosen subset can make a difference. A common hypothesis in the field (Jiang et al., 2023) is that expanding the candidate pool should increase complementarity and therefore improve collective performance. However, using the Oracle baseline, we find clear diminishing returns as more models are added (Fig. 5). The largest gains occur when moving from a very small pool to a moderate one, after which additional models contribute only marginal improvements. By contrast, how we select a small subset matters substantially: comparing random selection to choosing the top-k models by average accuracy, the latter consistently delivers stronger performance.

This suggests that careful curation can outweigh simply scaling the pool; a carefully selected moderate pool plus a robust router may offer most of the

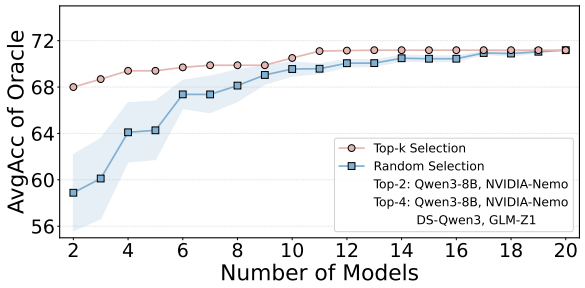

Figure 5: Comparison of Oracle performance under selected versus random subsets. Adding more models yields diminishing returns, but a well-chosen small subset can outperform a larger random pool.

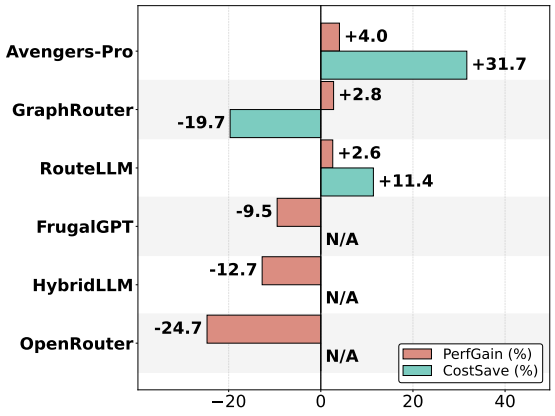

Figure 6: Performance gains and cost savings of various routing methods relative to the GPT-5. Cost savings are reported only for methods achieving accuracy equal to or higher than GPT-5; otherwise marked as N/A.

achievable benefit without the overhead of maintaining a very large pool. Future work should study model pool curation jointly with routing: selecting a small set that maximizes complementarity, potentially via coverage or diversity-style selection.

4.2.2 Performance-Cost Settings

Models complement each other on performance and cost-efficiency. Similar to the performance-oriented setting, we observe complementarity among models in terms of performance and cost. As Appendix Table 8 shows, while GPT-5 achieves the best average accuracy and leads on HLE, substantially cheaper models such as Qwen3-235B and DeepSeek-R1 can match proprietary-level accuracy on selected mathematics and QA tasks. These results support the central premise behind routing for performance-cost tradeoffs.

Effective routing improves upon the Best Single and reduces cost without sacrificing performance, but not all routers succeed. Leveraging model complementarity, as shown in Fig. 6, top routing methods achieve up to a 4% average accuracy gain over the Best Single model and up to a 31.7% cost reduction while matching Best Single performance. However, these gains are not universal: several routers fail to outperform the Best Single, and some (especially binary routers, such as HybridLLM and FrugalGPT) struggle to trade cost for savings while preserving Best Single accuracy. Notably, the commercial router OpenRouter yields the smallest (indeed, negative) performance improvement (-24.7%) relative to the Best Single model and fails to match its performance, despite operating over a much larger candidate pool.[†]

[†]OpenRouter uses a platform-defined model pool that differs from ours and is not user-configurable (OpenRouter, Inc., 2025a); we provide details for their pool in Appendix Table 16.

These results reinforce the appropriateness of Best Single as a baseline: a non-trivial fraction of routing strategies do not outperform this simple alternative. More broadly, the variability across routers indicates substantial remaining headroom for robust performance-cost routing, particularly for methods that can deliver cost reductions without sacrificing accuracy.

Avengers-Pro achieves a Pareto-optimal performance-cost tradeoff. We visualize the Pareto frontier, a stylish analysis for multi-objective optimization in Fig. 7. Unlike the performance-oriented setting (where several routing methods are competitive), Avengers-Pro nearly dominates the frontier. That is, relative to any single model or other routing methods, Avengers-Pro is almost always cheaper at comparable performance or achieves higher performance at comparable cost. No other methods or single models can be simultaneously cheaper and more accurate than Avengers-Pro. This is further supported by the ParetoDist metric in Fig. 7; Avengers-Pro attains ParetoDist near zero, generally being Pareto-optimal, whereas other routers exhibit substantially larger distances.

These results have two implications. First, they demonstrate that strong performance-cost routing is achievable in practice: the Pareto frontier is not merely a theoretical construct, and a well-designed router can operate near it. Second, they indicate that future progress should be assessed not only by average accuracy, but also by whether new methods shift the frontier—that is, expand the set of attainable operating points toward simultaneously

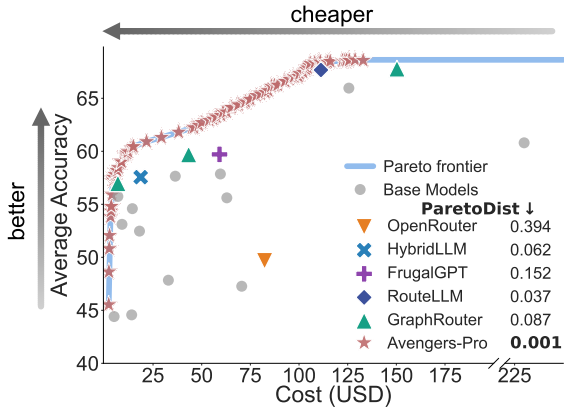

Figure 7: Average accuracy versus total inference cost for all base models and routing methods, with the empirical Pareto frontier highlighted.

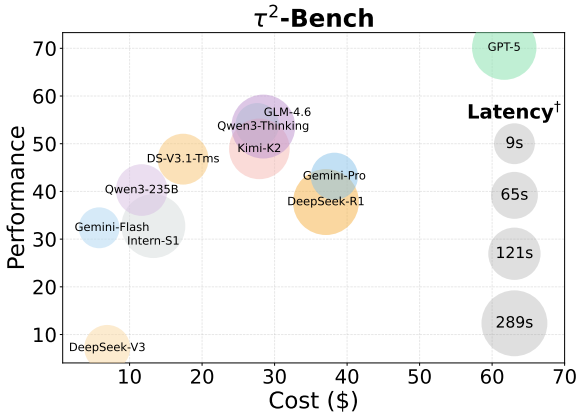

Figure 8: Model performance-cost-latency tradeoffs on τ^2-Bench. † Latency estimates were derived from token usage together with OpenRouter serving statistics (Jan. 5, 2026).

lower cost and higher performance.

LLMRouterBench enables extending routing to performance-cost-latency optimization. We note that LLMRouterBench also enables latency-aware analysis. By tracking the number of consumed tokens (supported by our LLMRouterBench) and combining it with serving statistics reported by OpenRouter (e.g., time-to-first-token and tokens-per-second), we estimate response latency for different models under a unified protocol. Although approximate, these estimates provide a practical basis for latency-aware comparison across models. To date, however, routing research has largely focused on optimizing accuracy and/or cost, and, to our knowledge, we do not notice any methods that explicitly target the joint performance-cost-latency tradeoffs.

In Fig. 8, we illustrate how different models vary along these three dimensions. Models that are similar in accuracy and cost can nonetheless differ markedly in latency. For example, Qwen3-Thinking and GLM-4.6 have similar performance and cost but differ notably in latency (262.1s vs. 32.4s). This introduces an additional axis of complementarity that is directly relevant to user experience. From a practical standpoint, when two models deliver comparable accuracy at similar cost, users will often prefer the model that responds faster. Accordingly, our dataset provides the necessary signals to support systematic exploration of this tri-objective setting.

5 Conclusions

We present LLMRouterBench, a large-scale benchmark and unified framework for LLM routing under both performance-oriented and performance-cost trade-off settings. By consolidating over 400K instances spanning 21 datasets and 33 models, and by introducing comprehensive metrics together with 10 representative baselines, LLMRouterBench provides a solid foundation for systematic study in this rapidly evolving area. Our results reaffirm the central motivation of LLM routing—strong complementarity across models—but also challenge several prevailing claims in the literature. Under a unified evaluation, most routing methods collapse to similar performance, and multiple recent approaches, including widely deployed commercial routers, fail to reliably outperform a simple baseline. A large and persistent gap to the Oracle remains, driven primarily by systematic model-recall failures rather than insufficient ensemble capacity. We further demonstrate that common design choices, such as the selection of backbone embedding models or aggressive scaling of ensemble size, yield limited gains in practice. Additionally, we show that our benchmark enables extending routing to tradeoffs for latency.

Acknowledgements

This research was supported by the National Key Research and Development Project of China (No. 2024YFE0210900), the National Natural Science Foundation of China (No. U22B2036, No. 62506186), the Technological Innovation Team of Shaanxi Province (No. 2025RS-CXTD009), and the International Cooperation Project of Shaanxi Province (No. 2025GH-YBXM-017), and the Shanghai Artificial Intelligence Laboratory.

Limitations

Our work has three limitations. First, while we evaluate a broad set of routing methods, we do not cover all existing approaches. Given the large and growing number of routing methods, we focus on recent approaches with publicly available implementations. LLMRouterBench is modular by design, allowing additional routers to be integrated via lightweight adapters without reimplementing the full pipeline.

Second, the benchmark is built from 21 widely used datasets that reflect common evaluation regimes for contemporary lightweight and flagship LLMs. Other settings, such as domain-specific verticals, very long-context tasks, and multimodal benchmarks, are not included. The routing formulations, metrics, and analysis procedures used in LLMRouterBench are generic and can be applied to these settings by adding new dataset evaluators.

Third, the latency analysis is approximate. We estimate latency using token-level usage statistics together with throughput figures reported by OpenRouter. These estimates correspond to a specific provider configuration and do not capture deployment-level effects such as network conditions, batching, caching, or router-side overhead, and should be interpreted as indicative rather than definitive.

References

Anthropic. 2025. System card: Claude opus 4 and claude sonnet 4. https://www.anthropic.com/news/claude-4.

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

Lei Bai, Zhongrui Cai, Yuhang Cao, Maosong Cao, Weihan Cao, Chiyu Chen, Haojiong Chen, Kai Chen, Pengcheng Chen, Ying Chen, and 1 others. 2025. Intern-s1: A scientific multimodal foundation model. *arXiv preprint arXiv:2508.15763*.

Victor Barres, Honghua Dong, Soham Ray, Xujie Si, and Karthik Narasimhan. 2025. τ^2-Bench: Evaluating conversational agents in a dual-control environment. *arXiv preprint arXiv:2506.07982*.

Aarti Basant, Abhijit Khairnar, Abhijit Paithankar, Abhinav Khattar, Adithya Renduchintala, Aditya Malte, Akhiad Bercovich, Akshay Hazare, Alejandra Rico, Aleksander Ficek, and 1 others. 2025. Nvidia nemotron nano 2: An accurate and efficient hybrid mamba-transformer reasoning model. *arXiv preprint arXiv:2508.14444*.

Akhiad Bercovich, Itay Levy, Izik Golan, Mohammad Dabbah, Ran El-Yaniv, Omri Puny, Ido Galil, Zach Moshe, Tomer Ronen, Najeeb Nabwani, Ido Shahaf, Oren Tropp, Ehud Karpas, Ran Zilberstein, Jiaqi Zeng, Soumye Singhal, Alexander Bukharin, Yian Zhang, Tugrul Konuk, and 114 others. 2025. Llama-nemotron: Efficient reasoning models. *Preprint*, arXiv:2505.00949.

Mats Byrkjeland, Frederik Gørvell de Lichtenberg, and Björn Gambäck. 2018. Ternary twitter sentiment classification with distant supervision and sentiment-specific word embeddings. In *Proceedings of the 9th workshop on computational approaches to subjectivity, sentiment and social media analysis*, pages 97–106.

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

Lingjiao Chen, Matei Zaharia, and James Zou. 2024a. FrugalGPT: How to use large language models while reducing cost and improving performance. *Transactions on Machine Learning Research*.

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, and 39 others. 2021a. Evaluating large language models trained on code. *Preprint*, arXiv:2107.03374.

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

Zhiyu Chen, Wenhu Chen, Charese Smiley, Sameena Shah, Iana Borova, Dylan Langdon, Reema Moussa, Matt Beane, Ting-Hao Huang, Bryan Routledge, and William Yang Wang. 2021b. Finqa: A dataset of numerical reasoning over financial data. *Proceedings of EMNLP 2021*.

Deep Cogito. 2025. Cogito v1 preview - llama 8b. https://huggingface.co/deepcogito/cogito-v1-preview-llama-8B. Hugging Face model card.

Dujian Ding, Ankur Mallick, Chi Wang, Robert Sim, Subhabrata Mukherjee, Victor Ruhle, Laks VS Lakshmanan, and Ahmed Hassan Awadallah. 2024. Hybrid llm: Cost-efficient and quality-aware query routing. *arXiv preprint arXiv:2404.14618*.

Dujian Ding, Ankur Mallick, Shaokun Zhang, Chi Wang, Daniel Madrigal, Mirian Del Carmen Hipolito Garcia, Menglin Xia, Laks V. S. Lakshmanan, Qingyun Wu, and Victor Rühle. 2025. BEST-route: Adaptive LLM routing with test-time optimal compute. In *Forty-second International Conference on Machine Learning*.

Jacob Fein-Ashley, Dhruv Parikh, Rajgopal Kannan, and Viktor Prasanna. 2025. Mixture of thoughts: Learning to aggregate what experts think, not just what they say. *arXiv preprint arXiv:2509.21164*.

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

Tao Feng, Haozhen Zhang, Zijie Lei, Pengrui Han, Mostofa Patwary, Mohammad Shoeybi, Bryan Catanzaro, and Jiaxuan You. 2025b. Fusionfactory: Fusing llm capabilities with multi-llm log data. *arXiv preprint arXiv:2507.10540*.

Nigel Fernandez, Branislav Kveton, Ryan A Rossi, Andrew S Lan, and Zichao Wang. 2025. Radar: Reasoning-ability and difficulty-aware routing for reasoning llms. *arXiv preprint arXiv:2509.25426*.

Gemini Team. 2025. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*.

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

Granite Team, IBM. 2025. Granite-3.3-8b-instruct. https://huggingface.co/ibm-granite/granite-3.3-8b-instruct. Hugging Face model card, release date: 2025-04-16.

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

Etash Guha, Ryan Marten, Sedrick Keh, Negin Raoof, Georgios Smyrnis, Hritik Bansal, Marianna Nezhurina, Jean Mercat, Trung Vu, Zayne Sprague, Ashima Suvarna, Benjamin Feuer, Liangyu Chen, Zaid Khan, Eric Frankel, Sachin Grover, Caroline Choi, Niklas Muennighoff, Shiye Su, and 31 others. 2025. Openthoughts: Data recipes for reasoning models. *Preprint*, arXiv:2506.04178.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025a. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Xiyu Guo, Shan Wang, Chunfang Ji, Xuefeng Zhao, Wenhao Xi, Yaoyao Liu, Qinglan Li, Chao Deng, and Junlan Feng. 2025b. Towards generalized routing: Model and agent orchestration for adaptive and efficient inference. *arXiv preprint arXiv:2509.07571*.

Qitian Jason Hu, Jacob Bieker, Xiuyu Li, Nan Jiang, Benjamin Keigwin, Gaurav Ranganath, Kurt Keutzer, and Shriyash Kaustubh Upadhyay. 2024. Routerbench: A benchmark for multi-llm routing system. *arXiv preprint arXiv:2403.12031*.

Canbin Huang, Tianyuan Shi, Yuhua Zhu, Ruijun Chen, and Xiaojun Quan. 2025a. Lookahead routing for large language models. *arXiv preprint arXiv:2510.19506*.

Zhongzhan Huang, Guoming Ling, Yupei Lin, Yandong Chen, Shanshan Zhong, Hefeng Wu, and Liang Lin. 2025b. Routereval: A comprehensive benchmark for routing llms to explore model-level scaling up in llms. *arXiv preprint arXiv:2503.10657*.

Hugging Face. 2025. HuggingChat Omni. https://huggingface.co/chat/settings/omni. [Accessed: 2026-1-06].

Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Keming Lu, Kai Dang, Yang Fan, Yichang Zhang, An Yang, Rui Men, Fei Huang, Bo Zheng, Yibo Miao, Shanghaoran Quan, and 5 others. 2024. Qwen2.5-coder technical report. *Preprint*, arXiv:2409.12186.

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

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

Carlos E Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. 2023. Swe-bench: Can language models resolve real-world github issues? *arXiv preprint arXiv:2310.06770*.

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

Ruihan Jin, Pengpeng Shao, Zhengqi Wen, Jinyang Wu, Mingkuan Feng, Shuai Zhang, and Jianhua Tao. 2025. RadialRouter: Structured representation for efficient and robust large language models routing. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 14587–14600, Suzhou, China. Association for Computational Linguistics.

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

Tianle Li, Wei-Lin Chiang, Evan Frick, Lisa Dunlap, Tianhao Wu, Banghua Zhu, Joseph E Gonzalez, and Ion Stoica. 2024. From crowdsourced data to high-quality benchmarks: Arena-hard and benchbuilder pipeline. *arXiv preprint arXiv:2406.11939*.

Zehan Li, Xin Zhang, Yanzhao Zhang, Dingkun Long, Pengjun Xie, and Meishan Zhang. 2023. Towards general text embeddings with multi-stage contrastive learning. *arXiv preprint arXiv:2308.03281*.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. *arXiv preprint arXiv:2305.20050*.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, and 1 others. 2024a. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Hongwei Liu, Zilong Zheng, Yuxuan Qiao, Haodong Duan, Zhiwei Fei, Fengzhe Zhou, Wenwei Zhang, Songyang Zhang, Dahua Lin, and Kai Chen. 2024b. Mathbench: Evaluating the theory and application proficiency of llms with a hierarchical mathematics benchmark. *arXiv preprint arXiv:2405.12209*.

Junnan Liu, Hongwei Liu, Linchen Xiao, Ziyi Wang, Kuikun Liu, Songyang Gao, Wenwei Zhang, Songyang Zhang, and Kai Chen. 2024c. Are your llms capable of stable reasoning? *arXiv preprint arXiv:2412.13147*.

Zhaowei Liu, Xin Guo, Fangqi Lou, Lingfeng Zeng, Jinyi Niu, Zixuan Wang, Jiajie Xu, Weige Cai, Ziwei Yang, Xueqian Zhao, Chao Li, Sheng Xu, Dezhi Chen, Yun Chen, Zuo Bai, and Liwen Zhang. 2025. Fin-r1: A large language model for financial reasoning through reinforcement learning. *Preprint*, arXiv:2503.16252.

Keming Lu, Hongyi Yuan, Runji Lin, Junyang Lin, Zheng Yuan, Chang Zhou, and Jingren Zhou. 2024. Routing to the expert: Efficient reward-guided ensemble of large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 1964–1974, Mexico City, Mexico. Association for Computational Linguistics.

Yifan Lu, Rixin Liu, Jiayi Yuan, Xingqi Cui, Shenrun Zhang, Hongyi Liu, and Jiarong Xing. 2025. Routerarena: An open platform for comprehensive comparison of llm routers. *arXiv preprint arXiv:2510.00202*.

Kaijing Ma, Xinrun Du, Yunran Wang, Haoran Zhang, Zhoufutu Wen, Xingwei Qu, Jian Yang, Jiaheng Liu, Minghao Liu, Xiang Yue, Wenhao Huang, and Ge Zhang. 2024. Kor-bench: Benchmarking language models on knowledge-orthogonal reasoning tasks. *Preprint*, arXiv:2410.06526.

Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E Gonzalez, M Waleed Kadous, and Ion Stoica. 2024. Routellm: Learning to route llms with preference data. *arXiv preprint arXiv:2406.18665*.

OpenAI. 2025a. GPT-5. https://openai.com/gpt-5/. [Accessed: 2026-1-06].

OpenAI. 2025b. Gpt-5 system card. https://cdn.openai.com/gpt-5-system-card.pdf.

OpenRouter, Inc. 2025a. Auto router - api, providers, stats. https://openrouter.ai/openrouter/auto.

OpenRouter, Inc. 2025b. Openrouter: A unified interface for large language models. https://openrouter.ai. Accessed: 2025-11-13.

Shivam Patel, Neharika Jali, Ankur Mallick, and Gauri Joshi. 2025. Proxrouter: Proximity-weighted llm query routing for improved robustness to outliers. *arXiv preprint arXiv:2510.09852*.

Long Phan, Alice Gatti, Ziwen Han, Nathaniel Li, Josephina Hu, Hugh Zhang, Chen Bo Calvin Zhang, Mohamed Shaaban, John Ling, Sean Shi, and 1 others. 2025. Humanity's last exam. *arXiv preprint arXiv:2501.14249*.

Soujanya Poria, Devamanyu Hazarika, Navonil Majumder, Gautam Naik, Erik Cambria, and Rada Mihalcea. 2019. Meld: A multimodal multi-party dataset for emotion recognition in conversations. *Preprint*, arXiv:1810.02508.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

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

Wei Song, Zhenya Huang, Cheng Cheng, Weibo Gao, Bihan Xu, GuanHao Zhao, Fei Wang, and Runze Wu. 2025. IRT-router: Effective and interpretable multi-LLM routing via item response theory. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15629–15644, Vienna, Austria. Association for Computational Linguistics.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V Le, Ed H Chi, Denny Zhou, , and Jason Wei. 2022. Challenging big-bench tasks and whether chain-of-thought can solve them. *arXiv preprint arXiv:2210.09261*.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, and 1 others. 2024. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*.

Kimi Team, Yifan Bai, Yiping Bao, Guanduo Chen, Jiahao Chen, Ningxin Chen, Ruijue Chen, Yanru Chen, Yuankun Chen, Yutian Chen, and 1 others. 2025a. Kimi k2: Open agentic intelligence. *arXiv preprint arXiv:2507.20534*.

MiniCPM Team, Chaojun Xiao, Yuxuan Li, Xu Han, Yuzhuo Bai, Jie Cai, Haotian Chen, Wentong Chen, Xin Cong, Ganqu Cui, and 1 others. 2025b. Minicpm4: Ultra-efficient llms on end devices. *arXiv preprint arXiv:2506.07900*.

Ryan Teknium, Jeffrey Quesnelle, and Chen Guang. 2024. Hermes 3 technical report. *Preprint*, arXiv:2408.11857.

Chenxu Wang, Hao Li, Yiqun Zhang, Linyao Chen, Jianhao Chen, Ping Jian, Peng Ye, Qiaosheng Zhang, and Shuyue Hu. 2025a. Icl-router: In-context learned model representations for llm routing. *arXiv preprint arXiv:2510.09719*.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *Advances in neural information processing systems*, 33:5776–5788.

Xinyuan Wang, Yanchi Liu, Wei Cheng, Xujiang Zhao, Zhengzhang Chen, Wenchao Yu, Yanjie Fu, and Haifeng Chen. 2025b. Mixllm: Dynamic routing in mixed large language models. *arXiv preprint arXiv:2502.18482*.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, and 1 others. 2024. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *arXiv preprint arXiv:2406.01574*.

Jason Wei, Nguyen Karina, Hyung Won Chung, Yunxin Joy Jiao, Spencer Papay, Amelia Glaese, John Schulman, and William Fedus. 2024. Measuring short-form factuality in large language models. *arXiv preprint arXiv:2411.04368*.

LLM-Core-Team Xiaomi. 2025. Mimo: Unlocking the reasoning potential of language model – from pretraining to posttraining. *Preprint*, arXiv:2505.07608.

Chulin Xie, Yangsibo Huang, Chiyuan Zhang, Da Yu, Xinyun Chen, Bill Yuchen Lin, Bo Li, Badih Ghazi, and Ravi Kumar. 2024. On memorization of large language models in logical reasoning. *arXiv preprint arXiv:2410.23123*.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Yanwei Yue, Guibin Zhang, Boyang Liu, Guancheng Wan, Kun Wang, Dawei Cheng, and Yiyan Qi. 2025. Masrouter: Learning to route llms for multi-agent systems. *arXiv preprint arXiv:2502.11133*.

Aohan Zeng, Xin Lv, Qinkai Zheng, Zhenyu Hou, Bin Chen, Chengxing Xie, Cunxiang Wang, Da Yin, Hao Zeng, Jiajie Zhang, and 1 others. 2025. Glm-4.5: Agentic, reasoning, and coding (arc) foundation models. *arXiv preprint arXiv:2508.06471*.

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

Kaiyan Zhang, Sihang Zeng, Ermo Hua, Ning Ding, Zhang-Ren Chen, Zhiyuan Ma, Haoxin Li, Ganqu Cui, Biqing Qi, Xuekai Zhu, and 1 others. 2024. Ultramedical: Building specialized generalists in biomedicine. *Advances in Neural Information Processing Systems*, 37:26045–26081.

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

Yiqun Zhang, Hao Li, Jianhao Chen, Hangfan Zhang, Peng Ye, Lei Bai, and Shuyue Hu. 2025c. Beyond gpt-5: Making llms cheaper and better via performance-efficiency optimized routing. *arXiv preprint arXiv:2508.12631*.

Yiqun Zhang, Hao Li, Chenxu Wang, Linyao Chen, Qiaosheng Zhang, Peng Ye, Shi Feng, Daling Wang, Zhen Wang, Xinrun Wang, and 1 others. 2025d. The avengers: A simple recipe for uniting smaller language models to challenge proprietary giants. *arXiv preprint arXiv:2505.19797*.

Richard Zhuang, Tianhao Wu, Zhaojin Wen, Andrew Li, Jiantao Jiao, and Kannan Ramchandran. 2024. Embedllm: Learning compact representations of large language models. *arXiv preprint arXiv:2410.02223*.

A Implementation Details

A.1 Data Collection

All ∼7B open-source models are deployed on NVIDIA A800-80G GPUs using vLLM 0.8.4 for efficient batched inference. Flagship model outputs are collected via OpenRouter, with the exception of GLM-4.6 and Intern-S1, which are accessed through official APIs. All model generations use temperature 0.2 and top_p 1.0, with remaining decoding parameters set to their default values. Each API request is retried up to 10 times upon failure; requests exceeding this limit are marked as failures and assigned a score of 0.

A.2 Experimental Setup

To ensure fair and consistent comparisons across baselines, we standardize the experimental setup as follows: (i) All experiments are conducted using a 70% training and 30% test split, repeated five times with different random seeds (42, 999, 2024, 2025, and 3407). (ii) Embedding-dependent methods uniformly utilize *gte-qwen2-7B-instruct* (Li et al., 2023) embeddings. (iii) For binary routers (RouteLLM, HybridLLM, FrugalGPT), we route between Qwen3-235B and GPT-5, as GPT-5 is the strongest model in our pool while Qwen3-235B offers competitive accuracy at much lower cost. All datasets and models used in this paper are publicly available and properly cited. Our usage complies with their original licenses and intended research purposes.

A.3 Baselines

RouterDC We adopt the official implementation of RouterDC (Chen et al., 2024b), replacing the original encoder with `gte-qwen2-7B-instruct` to enable fair comparison under a unified embedding backbone. Training is conducted using DeepSpeed with distributed multi-GPU parallelism across eight NVIDIA A800-80G GPUs, with a per-device batch size of 8. All other hyperparameters remain consistent with the original configuration. The resulting model contains approximately 7B trainable parameters.

MODEL-SAT Due to the incomplete release of the official implementation (Zhang et al., 2025b), we re-implement the core components of MODEL-SAT. We use `gte-qwen2-7B-instruct` as the embedding model and `Qwen2.5-7B-Instruct` as the language model, connected via a two-layer MLP projector. Training is performed using DeepSpeed with data parallelism over eight NVIDIA A800-80G GPUs, using a per-device batch size of 4. The learning rate is set to 1e-6 for both the embedding and language models, and 1e-5 for the projector. We first fine-tune only the projector for approximately 1,000 steps, then jointly fine-tune all components. A warmup ratio of 0.1 is used throughout. The final model contains roughly 14B trainable parameters.

EmbedLLM We use the official implementation of EmbedLLM (Zhuang et al., 2024), with query embeddings generated by `gte-qwen2-7B-instruct` to ensure consistency across methods. Input layer dimensions are adjusted accordingly. Training is conducted with an increased batch size of 32,768 for improved stability, while all other hyperparameters follow the original setting. The model has approximately 12M trainable parameters.

HybridLLM We adopt the official implementation of HybridLLM (Ding et al., 2024), substituting the original encoder with `gte-qwen2-7B-instruct` for consistency with other baselines. Training is performed using DeepSpeed with a distributed multi-GPU setup across eight NVIDIA A800-80G GPUs, with a batch size of 8 per device. All remaining hyperparameters follow the original configuration. We use `GPT-5` as the strong model and `Qwen3-235B` as the weak model throughout.

RouteLLM We adopt the official implementation of RouteLLM (Ong et al., 2024), using the *Matrix Factorization* (MF) router and `gte-qwen2-7B-instruct` to generate query embeddings. Throughout training and evaluation, we consistently use `GPT-5` as the strong model and `Qwen3-235B` as the weak model. Following the default setting in the official code, we set the win-rate threshold to **0.5**—that is, when the estimated win rate exceeds this threshold, we select the strong model; otherwise, we use the weak model. Under our configuration (`dim` = 128, projected from 3584-dimensional `gte-qwen2-7B-instruct` embeddings), the MF router itself is extremely lightweight, with only $\sim 4.6 \times 10^5$ trainable parameters, and is trained for 100 epochs, making the routing overhead negligible compared to querying the underlying LLMs.

FrugalGPT We follow the official FrugalGPT training pipeline and fine-tune an embedding-based scorer derived from `gte-Qwen2-7B-Instruct`. Training is conducted for two epochs using AdamW with a linear warm-up schedule. We set the learning rate to 3×10^{-5}, apply a weight decay of 0.01, fix the warm-up ratio at 0.03, and use a per-device batch size of 4. For routing, we adopt the FrugalGPT cascade strategy. Models are ranked by their average cost on the training set, and cascade evaluation is enabled with a default threshold of 0.5. Model-specific decision thresholds are learned automatically, and the cascade depth is capped at 2 models per query to balance cost and performance. We use `GPT-5` as the strong model and `Qwen3-235B` as the weak model throughout.

GraphRouter We use the official implementation of GraphRouter (Feng et al., 2025a). All query, task, and model description embeddings are generated using `gte-qwen2-7B-instruct`. Input layer dimensions are adjusted accordingly. To mitigate label bias caused by tied predictions, we replace argmax-based one-hot labels with multi-hot supervision that includes all tie-optimal models per query. For improved training stability, we increase the number of training epochs to 10,000. The model has approximately 0.1M trainable parameters. We follow the original paper in employing its three configurations: *Performance First (PF)*, *Balance (BL)*, and *Cost First (CF)*.

Avengers(-Pro) We adopt the official implementation of the clustering-based method proposed by Zhang et al. (2025c). Query embeddings are generated using `gte-qwen2-7B-instruct`, and k-means clustering is applied with $k = 64$. This method involves no neural network training. For Avengers-Pro, following the original paper, we vary the performance coefficient (i.e., $1-$cost coefficient) from 0 to 1 in increments of 0.01, resulting in 101 configurations.

OpenRouter We use the official API provided by OpenRouter with the `openrouter/auto` model, which is currently an advanced commercial LLM router available. All model generations use temperature 0.2 and top_p 1.0, with remaining decoding parameters set to their default values. Each API request is retried up to 10 times upon failure; requests exceeding this limit are marked as failures and assigned a score of 0. The detailed model pool supported by OpenRouter is summarized in Table 16. Note that GPT-5 and Gemini-2.5-pro, the strongest models used in LLMRouterBench, are included, and OpenRouter's model pool is even larger. We therefore treat OpenRouter as a representative commercial router baseline.

Model	Mathematics			Code			Logical			Knowledge				Affective	
	AIME	M500.	MBen.	MBPP	HE.	LCB.	KOR.	K&K.	BBH	MP.	GPQA	FQA.	MQA.	Emory.	MELD
DH-Llama3-it	0.00	29.80	38.67	54.00	51.22	13.08	31.92	15.57	49.07	36.66	27.78	49.96	57.58	38.16	47.16
DS-Qwen3	68.33	94.20	90.00	35.01	25.00	64.74	56.00	60.86	89.44	70.83	60.61	70.36	81.62	38.74	51.06
DS-Qwen	40.00	88.40	88.00	53.59	45.12	41.14	45.36	50.86	69.17	47.75	35.86	63.30	34.80	27.55	34.90
Fin-R1	11.67	75.40	66.67	68.99	77.44	6.82	33.20	19.00	61.57	48.65	27.27	68.70	63.24	34.72	50.41
GLM-Z1	61.67	94.60	95.33	62.94	60.37	61.52	53.84	47.57	84.26	67.33	56.06	68.70	71.01	35.01	45.86
Intern-S1-mini	76.67	91.40	64.67	44.05	43.29	26.26	58.16	79.43	83.06	56.04	48.99	70.36	68.42	36.87	50.41
Llama-3.1-it	8.33	49.80	46.67	60.68	61.59	16.78	24.16	13.71	58.70	46.65	26.26	52.92	68.26	33.72	48.05
UltraMedical	0.00	42.40	25.33	53.90	56.10	13.36	14.32	15.43	38.06	40.26	23.23	54.58	68.50	31.71	42.21
Llama-Nemo	45.00	90.80	50.00	61.50	62.80	46.07	29.52	22.86	26.20	41.46	33.84	31.12	41.48	29.41	38.80
MiMo-RL	0.00	26.60	11.33	56.78	52.44	6.45	29.84	4.00	29.91	20.78	4.04	17.00	29.38	28.12	39.69
MiniCPM	68.33	93.40	95.33	34.09	32.93	63.70	30.96	62.71	88.06	69.93	52.02	67.22	65.36	35.72	51.30
NVIDIA-Nemo	70.00	93.60	94.67	42.09	48.17	65.50	56.80	70.71	86.48	70.13	52.02	71.23	74.71	39.02	46.75
OpenThinker	41.67	85.60	62.67	11.81	7.32	44.17	11.04	17.29	42.50	42.06	31.82	61.55	52.08	22.81	33.44
Qwen-Coder	1.67	65.20	67.33	75.98	77.44	27.87	34.24	20.43	57.04	45.35	32.32	63.21	48.78	37.02	52.60
Qwen3-8B	76.67	93.60	95.33	54.72	62.20	67.68	53.92	77.00	83.98	69.13	57.58	74.46	79.18	38.74	54.22
Cogito-v1	1.67	51.20	56.00	51.95	69.51	17.25	42.96	24.14	69.44	57.04	35.35	61.64	65.75	37.45	54.95
Gemma-2-it	1.67	48.20	54.67	62.42	64.63	17.25	34.64	9.71	60.74	52.95	27.78	64.60	63.79	39.74	52.60
Glm-4-chat	1.67	49.80	49.33	62.83	74.39	16.87	37.28	12.43	46.94	47.05	23.23	57.28	62.14	39.02	55.28
Granite-3.3-it	6.67	69.00	59.33	36.55	51.22	14.22	32.00	21.71	31.39	44.66	31.82	62.34	61.19	36.59	50.32
Internlm3-it	6.67	69.00	60.00	60.47	65.85	21.23	33.84	25.86	61.94	55.14	33.84	61.99	67.24	36.87	50.97

Table 7: The performance of each model on each dataset under the routing for performance-oriented setting. The deep red and light red markers denote the best and second-best results, respectively.

Model	Mathematics		Code		Knowledge				IF	Tool Use
	AIME	LMB.	LCB.	SWE.	GPQA	HLE	MP.	SQA.	AHARD.	Tau2.
Claude-v4	36.67	61.16	58.10	34.60	68.69	4.82	83.17	15.14	54.67	49.28
DeepSeek-R1	85.00	77.69	80.09	28.60	79.80	15.99	84.23	27.12	63.53	37.77
DeepSeek-V3	43.33	61.98	66.64	25.00	59.60	3.61	78.00	27.76	60.13	7.19
DS-V3.1-Tms	55.00	73.55	67.30	25.40	76.26	8.94	84.03	24.46	66.27	46.76
Gemini-Flash	63.33	73.55	60.00	18.20	66.16	7.78	80.93	29.82	55.53	32.37
Gemini-Pro	85.00	50.41	79.15	34.60	84.85	21.69	85.67	53.77	77.00	43.17
GLM-4.6	81.67	63.64	63.13	21.80	75.25	15.52	80.33	25.68	66.40	53.96
GPT-5	88.33	79.34	86.45	15.80	84.85	26.32	87.37	47.90	69.73	70.14
GPT-5-Chat	71.67	57.85	62.18	8.80	73.74	5.70	81.40	40.50	68.87	–
Intern-S1	48.33	68.60	49.19	7.80	65.66	8.85	81.97	14.77	67.07	32.73
Kimi-K2	65.00	71.90	67.30	23.40	73.23	6.12	80.30	29.03	71.07	48.92
Qwen3-235B	76.67	76.03	64.17	16.20	58.59	9.22	82.77	53.77	73.07	40.29
Qwen3-Thinking	86.67	52.89	78.39	21.40	80.30	7.78	79.47	48.27	73.93	53.60

Table 8: The performance of each model on each dataset under the routing for performance-cost setting. The deep red and light red markers denote the best and second-best results, respectively. Note that GPT-5-Chat has no score on the τ^2-Bench benchmark because this model does not support tool calling.

Model	Mathematics		Code		Knowledge				IF	Tool Use	Total
	AIME	LMB.	LCB.	SWE.	GPQA	HLE	MP.	SQA.	AHARD.	Tau2.	
Claude-v4	1.26	1.91	14.20	35.72	2.65	23.71	23.40	7.33	15.94	103.72	229.83
DeepSeek-R1	2.09	2.70	30.46	10.03	3.25	72.03	24.62	6.19	8.52	37.08	196.97
DeepSeek-V3	0.15	0.17	1.53	2.31	0.18	1.93	1.76	0.48	0.80	6.92	16.22
DS-V3.1-Tms	0.17	0.19	1.19	2.30	0.25	4.35	1.67	0.80	1.08	17.38	29.39
Gemini-Flash	2.25	1.90	6.16	3.94	2.65	71.07	10.96	0.63	5.49	5.80	110.86
Gemini-Pro	9.18	10.96	142.59	80.32	16.59	277.72	117.18	41.12	34.63	38.25	768.53
GLM-4.6	2.79	4.02	2.32	5.10	5.35	100.59	30.60	26.54	2.51	27.61	207.41
GPT-5	4.57	4.09	54.92	24.27	7.94	137.59	36.17	57.25	27.85	61.67	416.31
GPT-5-Chat	0.08	1.57	4.02	11.52	1.85	18.41	13.44	2.37	6.97	–	60.22
Intern-S1	0.46	0.50	7.10	2.86	0.81	11.09	5.59	3.78	1.99	13.28	47.45
Kimi-K2	0.57	0.60	3.88	4.79	0.63	2.35	3.86	0.46	1.70	27.90	46.75
Qwen3-235B	0.31	0.23	1.04	1.36	0.07	4.51	1.80	0.20	0.74	11.67	21.94
Qwen3-Thinking	1.13	1.04	14.74	8.88	2.42	17.61	25.64	12.04	8.02	28.44	119.95

Table 9: Model inference cost comparison across different datasets for performance–cost tradeoff setting. The deep red and light red markers denote the lowest and second-lowest costs, respectively. Note that GPT-5-Chat has no score on the τ^2-Bench benchmark because this model does not support tool calling.

Model	Abbr.	Params
DeepHermes-3-Llama-3-8B-Preview (Teknium et al., 2024)	DH-Llama3-it	8B
DeepSeek-R1-0528-Qwen3-8B (Guo et al., 2025a)	DS-Qwen3	8B
DeepSeek-R1-Distill-Qwen-7B (Guo et al., 2025a)	DS-Qwen	7B
Fin-R1 (Liu et al., 2025)	Fin-R1	7B
GLM-Z1-9B-0414 (GLM et al., 2024)	GLM-Z1	9B
Intern-S1-mini (Bai et al., 2025)	Intern-S1-mini	8B
Llama-3.1-8B-Instruct (Grattafiori et al., 2024)	Llama-3.1-it	8B
Llama-3.1-8B-UltraMedical (Zhang et al., 2024)	UltraMedical	8B
Llama-3.1-Nemotron-Nano-8B-v1 (Bercovich et al., 2025)	Llama-Nemo	8B
MiMo-7B-RL-0530 (Xiaomi, 2025)	MiMo-RL	7B
MiniCPM4.1-8B (Team et al., 2025b)	MiniCPM	8B
NVIDIA-Nemotron-Nano-9B-v2 (Basant et al., 2025)	NVIDIA-Nemo	9B
OpenThinker3-7B (Guha et al., 2025)	OpenThinker	7B
Qwen2.5-Coder-7B-Instruct (Hui et al., 2024)	Qwen-Coder	7B
Qwen3-8B (Yang et al., 2025)	Qwen3-8B	8B
Cogito-v1-preview-llama-8B (Deep Cogito, 2025)	Cogito-v1	8B
Gemma-2-9b-it (Team et al., 2024)	Gemma-2-it	9B
Glm-4-9b-chat (GLM et al., 2024)	Glm-4-chat	9B
Granite-3.3-8b-instruct (Granite Team, IBM, 2025)	Granite-3.3-it	8B
Internlm3-8b-instruct (Cai et al., 2024)	Internlm3-it	8B

Table 10: Model pool for the performance-oriented setting: open-source models around 7B parameters.

Model	Abbr.	Input Price	Output Price
Claude-sonnet-4(Anthropic, 2025)	Claude-v4	$3.00/1M	$15.00/1M
Gemini-2.5-flash(Gemini Team, 2025)	Gemini-Flash	$0.30/1M	$2.50/1M
Gemini-2.5-pro(Gemini Team, 2025)	Gemini-Pro	$1.25/1M	$10.00/1M
GPT-5-chat(OpenAI, 2025b)	GPT-5-Chat	$1.25/1M	$10.00/1M
GPT-5-medium(OpenAI, 2025b)	GPT-5	$1.25/1M	$10.00/1M
Qwen3-235b-a22b-2507(Yang et al., 2025)	Qwen3-235B	$0.09/1M	$0.60/1M
Qwen3-235b-a22b-thinking-2507(Yang et al., 2025)	Qwen3-Thinking	$0.30/1M	$2.90/1M
Deepseek-v3-0324(Liu et al., 2024a)	DeepSeek-V3	$0.25/1M	$0.88/1M
Deepseek-v3.1-terminus(Liu et al., 2024a)	DS-V3.1-Tms	$0.27/1M	$1.00/1M
Deepseek-r1-0528(Guo et al., 2025a)	DeepSeek-R1	$0.50/1M	$2.15/1M
GLM-4.6(Zeng et al., 2025)	GLM-4.6	$0.60/1M	$2.20/1M
Kimi-k2-0905(Team et al., 2025a)	Kimi-K2	$0.50/1M	$2.00/1M
Intern-s1(Bai et al., 2025)	Intern-S1	$0.18/1M	$0.54/1M

Table 11: Model pool for the performance–cost setting: flagship models.

Dataset	Abbrev.	Category	Metrics	Size
		Routing for Performance-Oriented Datasets		
AIME	AIME	Mathematics	Accuracy, 0-shot	60
MATH500 (Lightman et al., 2023)	M500.	Mathematics	Accuracy, 0-shot	500
MathBench (Liu et al., 2024b)	MBen.	Mathematics	Accuracy, 0-shot	150
MBPP (Austin et al., 2021)	MBPP	Code	Pass@1, 0-shot	974
HumanEval (Chen et al., 2021a)	HE.	Code	Pass@1, 0-shot	164
LiveCodeBench (Jain et al., 2024)	LCB.	Code	Pass@1, 0-shot	1055
KORBench (Ma et al., 2024)	KOR.	Logic	Accuracy, 3-shot	1250
Knights and Knaves (Xie et al., 2024)	K&K.	Logic	Accuracy, 0-shot	700
BBH (Suzgun et al., 2022)	BBH	Logic	Accuracy, 3-shot	1080
MMLU-Pro (Wang et al., 2024)	MP.	Knowledge	Accuracy, 0-shot	1000
GPQA (Rein et al., 2024)	GPQA	Knowledge	Accuracy, 0-shot	198
FinQA (Chen et al., 2021b)	FQA.	Knowledge	Accuracy, 0-shot	1147
MedQA (Jin et al., 2021)	MQA.	Knowledge	Accuracy, 0-shot	1273
EmoryNLP (Byrkjeland et al., 2018)	Emory.	Affective	Accuracy, 0-shot	697
MELD (Poria et al., 2019)	MELD	Affective	Accuracy, 0-shot	1232
Total				**11,480**
		Routing for Performance–Cost Datasets		
AIME	AIME	Mathematics	Accuracy, 0-shot	60
LiveMathBench (Liu et al., 2024c)	LMB.	Mathematics	Accuracy, 0-shot	121
LiveCodeBench (Jain et al., 2024)	LCB.	Code	Pass@1, 0-shot	1055
SWE-Bench (Jimenez et al., 2023)	SWE.	Code	Pass@1, 0-shot	500
GPQA (Rein et al., 2024)	GPQA	Knowledge	Accuracy, 0-shot	198
HLE (Phan et al., 2025)	HLE	Knowledge	LLM as judge†, 0-shot	2158
MMLU-Pro (Wang et al., 2024)	MP.	Knowledge	Accuracy, 0-shot	3000
SimpleQA (Wei et al., 2024)	SQA.	Knowledge	LLM as judge†, 0-shot	4326
ArenaHard (Li et al., 2024)	AHARD.	Instruction Following (IF)	LLM as judge†, 0-shot	750
τ^2-Bench (Barres et al., 2025)	TAU2.	Tool Use	Success Rate, 0-shot	278
Total				**12,446**

Table 12: Detailed information of the datasets (with abbreviations). "LLM as judge"† means that we use an auxiliary LLM to score model outputs with the **official prompts**: for HLE and SimpleQA we adopt `o3-mini` as the judge model, and for ArenaHard we adopt `deepseek-v3-0324`.

Model	Mathematics			Code			Logical			Knowledge				Affective		Avg
	AIME	M500.	MBen.	MBPP	HE.	LCB.	KOR.	K&K.	BBH	MP.	GPQA	FQA.	MQA.	Emory.	MELD	
DH-Llama3-it	0.00	28.00	38.22	54.74	51.60	12.81	32.73	15.45	51.11	36.53	31.00	50.87	57.17	38.53	48.16	36.46
DS-Qwen3	76.67	94.67	92.00	36.86	25.60	64.35	55.88	60.19	89.07	71.27	59.33	70.56	79.90	38.53	50.97	64.39
DS-Qwen	40.00	87.87	88.00	53.65	42.40	40.95	44.12	50.33	69.51	47.60	34.33	64.05	34.55	29.35	33.62	50.69
Fin-R1	15.56	75.33	66.67	68.81	76.40	7.51	33.69	18.29	61.91	48.93	26.33	68.58	61.73	36.81	51.46	47.87
GLM-Z1	62.22	95.60	94.22	61.57	60.80	60.13	53.48	48.15	83.83	66.33	53.67	68.41	70.26	34.13	44.43	63.82
Intern-S1-mini	78.89	91.20	63.56	45.87	42.00	25.24	58.27	79.34	84.07	54.80	44.00	70.96	68.27	37.19	49.78	59.56
Llama-3.1-it	8.89	49.47	44.44	61.84	61.60	17.03	23.95	13.46	60.56	48.67	25.00	53.60	67.33	33.75	46.86	41.10
UltraMedical	0.00	41.20	25.33	54.74	52.40	12.68	14.37	14.88	41.42	41.47	26.00	54.36	69.63	33.65	42.27	34.96
Llama-Nemo	46.67	90.53	47.11	61.64	60.40	44.98	30.28	23.03	26.98	41.27	34.00	30.84	40.63	29.92	38.22	43.10
MiMo-RL	0.00	24.67	12.89	57.06	50.40	5.99	30.01	2.65	32.84	20.73	4.00	17.19	30.31	28.11	39.08	23.73
MiniCPM	71.11	93.60	95.56	34.81	30.80	61.96	30.01	60.28	87.78	69.13	51.33	67.71	65.86	32.98	50.81	60.25
NVIDIA-Nemo	71.11	93.60	95.56	43.62	46.80	64.73	56.15	70.62	87.22	69.13	50.33	71.20	74.03	38.53	46.22	65.26
OpenThinker	40.00	85.33	63.56	12.29	8.00	43.28	11.28	17.16	44.57	42.20	28.33	61.44	53.25	24.00	32.49	37.81
Qwen-Coder	4.44	64.80	66.67	75.49	76.40	27.63	34.91	20.76	59.14	44.47	34.67	63.47	49.27	39.10	51.73	47.50
Qwen3-8B	73.33	93.47	92.89	53.92	58.40	67.57	54.28	75.36	83.83	67.73	54.00	74.51	77.80	39.58	53.51	68.01
Cogito-v1	0.00	50.00	53.33	51.81	69.20	15.90	42.68	23.41	71.23	57.80	38.67	63.36	64.97	38.82	54.05	46.35
Gemma-2-it	2.22	46.27	52.44	61.77	67.20	16.85	35.92	9.19	62.84	53.67	28.00	65.91	64.29	39.58	51.24	43.83
Glm-4-chat	3.33	48.00	47.56	64.30	72.00	16.72	37.41	13.18	48.02	46.60	25.67	57.32	60.99	38.53	55.78	42.36
Granite-3.3-it	10.00	68.40	57.33	36.11	48.80	13.12	33.37	21.90	30.37	44.13	31.00	62.02	60.73	36.33	49.51	40.21
Internlm3-it	8.89	69.87	56.44	60.61	62.40	20.82	34.06	26.16	63.21	56.00	33.00	61.85	66.75	36.52	50.05	47.11
Dataset Oracle	78.89	95.60	95.56	75.49	76.40	67.57	58.27	79.34	89.07	71.27	59.33	74.51	79.90	39.58	55.78	73.10
Oracle	87.78	98.67	100.00	93.65	96.80	79.68	77.43	99.91	99.32	93.93	97.00	88.21	98.06	76.48	87.62	91.64
Random Router	15.56	70.67	68.44	54.06	58.40	34.51	36.94	35.55	61.42	51.47	43.33	58.13	59.95	35.85	47.51	48.79
RouterDC	80.00	89.47	57.78	63.00	69.60	49.72	48.21	76.87	71.67	55.80	33.34	66.84	66.33	39.71	51.57	61.33
EmbedLLM	81.11	94.67	93.78	69.42	73.20	64.16	57.76	77.35	88.03	69.20	63.00	71.11	75.81	38.37	51.62	71.24
GraphRouter	77.78	92.93	93.78	66.62	74.00	65.49	58.83	80.76	85.74	66.73	53.33	73.10	74.66	39.23	51.30	70.29
MODEL-SAT	70.00	93.87	93.34	73.11	76.40	67.26	61.07	79.24	89.75	70.00	61.33	73.33	79.32	38.09	52.11	71.88
Avengers	71.11	93.20	94.22	76.04	75.20	65.17	61.49	77.25	90.31	70.00	59.33	73.98	79.42	37.80	54.54	71.94

Table 13: The performance of all base models and routing methods on each dataset under the routing for performance-oriented setting. All results are computed on the 30% test split and averaged over five random seeds used in Experiments. The deep red and light red markers denote the best and second-best results, respectively.

Model	Mathematics		Code		Knowledge				IF	Tool Use	Avg
	AIME	LMB.	LCB.	SWE.	GPQA	HLE	MP.	SQA.	AHARD.	Tau2.	
Claude-v4	41.11	61.62	56.34	35.33	71.33	4.79	83.76	15.55	54.08	48.81	47.27
DeepSeek-R1	84.44	77.84	78.42	28.40	80.67	16.46	85.11	27.70	64.81	34.76	57.86
DeepSeek-V3	45.56	67.57	65.17	25.20	62.67	3.83	78.87	28.18	60.59	6.43	44.41
DS-V3.1-Tms	56.67	76.76	65.55	25.47	77.00	9.08	84.49	24.98	66.14	45.00	53.11
Gemini-Flash	60.00	71.89	58.49	20.27	62.67	7.69	81.33	29.86	55.45	30.71	47.84
Gemini-Pro	82.22	46.49	77.54	34.53	84.00	21.43	86.73	54.51	77.17	43.33	60.80
GLM-4.6	86.67	65.41	61.51	21.60	76.67	15.16	81.04	26.36	66.18	55.48	55.61
GPT-5	87.78	82.16	85.68	16.67	84.33	26.57	88.49	48.47	69.90	69.52	65.96
GPT-5-Chat	73.33	57.84	60.32	10.13	75.00	5.74	82.13	40.69	67.02	-	52.47
Intern-S1	50.00	69.19	47.63	7.60	67.67	9.05	82.13	14.92	67.06	30.48	44.57
Kimi-K2	70.00	73.51	66.12	24.40	74.67	5.93	80.47	29.24	73.45	48.10	54.59
Qwen3-235B	80.00	78.92	62.78	16.40	58.67	9.48	83.80	54.33	74.20	38.81	55.74
Qwen3-Thinking	82.22	53.51	76.78	21.33	80.00	7.69	80.36	49.03	75.09	50.48	57.65
Dataset Oracle	87.78	82.16	85.68	35.33	84.33	26.57	88.49	54.51	77.17	69.52	69.16
Oracle	93.33	84.32	91.61	66.00	97.00	50.96	95.20	85.12	99.47	93.57	85.66
OpenRouter	43.33	57.30	59.12	11.33	76.33	13.25	86.02	42.91	57.4	-	49.67
Random Router	66.67	67.57	66.69	23.33	74.00	10.65	83.16	35.16	68.03	39.29	53.45
HybridLLM	80.00	78.91	64.11	16.26	60.67	11.05	83.73	54.24	74.26	52.38	57.56
FrugalGPT	80.00	81.08	65.74	15.87	59.67	23.92	83.67	55.22	84.80	47.14	59.71
RouteLLM	87.78	81.62	85.74	16.67	84.33	26.14	88.44	55.30	81.16	69.52	67.67
GraphRouter (CF)	80.00	78.92	63.41	16.40	58.67	9.44	83.71	54.19	85.07	40.00	56.98
GraphRouter (BL)	81.11	78.92	72.68	16.40	58.67	17.65	83.73	54.88	84.62	48.33	59.70
GraphRouter (PF)	85.55	80.54	84.41	28.80	85.33	20.28	85.49	57.15	86.31	64.05	67.79
Avengers (α=0.00)	45.56	66.49	62.71	16.40	58.67	3.92	79.33	54.21	68.01	0.00	45.53
Avengers (α=0.05)	80.00	78.38	65.55	23.60	77.00	8.33	84.20	54.30	69.42	2.86	54.36
Avengers (α=0.10)	80.00	78.38	65.36	23.73	77.00	8.98	84.11	54.30	69.34	31.67	57.29
Avengers (α=0.15)	80.00	78.38	65.30	23.60	77.00	8.98	84.09	54.30	69.03	34.52	57.52
Avengers (α=0.20)	80.00	78.38	67.70	23.60	77.00	9.07	84.22	54.35	69.07	34.52	57.79
Avengers (α=0.25)	80.00	78.38	71.42	24.13	77.00	9.10	84.27	54.36	69.78	36.19	58.46
Avengers (α=0.30)	80.00	78.38	75.02	24.13	77.00	9.35	84.33	54.33	69.73	44.76	59.70
Avengers (α=0.35)	80.00	78.38	76.28	23.87	77.00	9.75	84.62	54.36	70.00	49.52	60.38
Avengers (α=0.40)	80.00	78.38	76.59	24.13	77.33	22.07	84.62	54.35	70.66	54.05	62.22
Avengers (α=0.45)	80.00	78.92	76.66	24.67	78.00	25.52	85.02	54.41	70.35	56.90	63.05
Avengers (α=0.50)	80.00	78.38	77.22	24.67	78.67	25.96	86.49	54.44	69.65	61.67	63.71
Avengers (α=0.55)	82.22	77.30	79.18	24.67	79.67	26.08	87.31	54.56	69.38	63.57	64.39
Avengers (α=0.60)	82.22	77.30	81.14	28.53	79.67	26.14	87.49	54.70	70.53	65.00	65.27
Avengers (α=0.65)	84.44	77.30	82.46	33.07	80.00	26.17	87.44	55.15	71.11	65.00	66.21
Avengers (α=0.70)	84.44	77.30	84.79	34.53	81.00	26.14	87.71	55.27	71.95	64.76	66.79
Avengers (α=0.75)	83.33	78.38	85.05	34.53	82.33	26.11	87.69	56.66	71.46	65.24	67.08
Avengers (α=0.80)	85.56	81.08	85.74	34.53	82.33	26.11	87.84	56.73	72.39	66.43	67.88
Avengers (α=0.85)	85.56	80.54	85.74	35.47	84.33	26.23	88.04	56.80	73.94	67.38	68.40
Avengers (α=0.90)	85.56	80.54	85.74	34.80	84.33	26.20	88.00	57.18	74.65	67.86	68.49
Avengers (α=0.95)	85.56	80.54	85.80	35.47	83.67	26.30	87.98	57.20	75.22	68.10	68.58
Avengers (α=1.00)	85.56	80.54	85.11	35.20	83.33	26.36	88.16	57.55	75.88	68.10	68.58

Table 14: The performance of all base models and routing methods on each dataset under the routing for performance–cost tradeoff setting. All results are computed on the 30% test split and averaged over five random seeds used in Experiments. The deep red and light red markers denote the best and second-best results, respectively. Note that GPT-5-Chat and OpenRouter have no score on the τ^2-Bench benchmark because this model does not support tool calling. For GraphRouter, we report three configurations—Performance First (PF), Balance (BL), and Cost First (CF)—as employed in the original paper. For Avengers-Pro, we report 21 configurations obtained by varying the performance coefficient from 0 to 1 in increments of 0.05 due to space constraints.

Model	Mathematics		Code		Knowledge				IF	Tool Use	Total
	AIME	LMB.	LCB.	SWE.	GPQA	HLE	MP.	SQA.	AHARD.	Tau2.	
Claude-v4	0.37	0.57	4.35	10.89	0.85	6.98	7.04	2.21	4.88	32.43	70.56
DeepSeek-R1	0.62	0.83	9.50	3.05	0.97	21.31	7.46	1.85	2.53	11.49	59.60
DeepSeek-V3	0.04	0.05	0.47	0.70	0.05	0.57	0.53	0.14	0.24	2.20	5.00
DS-V3.1-Tms	0.05	0.06	0.37	0.70	0.07	1.27	0.50	0.24	0.32	5.45	9.05
Gemini-Flash	0.70	0.65	2.08	1.21	0.78	20.56	3.35	0.17	1.57	1.81	32.88
Gemini-Pro	2.80	3.34	43.08	24.09	4.84	81.62	35.50	12.31	10.18	12.13	229.89
GLM-4.6	0.75	1.27	0.70	1.55	1.59	30.38	9.30	8.01	0.76	8.59	62.90
GPT-5	1.36	1.43	17.65	7.31	2.30	40.27	10.87	17.09	8.32	18.99	125.60
GPT-5-Chat	0.02	0.46	1.23	3.52	0.57	5.35	4.06	0.71	2.12	-	18.04
Intern-S1	0.14	0.16	2.18	0.85	0.24	3.20	1.68	1.12	0.59	3.88	14.02
Kimi-K2	0.16	0.19	1.22	1.47	0.18	0.68	1.16	0.14	0.52	8.65	14.37
Qwen3-235B	0.09	0.07	0.32	0.42	0.02	1.33	0.55	0.06	0.22	3.85	6.93
Qwen3-Thinking	0.30	0.32	4.45	2.68	0.69	5.39	7.70	3.61	2.39	8.90	36.43
Dataset Oracle	1.36	1.43	17.65	10.89	2.30	40.27	10.87	12.31	10.18	18.99	126.26
Oracle	0.12	0.09	3.56	1.86	0.13	9.78	0.65	1.07	0.26	5.52	23.04
OpenRouter	1.80	2.90	4.22	4.44	2.49	37.47	18.06	2.98	7.95	-	82.31
Random Router	0.27	0.76	6.90	4.80	1.03	16.35	6.98	3.81	2.56	8.72	52.17
HybridLLM	0.14	0.29	1.71	0.96	0.20	4.68	0.84	0.08	0.92	8.82	18.65
FrugalGPT	0.10	0.11	4.08	4.45	0.29	37.05	0.67	3.23	0.41	8.77	59.15
RouteLLM	1.36	1.40	17.65	7.31	2.30	40.67	10.92	2.26	8.41	18.99	111.26
GraphRouter (CF)	0.09	0.07	0.33	0.42	0.02	1.34	0.56	0.06	0.22	3.84	6.94
GraphRouter (BL)	0.13	0.07	9.57	0.49	0.02	23.40	0.56	1.22	0.42	7.41	43.29
GraphRouter (PF)	2.29	0.94	22.13	13.22	4.22	58.14	20.92	6.66	5.75	16.05	150.32
Avengers (α=0.00)	0.04	0.05	0.31	0.42	0.02	0.57	0.47	0.06	0.30	0.00	2.23
Avengers (α=0.05)	0.09	0.07	0.37	0.71	0.07	1.16	0.51	0.06	0.30	0.06	3.40
Avengers (α=0.10)	0.09	0.07	0.37	0.71	0.07	1.30	0.51	0.06	0.28	0.94	4.40
Avengers (α=0.15)	0.09	0.07	0.37	0.71	0.07	1.30	0.51	0.06	0.28	1.10	4.57
Avengers (α=0.20)	0.09	0.07	1.15	0.71	0.07	1.32	0.52	0.06	0.28	1.10	5.37
Avengers (α=0.25)	0.09	0.07	2.06	0.71	0.07	1.39	0.58	0.06	0.28	1.88	7.21
Avengers (α=0.30)	0.09	0.07	3.62	0.71	0.07	1.65	0.61	0.06	0.43	3.81	11.13
Avengers (α=0.35)	0.09	0.07	4.32	1.04	0.07	2.44	0.74	0.10	0.61	4.95	14.44
Avengers (α=0.40)	0.09	0.07	4.46	1.51	0.20	29.20	1.21	0.12	1.08	7.10	45.05
Avengers (α=0.45)	0.09	0.08	5.01	1.65	0.32	37.98	1.53	0.14	1.46	7.60	55.86
Avengers (α=0.50)	0.42	0.49	5.75	1.93	0.64	39.14	3.29	0.19	1.98	8.93	62.75
Avengers (α=0.55)	0.53	0.63	9.29	1.65	0.77	39.42	4.38	0.23	2.77	9.25	68.90
Avengers (α=0.60)	0.53	0.63	11.94	5.26	0.77	39.66	4.69	0.96	3.34	10.41	78.17
Avengers (α=0.65)	0.62	0.76	13.53	9.32	1.40	39.84	5.80	1.53	3.91	10.63	87.34
Avengers (α=0.70)	0.62	0.78	16.45	10.38	1.69	40.06	7.49	1.71	4.49	11.55	95.21
Avengers (α=0.75)	0.81	0.83	16.60	10.38	2.06	40.06	7.94	2.71	4.80	13.52	99.71
Avengers (α=0.80)	1.19	1.20	17.40	10.38	2.06	40.67	8.59	3.54	5.20	14.19	104.41
Avengers (α=0.85)	1.19	1.24	17.40	12.61	2.30	40.91	9.53	4.77	5.86	14.34	110.14
Avengers (α=0.90)	1.19	1.24	17.40	15.72	2.30	41.08	9.71	5.88	6.39	15.11	116.01
Avengers (α=0.95)	1.19	1.24	17.27	17.18	2.91	41.43	12.17	6.60	6.69	16.94	123.62
Avengers (α=1.00)	1.19	1.25	18.73	19.74	4.39	41.54	14.57	7.58	7.06	16.93	132.98

Table 15: Inference cost comparison for all base models and routing methods across different datasets. All results are computed on the 30% test split and averaged over five random seeds used in Experiments. The deep red and light red markers denote the lowest and second-lowest costs, respectively. Note that GPT-5-Chat and OpenRouter have no score on the τ^2-Bench benchmark because this model does not support tool calling. For GraphRouter, we report three configurations—Performance First (PF), Balance (BL), and Cost First (CF)—as employed in the original paper. For Avengers-Pro, we report 21 configurations obtained by varying the performance coefficient from 0 to 1 in increments of 0.05 due to space constraints.

Provider	Models
OpenAI	gpt-5, gpt-5-mini, gpt-5-nano, gpt-4.1, gpt-4.1-mini, gpt-4.1-nano, gpt-4o-mini, chatgpt-4o-latest
Anthropic	claude-3.5-haiku, claude-opus-4-1, claude-sonnet-4-0, claude-3-7-sonnet-latest
Google	gemini-2.5-pro, gemini-2.5-flash
Mistral	mistral-large-latest, mistral-medium-latest, mistral-small-latest, mistral-nemo
X.AI	grok-3, grok-3-mini, grok-4
DeepSeek	deepseek-r1
Meta-Llama	llama-3.1-70b-instruct, llama-3.1-405b-instruct
MistralAI	mixtral-8x22b-instruct
Perplexity	sonar
Cohere	command-r-plus, command-r

Table 16: Model pool supported by the `openrouter/auto` routing method on OpenRouter. The available models may change over time; we use the model pool as of December 1, 2025.

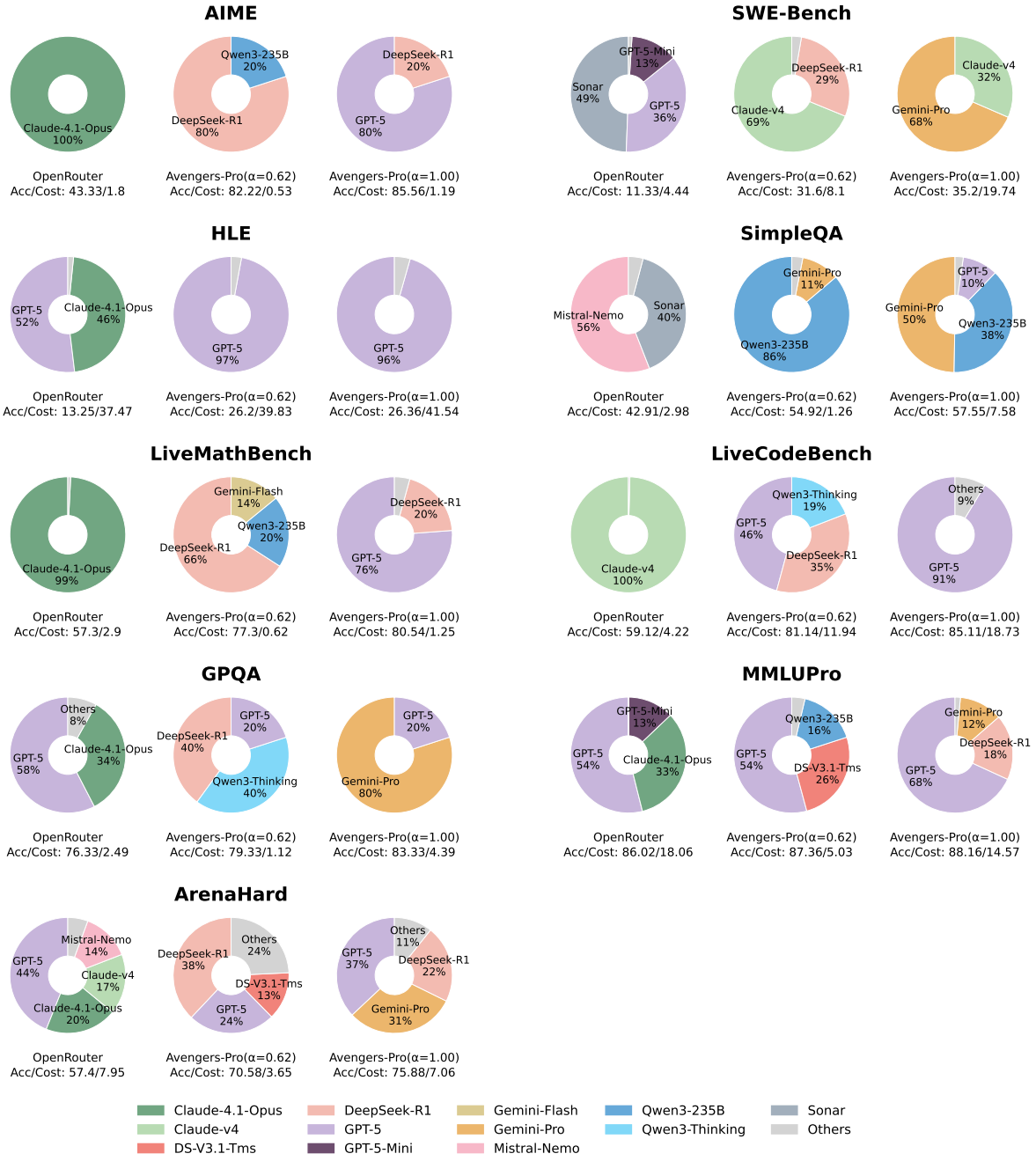

Figure 9: Routing distributions of OpenRouter and Avengers-Pro; Avengers-Pro (cost-matched: $\alpha = 0.62$, highest-accuracy: $\alpha = 1.00$); Models selected in less than 5% of queries are grouped into "Others".